Rich and Robust Bio-Inspired Locomotion Control for Humanoid Robots

THÈSE Nº 7879 (EPFL) et Nº 612 (UCL) (2017)

PRÉSENTÉE LE 30 AOÛT 2017

À L'ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

À LA FACULTÉ DES SCIENCES ET TECHNIQUES DE L'INGÉNIEUR

LABORATOIRE DE BIOROBOTIQUE

PROGRAMME DOCTORAL EN ROBOTIQUE, CONTRÔLE ET SYSTÈMES INTELLIGENTS

ET

À L'UNIVERSITÉ CATHOLIQUE DE LOUVAIN

INSTITUTE OF MECHANICS, MATERIALS, AND CIVIL ENGINEERING

ÉCOLE DOCTORALE EN SCIENCES DE L'INGÉNIEUR ET ART DE BÂTIR ET URBANISME

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Nicolas Benoît Dominique VAN DER NOOT

acceptée sur proposition du jury:

Prof. K. Aminian, président du jury
Prof. A. Ijspeert, Prof. R. Ronsse, directeurs de thèse
Prof. P. Fisette, rapporteur
Prof. H. Geyer, rapporteur
Prof. S. Micera, rapporteur

UCL Université catholique de Louvain

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Suisse
2017

Respice finem
"Do not lose sight of the aim"
– van der Noot family's motto –

To *my parents*
who always believed in me,

my grand-father Paul
who was an example for me,

and my Mary
who stole the key to my heart.

Acknowledgements

This thesis has been carried out in two distinct laboratories: the Center for Research in Mechatronics (CEREM) at the Université catholique de Louvain (UCL, Belgium) and the Biorobotics Laboratory (BioRob) at the École Polytechnique Fédérale de Lausanne (EPFL, Switzerland). During the four years of my Ph.D. thesis, I received help, advice and support from many amazing people, both in Belgium and in Switzerland. I would like to thank them in the next paragraphs.

First of all, I want to thank my two advisors: Prof. Renaud Ronsse (UCL) and Prof. Auke Jan Ijspeert (EPFL) for giving me the opportunity to pursue my Ph.D. in their laboratories (respectively CEREM and BioRob). During my whole thesis, they shared their expertise, provided useful advice and helped me with publications writing. I especially appreciated the balance they found between guidance and freedom to explore my own paths. I also want to thank them for their involvement in the administrative process to turn my thesis into a joint Ph.D. between UCL and EPFL.

I would like to express my gratitude to Professors Paul Fisette, Hartmut Geyer, Silvestro Micera and Kamiar Aminian for agreeing to be part of my thesis jury. I also want to thank the Belgian F.R.S.-FNRS (Aspirant #16744574) and the European Community's Seventh Framework Programme (WALK-MAN) to have funded my Ph.D. thesis.

During this thesis, I had the opportunity to supervise the master theses of François Heremans, Adrien De Coninck, Bruno Somers and Philippe Greiner, as well as the semester project of Matthew Harding. They all did a wonderful job, and I want to thank them for that. Most of their work is presented in the following chapters of this thesis.

I also wish to express my gratitude to all my colleagues and friends from my two laboratories (i.e. CEREM and BioRob). In my case, I was twice lucky to find so nice people contributing to the good atmosphere prevailing in these two places. On top of this, they were all ready to help in many different ways, which was very useful throughout all my thesis.

In particular, I am deeply grateful to the members of the Walk-Man and the Robotran teams. I was very happy to collaborate with them and I think that it resulted in fruitful collaborations, both for the progress made in the Walk-Man work packages and for the developments of the Robotran simulator.

The work presented here was initiated during my master thesis. Therefore, I would like to thank Allan Barrea with whom I had the pleasure to work on this master thesis. I am also grateful to Jesse van den Kieboom for all the help he provided us during that time, and for the tools he developed for us, and that I still used during my whole Ph.D. project.

Acknowledgements

Last but not least, I would like to express my gratitude to my friends and family for the good times together, for their love and for their support. In particular, I want to address my special thanks and love to my wife Marie-Sophie. I met her at the beginning of my Ph.D. and I had the extreme pleasure of marrying her by the end of this thesis. She was always there to help me in the difficult moments and to push me to give the best of myself. Our story started during this Ph.D. thesis, and I know that the next chapters will be wonderful !

Louvain-la-Neuve,
11th May 2017

Nicolas Van der Noot

Abstract

Bipedal locomotion is a challenging task in the sense that it requires to maintain dynamic balance while steering the gait in potentially complex environments. Yet, humans usually manage to move without any apparent difficulty, even on rough terrains. This requires a complex control scheme which is far from being understood.

In this thesis, we take inspiration from the impressive human walking capabilities to design neuromuscular controllers for humanoid robots. More precisely, we control the robot motors to reproduce the action of virtual muscles commanded by stimulations (i.e. neural signals), similarly to what is done during human locomotion. Because the human neural circuitry commanding these muscles is not completely known, we make hypotheses about this control scheme to simplify it and progressively refine the corresponding rules.

This thesis thus aims at developing new walking algorithms for humanoid robots in order to obtain fast, human-like and energetically efficient gaits. In particular, gait robustness and richness are two key aspects of this work. In other words, the gaits developed in the thesis can be steered by an external operator, while being resistant to external perturbations. This is mainly tested during blind walking experiments on COMAN, a 95 cm tall humanoid robot. Yet, the proposed controllers can be adapted to other humanoid robots.

In the beginning of this thesis, we adapt and port an existing reflex-based neuromuscular model to the real COMAN platform. When tested in a 2D simulation environment, this model was capable of reproducing stable human-like locomotion. By porting it to real hardware, we show that these neuromuscular controllers are viable solutions to develop new controllers for robotics locomotion.

Starting from this reflex-based model, we progressively iterate and transform the stimulation rules to add new features. In particular, gait modulation is obtained with the inclusion of a central pattern generator (CPG), a neural circuit capable of producing rhythmic patterns of neural activity without receiving rhythmic inputs.

Using this CPG, the 2D walker controllers are incremented to generate gaits across a range of forward speeds close to the normal human one. By using a similar control method, we also obtain 2D running gaits whose speed can be controlled by a human operator. The walking controllers are later extended to 3D scenarios (i.e. no motion constraint) with the capability to adapt both the forward speed and the heading direction (including steering curvature). In parallel, we also develop a method to automatically learn stimulation networks for a given task and we study how flexible feet affect the gait in terms of robustness and energy efficiency.

Abstract

In sum, we develop neuromuscular controllers generating human-like gaits with steering capabilities. These controllers recruit three main components: (i) virtual muscles generating torque references at the joint level, (ii) neural signals commanding these muscles with reflexes and CPG signals, and (iii) higher level commands controlling speed and heading.
Most of these developments are performed on a simulated model of the COMAN robot, in which hardware limits are taken into account. More precisely, actuators are modeled and noise components are incorporated, such that the computed torque references differ from the actual ones, as would happen on a real robotic device. Moreover, only sensory information available to the real platform is used in the simulation environment. Using this framework, the algorithms developed in this thesis could thus be tested on real robots.
Interestingly, these developments target humanoid robots locomotion but can also be used to better understand human locomotion. In particular, the recruitment of a CPG during human locomotion is still a matter open to debate. This question can thus benefit from the experiments performed in this thesis. In turn, these developments could possibly be valuable in the fields of prostheses, orthoses, exoskeletons, rehabilitation robotics and computer graphics animation.

Key words: Locomotion Control, Biologically-Inspired Robots, Humanoid Robots, Central Pattern Generator, Sensory Feedback, Gait Modulation.

Résumé

Les humains parviennent à se déplacer avec aisance, même lorsqu'ils sont confrontés à des terrains irréguliers (par exemple rocailleux). Pourtant, la locomotion bipède est une tâche complexe, qui requiert de garantir l'équilibre dynamique tout en adaptant la démarche pour guider le corps dans des environnements parfois difficiles d'accès. Ce contrôle subtil est encore loin d'être compris.

Dans cette thèse, nous tentons de reproduire les caractéristiques de la marche humaine afin d'élaborer des contrôleurs neuro-musculaires de locomotion à destination de robots humanoïdes. Plus précisément, nous contrôlons les moteurs de ces robots afin qu'ils reproduisent l'action de muscles virtuels. L'activité musculaire résulte alors de signaux nerveux appelés stimulations. Étant donné que les circuits nerveux humains en charge de moduler la marche ne sont pas connus, nous émettons des hypothèses à leur sujet, en vue de les simplifier et de progressivement affiner les règles de contrôle qui y correspondent.

L'objectif principal de cette thèse est donc de développer de nouveaux algorithmes de marche pour des robots humanoïdes, afin d'obtenir des démarches rapides, efficientes d'un point de vue énergétique et proches de celles des humains. En particulier, la robustesse et la richesse de la locomotion sont deux aspects fondamentaux de ce travail. En d'autres mots, les démarches obtenues au sein de cette thèse peuvent être modulées par un opérateur externe, tout en étant robustes à des perturbations externes. Ces développements sont principalement testés au cours d'expériences de locomotion sans retour visuel effectuées avec COMAN, un robot humanoïde d'une taille de 95 cm. Les contrôleurs développés ici pourraient toutefois être adaptés à d'autres robots humanoïdes.

Dans les premiers chapitres de cette thèse, nous adaptons un modèle neuro-musculaire pré-existant, et l'implémentons sur le robot réel COMAN. Initialement développé dans un environnement de simulation en 2D et contrôlé au moyen de réflexes, celui-ci est capable de générer une locomotion stable et similaire à celle des humains. En le portant sur un vrai robot, nous montrons que les contrôleurs neuro-musculaires sont des solutions viables afin de développer de nouveaux algorithmes de locomotion pour les robots humanoïdes.

Sur base de ce modèle, nous adaptons progressivement les règles de stimulations et en développons de nouvelles, afin d'augmenter les capacités du marcheur. En particulier, nous incluons un CPG (pour *central pattern generator*, en français *oscillateurs spinaux*), c'est-à-dire un circuit nerveux capable de produire des activités cycliques sans recevoir d'entrées cycliques.

En utilisant ce CPG, les contrôleurs de marche 2D sont incrémentés afin d'obtenir des démarches capables d'atteindre un éventail de vitesses semblable à celui utilisé le plus fréquem-

ment par l'être humain. En utilisant une stratégie similaire, nous obtenons également des robots capables de courir tout en modulant leur vitesse. Le contrôleur de marche est par la suite développé pour permettre au robot de marcher dans un environnement 3D (c'est à dire sans contrainte), tout en étant capable d'adapter à la fois sa vitesse d'avance et sa direction, et donc aussi la courbure de ses trajectoires. En parallèle, nous développons une méthode afin d'apprendre automatiquement des réseaux de stimulations exercés à réaliser une tâche particulière. Finalement, nous étudions aussi comment la robustesse et l'efficacité énergétique du marcheur sont impactées par l'utilisation de prothèses au lieu de pieds rigides.
Pour résumer, nous développons des contrôleurs capables de produire des locomotions similaires à celles des humains, tout en permettant de moduler les déplacements. Ces contrôleurs utilisent principalement trois composants: (i) des muscles virtuels en charge de générer des couples au niveau des articulations, (ii) des stimulations pour contrôler ces muscles sur base de réflexes et des signaux générées par un CPG, et (iii) des commandes de haut niveau qui adaptent la vitesse et la direction du marcheur.
La plupart de ces développements sont réalisé au sein d'un simulateur reproduisant le comportement du robot COMAN, tout en incluant certaines contraintes liées au hardware. Ainsi, les moteurs sont modélisés et une composante de bruit est ajoutée. De cette manière, les couples de référence et les vrais couples au niveau articulaire diffèrent, de manière similaire à ce qui se passerait sur un vrai robot. De plus, seules les informations qui peuvent être obtenues par le biais des capteurs du robot sont utilisées. Dès lors, les contrôleurs développés au sein de cette thèse pourraient être testés sur de véritables robots.
Ces développements ont pour but principal de faire marcher des robots humanoïdes. Néanmoins, ils peuvent aussi être utilisés pour améliorer notre compréhension de la marche humaine. En particulier, l'utilisation d'un CPG durant la locomotion humaine est une hypothèse encore controversée. Les expériences effectuées au sein de cette thèse permettent donc d'alimenter le débat. De plus, ces développements pourraient être utiles dans d'autres domaines, tels que ceux des prothèses, des orthèses, des exosquelettes, de la robotique de réhabilitation et de l'animation générée par ordinateur.

Mots clefs: Contrôle de la Locomotion, Robots Bio-Inspirés, Robots Humanoïdes, Oscillateurs Spinaux, Feedback Sensoriel, Modulation de la Démarche.

Contents

List of Figures

List of Tables

1 Introduction

Locomotion is a key aspect of our everyday life. From walking to running, it is at the heart of many of our activities. Locomotion is also critical to most animals for survival, in order to grab food, to chase preys or to escape predators. Similarly, a locomotion deficiency can be a huge handicap for humans, usually requiring many environmental adaptations (like wheelchair ramps) and help from healthy people.

As evolution progressed, all species adapted their locomotion to their environment and their needs, resulting in diverse motions like swimming, crawling, walking or flying. Most of them developed amazing locomotion skills, moving efficiently and robustly over rough terrains, while exhibiting impressive adaptation capabilities to steer their gait.

Despite the apparent ease with which most animals move in complex environments, the neuro-musculo-skeletal system in charge of producing these behaviors is far from being trivial. Locomotion is usually achieved through the coordinated action of hundreds of muscles, commanded by a complex neural circuitry regulating these muscles. In particular, this muscle control must be able to adapt the body motion to higher level commands, while incorporating feedback originating from muscles and skin afferents as well as from some senses like vision (Rossignol et al., 2006).

Due to this high complexity, inherited from millions of years of evolution, these locomotion skills are far from being understood, in particular from the control point of view. Reproducing similar performances on mechanical devices is therefore a huge challenge. For more than half a century, robots were minimalist devices, far from achieving the complex performances observed in nature. During that time, robotics primary focus was on industrial applications, mainly targeting highly repetitive and high precision positioning tasks, in contrast to the behavior richness of biological systems (Schaal, 2007).

In this industrial context, wheeled mobile robots are particularly appealing because they can efficiently and robustly move on flat ground, without complex stability controller. However, they are usually restricted to operate in environments designed for their mobility. In contrast,

legged robots have the unique ability to step onto or over obstacles or unsafe footholds, therefore allowing them to traverse terrains that would be impassable to wheeled mobile robots (Chestnutt et al., 2005). Therefore, it is not surprising that biological evolution favored legged animals over wheeled locomotion.

Interestingly, there are currently many interactions between biology and robotics. Previously, these interactions went mainly in one direction, with roboticists taking inspiration from biology in terms of morphologies, modes of locomotion, and control mechanisms. In particular, many robot bodies were directly inspired by animal morphologies, from snake robots, then quadruped robots, to humanoid robots. However, robotics is now also exploring biological questions, with robots being used as scientific tools to test biological hypotheses (Ijspeert, 2008). In turn, this could possibly be valuable in the fields of prostheses, orthoses, exoskeletons and rehabilitation robotics.

In a world designed for humans, using humanoid robots is valuable because their body, roughly similar to ours, is potentially adapted to environments designed for our needs, like ladders or stairs (Schaal, 2007). Interestingly, they also offer the possibility to manipulate tools designed to comply with human dexterity, so that these tools do not need to be adapted for these robots (Fitzpatrick et al., 2016). This is particularly appealing in contexts where the robot is expected either to take over a human laborious duty, or to co-work in synergy with human operators. Also, it might be easier for people to interact with humanoid robots, rather than with robots with a non-human shape (Wahde and Pettersson, 2002). Indeed, people are accustomed to work with other people, so that human–robot interactions can be enhanced by taking advantage of the communication channels that already exist between people (Fitzpatrick et al., 2016).

In this thesis, we mainly target the development of controllers to achieve human-like locomotion with biped robots. More precisely, we take inspiration from biology and real human experiments to develop neuromuscular controllers to command virtual muscles, in turn generating walking and running gaits. In particular, we put a special emphasis on gaits richness and robustness. In other words, the resulting gaits can be steered through high-level commands to provide rich locomotion behaviors, while being robust to environmental uncertainties. Interestingly, these developments can possibly also be used to investigate the control of real human locomotion, which is far from being understood (Minassian et al., 2017).

1.1 Biped robots locomotion

Bipedal walking with robots can be achieved using different methods. The corresponding gaits can first be divided into two main categories: static walking and dynamic walking. Basically, static stability means that the vertical projection of the center of mass (COM) always stays within the support polygon formed by the feet (Hobbelen and Wisse, 2007). In other words, the walker is in static equilibrium at every moment of the gait, so that it would not fall if its joints were suddenly frozen during its motion. Even if this locomotion is straightforward to

obtain, static walking drastically limits the walking speed and the achievable step lengths. Consequently, the biped can only cross over small obstacles, thus limiting the interest of using humanoid robots.

In contrast, the COM can escape the support polygon during dynamic walking, such that the biped is only stable in motion. Making a robot walk dynamically is more difficult than implementing static walking since the inertia effects play a prominent role and must therefore be taken into account. Nonetheless, dynamic walking offers much better performances, for instance in terms of speed, energy efficiency and behavior richness (e.g. crossing over larger obstacles). Importantly, it is also the way humans walk and is thus the walking type addressed by this thesis.

The most popular methods developed to achieve dynamic walking rely on the zero-moment point (ZMP) as an indicator of gait feasibility (Vukobratovic and Borovac, 2004). In short, the ZMP can be thought of as the generalization of the COM in a dynamic context. Just as keeping the COM in the support polygon is a necessary and sufficient condition to keep static balance, maintaining the ZMP in the support polygon is a sufficient condition to ensure dynamic equilibrium. However, as explained in Section 1.2, ensuring this ZMP constraint is not a necessary condition to maintain dynamic stability. More information about the ZMP computation can be found in (Van der Noot and Barrea, 2014), see Appendix B.

The ZMP can thus be used to generate walking patterns guaranteeing dynamic stability at every moment during the gait. Many locomotion experiments were successfully conducted using this indicator, for example with ASIMO (Chestnutt et al., 2005) (see Figure 1.1a) or with the HRP-2 platform (Kaneko et al., 2002) (see Figure 1.1b).

However, there are several shortcomings related to these ZMP-based bipedal controllers. In most gaits, the generated locomotion pattern looks quite unnatural: the waist vertical position is maintained low, the knees are permanently bent and the feet are kept parallel to the ground (Kurazume et al., 2005; Sardain and Bessonnet, 2004b). Even if faster speeds can be obtained than for static walkers, the ZMP-based controllers still generate much slower speeds than the ones achieved by healthy humans displaying the same morphology. This also results in poor energy inefficiency (Dallali, 2011). Moreover, computing the ZMP position requires lots of computing operations and a perfect knowledge of the robot dynamics and its environment. Finally, the ZMP computation is deteriorated when applied to human-like gaits involving important feet strikes, as illustrated in Appendix B. In fact, it is important to note that real human walking does not ensure ZMP-based stability (i.e. the ZMP stability criterion is frequently violated during human locomotion).

Other concepts frequently used to achieve dynamic walking include the inverted pendulum model (IPM). In its most basic version, the IPM models the biped robot as a single point mass with force vectors at the feet level, in order to produce desired motions for the COM (Faraji et al., 2014b). For instance, this was applied to generate a foot-step planner for the Atlas robot (Faraji et al., 2014a) (see Figure 1.1c).

(a) ASIMO (b) HRP-2 (c) Atlas

Figure 1.1: Humanoid robots achieving dynamic walking: (a) ASIMO developed by Honda (image taken from http://asimo.honda.com/); (b) the HRP-2 platform developed by the Intelligent Systems Research Institute (AIST, Japan) (image taken from http://www.plasticpals.com); (c) the Atlas robot developed by the American robotics company Boston Dynamics (image taken from http://www.bostondynamics.com/robot_Atlas.html).

Strategies relying on the computation of the ZMP and/or the IPM commonly recruit inverse kinematics/dynamics methods to obtain position or torque commands at the joint level. These inverse methods usually require to avoid singular configurations, thus preventing the leg to reach full extension during the stance phase (Kurazume et al., 2005). This has a direct impact on the energetic consumption, since a bended knee requires to maintain a torque balancing the body static and dynamic forces. Some contributions however managed to address this problem (Ogura et al., 2006).

Many other algorithms exist to achieve dynamic walking like the virtual model control framework, which recruits virtual components to generate virtual forces on robot systems (Pratt et al., 2001). Yet, we do not cover them to focus on more bio-inspired methods, as detailed in the next sections.

1.2 Limit cycle walking

In contrast to stability criteria like the ZMP, the limit cycle walking concept relaxes the need to guarantee the local stability at every instant during the gait. More formally, limit cycle walking is a nominally periodic sequence of steps that is stable as a whole but not locally stable at

every instant in time. It consists of a series of repetitions of a closed trajectory in state space (i.e. a limit cycle). The nominal motion is stable as a whole because neighboring trajectories eventually, over the course of multiple steps, approach the nominal trajectory (Hobbelen and Wisse, 2007).

This strategy allows the biped to walk much more efficiently by taking better advantage of inertia effects and corresponds to the way humans walk. Indeed, the ZMP stability criterion is sufficient to ensure dynamic stability but not necessary, as it introduces unnecessary constraints to guarantee local stability at every moment. In particular, humans frequently deport the ZMP at the border of the feet (i.e. violating the ZMP stability criterion), such that stability is recovered at next foot strike.

(Quasi-)passive walkers are successful implementations of this concept. When started on a shallow slope, a passive walker can reach a steady gait similar to the human one, despite the lack of active control or energy input, as demonstrated by the biped from (McGeer, 1990). Active energy introduction with basic control and sensing capabilities can be used to obtain quasi-passive walkers, therefore capable of walking on flat terrains like the Cornell powered biped of (Collins and Ruina, 2005).

Although they display human-like gait patterns and require zero (or little) energetic consumption, (quasi-)passive walkers are usually limited to very controlled environments, since they usually lack steering control and can poorly resist to external perturbations like obstacles or collisions.

1.3 Reflex-based approach

Another avenue to explore the limit cycle concept is through the development of bio-inspired controllers. Bio-inspired approaches aim to discover and capture essential ideas that underpin a biological system, so that the same ideas can be technologically implemented. Regarding walking control, these approaches mainly consist in reproducing human locomotion mechanisms through the coordinated actuation of muscle groups.

The seminal paper of (Geyer and Herr, 2010) developed a bipedal model actuated by a human-like neuromuscular model. More precisely, a musculo-skeletal model generated forces, later converted to joint torques, in order to achieve locomotion. This was tested in simulation for 2D scenarios (i.e. without lateral degree of freedom) using a seven-segment human model. By recruiting reflexes to drive these muscles, human-like walking patterns could emerge, generating leg kinematics and dynamics similar to the ones measured on healthy human subjects. Furthermore, the generated muscle activation patterns appeared to be similar to human walking experiments. In addition, the simulated viscoelastic properties of these virtual muscles provided robustness to external perturbations.

This result was further extended, notably in (Song and Geyer, 2012) to modulate the walking speed, in (Desai and Geyer, 2013) to control swing leg placement, in (Song and Geyer, 2013) to add lateral control and to maintain stability for 3D scenarios, and in (Song and Geyer, 2015b) to control running gaits. Most of these developments are summarized in (Song and Geyer, 2015a). Interestingly, part of this model was also adapted to control a powered ankle-foot prosthesis (Eilenberg et al., 2010), thus further enhancing the bio-inspired framework. Finally, this approach was applied to provide realistic motions of 3D animated characters, ranging from humans (Wang et al., 2012) to biped animals like ostriches (Geijtenbeek et al., 2013).

Importantly, the biped embodiment used in this thesis (i.e. the COMAN humanoid platform, presented in Section 2.1) is equipped with DC motors at the joint level, far from the human biological muscles. Therefore, we consider virtual muscles, implemented as a set of equations called the Hill muscle model (Hill, 1938). More information is provided in Chapter 3.

1.4 Central pattern generators for locomotion

In the previous section, the muscles were only commanded by reflexes, i.e. by feedback signals. Gait modulation could then be achieved by adapting different reflex parameters (stimulation gains, reference angles, offset values...). An alternative bio-inspired gait modulation strategy requires the addition of a central pattern generator (CPG). CPGs are neural circuits capable of producing rhythmic patterns of neural activity without receiving rhythmic inputs. They feature valuable properties like distributed control, redundancies handling, and locomotion modulation using simple control signals (Ijspeert, 2008). In contrast to reflexes, CPGs generate descending (i.e. feed-forward) signals. Indeed, these circuits can function in vitro, even when isolated from the brain and from the motor and sensory apparatus of the limbs, as observed during the locomotion of decerebrated animals (Shik et al., 1966). Yet, they can still be modulated by sensory information, for instance to synchronize the gait with environmental interactions.

CPGs have already been identified in both invertebrate and vertebrate animals (Ijspeert, 2008). In particular, isolated spinal cord experiments, mainly from newborn rats and mice, were widely used to better understand the organization of CPGs in the mammalian spinal cord (Kiehn and Butt, 2003). In (Ijspeert et al., 2007), a spinal cord model of an amphibious salamander could explain the ability of salamanders to switch between swimming and walking through the modulation of signals driving the implemented CPG. Finally, (McCrea and Rybak, 2008) reported several experiments of fictive locomotion in decerebrated cats that could be reproduced with a particular CPG architecture.

Despite these CPG identifications in many vertebrates, their involvement in human locomotion is still a matter open to discussion (Dimitrijevic et al., 1998). In particular, there is no direct equivalent of fictive locomotion in humans, in contrast to the cats experiments aforementioned. Therefore, it is not possible to provide the same degree of evidence regarding the recruitment of CPGs during human locomotion as in animals (Minassian et al., 2017).

A first path to explore the CPG involvement in human muscles control is to study the spontaneous rhythmic activities observed in individuals following spinal cord injury (SCI). These kinds of studies were first reported by (Bussel et al., 1988), demonstrating that rhythmic movements of the trunk and the lower limbs could be generated within the spinal cord in humans deprived of supraspinal control. Also, (Calancie, 2006) reported that six individuals with SCI developed involuntary and rhythmic contractions in some of their leg muscles. Interestingly, the related motion patterns appeared to share many features with those associated with a CPG for stepping.

Yet, it is important to note that the spontaneous emergence of similar movements in persons with SCI is extremely rare. Another exploration path consists in applying excitatory drives to patients with paralyzed limbs, in order to generate rhythmic movements. For example, (Dimitrijevic et al., 1998) could induce rhythmic phases of the lower limbs in paraplegic subjects with complete SCI. This was achieved by applying non-patterned electrical stimulations to the posterior structures of the lumbar spinal cord. Their findings suggest that an externally controlled sustained electrical stimulation of the spinal cord could replace the tonic drive generated by the brain, in order to obtain locomotor-like activity. In other words, human lumbar spinal cord circuits appear to be capable of converting tonic inputs into rhythmic outputs, which could be considered as further evidence for the existence of a CPG in the human spinal cord.

Gait adaptation is another activity which could potentially benefit from the inclusion of a CPG, for instance regarding speed modulation. More formally, the walking speed is related to the cycle of rhythmic activity and is mainly adapted by changing the duration of the stance phase, while the swing phase duration remains relatively constant (Frigon, 2012). In (Danner et al., 2015), experiments on subjects with motor-complete SCI suggested that the human lumbar spinal cord circuits could use a constant, repetitive input to generate burst signal combinations capable of obtaining a large range of locomotor outputs. Possible gait modulations could thus possibly emerge from similar strategies.

However, all these results obtained from individuals with SCI do not give information on whether the circuits generating the observed rhythmic activities are indeed recruited during active walking (Minassian et al., 2017). A first clue to investigate this hypothetical CPG recruitment during walking is to compare the EMGs of children and adults to the ones of newborn babies, as reported in the study of (Dominici et al., 2011). Indeed, irregular stepping can be evoked in neonates before disappearing four to six weeks after birth. Stepping then usually reappears six to eight months postnatally, evolving into intentional walking. During the neonates irregular stepping experiments, (Dominici et al., 2011) reported that no specific activation pattern was observed on foot contact. This could be due to immature sensory mechanisms in newborn babies.

When comparing these human neonate EMG signals to the ones of different animal species, similar activation patterns were observed. This could possibly indicate that locomotion in

several animal species (including human) recruits common primitives, probably related to a common ancestral neural network. However, it is not clear if the circuits generating these patterns in newborn human babies, likely CPGs, are preserved in the neural control of adult walking. Yet, (Dominici et al., 2011) reported that some EMG patterns identified in newborn human babies seemed to be retained and adapted in adult locomotion, while new ones appeared (with more powerful descending and sensory influences on the locomotion control). This suggests that at least part of muscle flexion and extension synergies are controlled by CPG signals. A more in-depth review of the possible recruitment of CPGs during human locomotion experiments can be found in (Minassian et al., 2017).

Another path to investigate the potential recruitment of CPGs during human locomotion is through the development of numerical simulations or during tests on biped robots. In particular, these tools allow researchers to test conceptual models of locomotor circuits, by comparing the generated locomotor patterns to recorded human ones, and to explore possible modifications to better match biological data. These models are also useful to suggest new experiments and to predict their outcomes (Ijspeert, 2008).

For instance, different computational models recruiting a CPG showed that these oscillators could play a major role in human locomotion. In particular, the seminal work of (Taga, 1994) greatly contributed to scientific progresses in this area by showing the ability of a biped model to reach limit-cycle walking, while adapting its gait to changing environments in real-time. Interestingly, the introduction of time delays resulted in chaotic behaviors (i.e. non-periodic gaits obtained), similar to the gaits of patients with neural deficits.

Other contributions also showed the importance of CPGs in biped locomotion. For instance, (Aoi and Tsuchiya, 2005) achieved robust walking with a biped robot by recruiting nonlinear oscillators, both in numerical simulations and with a hardware platform. In (Paul et al., 2005), a neuromuscular model used a CPG as central element to investigate the effects of a spinal cord injury on locomotor abilities. Finally, (Dzeladini et al., 2014) incremented the controller of (Geyer and Herr, 2010) with the inclusion of a CPG. This CPG was used as a feedback predictor, in order to modulate the forward speed.

Importantly, modeling efforts investigating the potential role of CPG in human locomotion ubiquitously display their complex intertwining with feedback mechanisms (Rossignol et al., 2006). This is coherent with Kuo's framework combining feedback signals (i.e. reflexes) and feed-forward pathways (mainly coming from CPGs) in the control of a periodic task (Kuo, 2002).

1.5 Thesis context and overview

As outlined in the previous sections, there is an increasing interest in bringing humanoid robots in our day-to-day life, in particular due to their morphology potentially adapted to our environments. This was particularly outlined during the DARPA Robotics Challenge (DRC),

a contest where robots had to demonstrate capabilities to assist humans in responding to natural and man-made disasters (Johnson et al., 2016).

However, humanoid robots locomotion capabilities are still far from reaching the ones of healthy humans, especially regarding energy efficiency, robustness and behavior richness. This is mainly due to the intrinsic constraints which are inherent to more traditional controllers like the ones using inverse kinematics/dynamics modules. For instance, avoiding singularity configurations or ensuring dynamic balance at every moment of the gait are inherent constraints commonly found in traditional walkers. Yet, these constraints prevent these robotic walkers from achieving performances similar to the human ones.

Therefore, the walking algorithms developed in this thesis rely on bio-inspired concepts, in order to take advantage from human skills, obtained as the result of millions of years of evolution. This relaxes unnecessary dynamic walking constraints while copying mechanisms identified in human locomotion. However, because real human locomotion is far from being understood, it is not possible to copy all human control mechanisms. The approach followed in the this thesis is thus to recruit both simulation and hardware experiments to test and/or validate hypotheses about real human locomotion.

This thesis uses the reflex-based neuromuscular model of (Geyer and Herr, 2010) (presented in Section 1.3) as a starting point. The main purpose is to extend it, in order to obtain rich and robust gaits for 3D walking scenarios. Interestingly, the model of (Geyer and Herr, 2010) already provides some intrinsic robustness. The extension to 3D walking as well as the gait modulation can be achieved by using only reflexes, as demonstrated by (Song and Geyer, 2015b). However, we chose to explore another path in this thesis by recruiting a CPG on top of the reflex signals.

The main interest of CPGs regarding the thesis final goal is to offer to modulate the gait by adapting a reduced number of parameters. This is potentially valuable to obtain rich locomotion behaviors. Indeed, neuromuscular models are commonly tuned (possibly through an optimization process) to reach a single gait (Geyer and Herr, 2010; Wang et al., 2012; Geijtenbeek et al., 2013). In this thesis, we want to show that it is possible to steer the walker on-line, by modulating a few parameters according to higher-level commands.

More precisely, we aim at modulating both the walker forward speed and its heading direction (including steering curvature). As presented in Chapter 10, this can be obtained by adapting two scalar inputs (i.e. speed and heading references). This framework thus allows an operator to steer the gait by using straightforward commands. These two modulations are central to most gait locomotions but are far from representing the complete panel of human locomotion behaviors. Other motions could therefore be studied like stairs climbing, hopping or sidestepping. However, the two gait modulations addressed in this thesis show that it is possible to adapt neuromuscular models to obtain rich and robust bio-inspired locomotion control for humanoid robots.

The present thesis was funded by the *F.R.S.-FNRS - Fonds de la Recherche Scientifique* and by the *Walk-Man* EU collaborative project, which is detailed in Section 1.5.1. This thesis was carried out in two different laboratories (both of them being part of the Walk-Man consortium): the Center for Research in Mechatronics (CEREM) located at the Université catholique de Louvain (UCL, Belgium) and the Biorobotics Laboratory (BioRob) located at the École Polytechnique Fédérale de Lausanne (EPFL, Switzerland).

1.5.1 Walk-Man project

The Whole-body Adaptive Locomotion and Manipulation (Walk-Man) project is a 4 years integrated project funded by the European Commission. Its final purpose is to develop an anthropomorphic robotic platform capable of operating outside the laboratory space in unstructured environments as a result of natural and man-made disasters. This EU project covers hardware and software design, including the developments of the locomotion algorithms targeted by this thesis.

More precisely, the Walk-Man project aims at improving the current locomotion capabilities of humanoid systems in cluttered space, while maintaining balance against external disturbances. To achieve this purpose, the robot could potentially exploit all its limbs to obtain whole-body dynamics motion. In other words, the robot could possibly use contacts between its upper-body and the surrounding environment to improve its stability during locomotion. In parallel, manipulation capabilities are supposed to be improved through the design of new flexible hands combining robustness and adaptability.

The new robot developed in the frame of this EU project is called Walk-Man (i.e. the same name as the project), and is depicted in Figure 1.2.

This robot is expected to demonstrate the following skills: (i) dexterous, powerful manipulation, (ii) robust balanced locomotion and (iii) physical sturdiness. In particular, the robust balanced locomotion is at the heart of this thesis.

However, the Walk-Man platform is not the only robot used in this EU project. Indeed, other robots are supposed to be used to develop control algorithms, that could potentially be adapted to the Walk-Man platform. In this thesis, all the controllers are tested on the COMAN platform, a robot developed within the

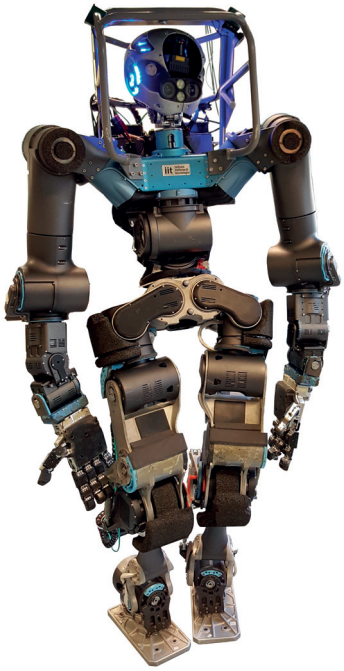


Figure 1.2: Walk-Man robot.

AMARSI European project and presented in Section 2.1.

More information about the Walk-Man EU project can be found at the following address: https://www.walk-man.eu/.

The Walk-Man consortium is composed of the five following partners:

- Istituto Italiano di Tecnologia (IIT) - coordinator
- École Polytechnique Fédérale de Lausanne (EPFL)
- Universita di Pisa (UNIPI)
- Karlsruhe Institute of Technology (KIT)
- Université catholique de Louvain (UCL)

1.5.2 Questions being addressed in the thesis

This thesis targets the development of neuromuscular controllers capable of steering humanoid robots in 2D or 3D environments. Therefore, it also offers to answer some questions about the resulting features of robots walking with these bio-inspired controllers.

The main questions that we want to answer in this thesis are the following:

A *Are neuromuscular controllers viable solutions to control the locomotion of humanoid robots ?*
Neuromuscular models are mainly studied in simulation to better understand human locomotion. This offers to improve the design of prostheses, orthoses and exoskeletons, but also to help for surgical operations related to body deformities affecting the walking gait. In contrast, very few developments are ported to real robots, favoring more traditional approaches closer to industrial robotics (like inverse kinematics/dynamics). In Chapter 4, the neuromuscular controller of (Geyer and Herr, 2010) is ported to a real robotic platform, in order to study its viability on real hardware. On top of that, many experiments are performed in simulation, in order to study both the resulting robustness and richness of the developed algorithms.

B *How do the biped gaits adapt on rough terrain and when subjected to unknown perturbations ?*
Robustness is an important topic to bring robots in our everyday environment, and even more to move inside devastated locations. The muscles viscoelastic properties already

provide some intrinsic robustness. The combined action of CPG and reflex signals also contributes to this robustness. In this thesis, distal muscles (i.e. close to the grounds, and therefore more impacted by environmental contacts) are mainly driven by reflexes, in order to better adapt to external forces. In many chapters of this thesis, the biped faces unknown perturbations (rough ground, random pushes...) while performing blind walking. This allows to measure the corresponding robustness.

C *How can we modulate the biped gait by means of a minimal set of high-level parameters ?*
This question is at the heart of many chapters of this thesis and aims at achieving rich locomotion behaviors through proper gait modulation. As previously mentioned, two gait modulations are studied in this thesis: forward speed modulation and the heading orientation (with the corresponding curvature). Chapters 6 and 7 study forward speed modulation in 2D, respectively for walking and for running gaits. Chapter 9 extends speed modulation to 3D walking scenarios, while Chapter 10 studies right/left steering. In all these chapters, a CPG is added to modulate the gait with a small set of parameters.

D *Which range of motion can be achieved with proper parameters modulation ?*
This question is related to the previous one and aims at quantifying the range of achievable forward speeds, as well as the maximum curvature obtained by using the bio-inspired controllers developed in this thesis.

E *Can the neuromuscular controllers work with flexible feet, and how do they affect the gait in terms of robustness and energy efficiency ?*
Most developments in this thesis are performed with the rigid feet used for the hardware experiments of Chapter 4. Human feet in contrast are flexible. Therefore, Chapter 5 studies the effects of replacing these rigid feet by flexible ones. More precisely, prostheses designed for children are characterized and reproduced in simulation. The neuromuscular controller is then tested using these prostheses and the resulting gaits are compared to the ones obtained with rigid feet.

F *Which method can be designed to automatically learn neural stimulations for a specific task ?*
There is no exact mathematical background to drive the developments of the stimulation computation rules, both for reflexes and for CPG signals. Most of these rules are elaborated based on human and stimulation experiments, by progressively refining and adapting the neural circuitry model (i.e. CPG and reflexes). This approach results in efficient and lightweight (i.e. with a low computational process) controller rules to drive the locomotion, but is difficult to generalize to more diverse motions. In Chapter 8, we investigate a new method to automatically learn stimulation patterns according to a specific task. This is studied for the following task: moving the COM to a desired position, while resisting to random pushes.

G *Do humans recruit a central pattern generator to control their locomotion ?*
Last but not least, this thesis develops bio-inspired locomotion controllers, which in turn can be used to investigate real human locomotion. Indeed, the neural control scheme used during human walking or running is far from being understood. In particular, the recruitment of a CPG during human locomotion is still a matter open to discussion (Dimitrijevic et al., 1998; Minassian et al., 2017). By introducing a CPG to drive the locomotion of a bipedal model (starting in Chapter 6), this work offers to evaluate the added value of this neural circuitry in gait generation and modulation.

1.5.3 Structure of the thesis

This thesis is structured according to the publications generated during the Ph.D. project. More specifically, each chapter of the core part of the thesis (i.e. from Chapter 3 to Chapter 10) corresponds to one publication. Therefore, the text of these chapters is adapted from the related papers, but with the purpose to stay as close as possible to the submitted or peer-reviewed versions.

The chapters ordering follows the following key thread. We start from the reflex-based model of (Geyer and Herr, 2010) and adapt it to test its performance on a real robot, as well as its adaptability to walk with different foot morphologies. The next chapters progressively increment the neuromuscular controller with the addition of a central pattern generator (CPG) and new rules, in order to finally obtain walking gaits in 3D environments, with steering (speed and heading) capabilities.

We now detail the different chapters. Chapter 2 overviews the general methods. This includes the presentation of COMAN (i.e. the biped embodiment used to test the neuromuscular algorithms), its simulation environment modeling it, the optimization method and the general structure of the controllers presented in the different chapters.

In order to test the reflex-based model of (Geyer and Herr, 2010) on the real humanoid platform, real-time constraints must be fulfilled. The virtual muscles, being implemented as Hill-muscle models (Hill, 1938), are problematic in this regard. Therefore, Chapter 3 studies different strategies to overcome this issue. The selected strategy is then applied in all the remaining chapters.

With this new strategy, the reflex-based model from (Geyer and Herr, 2010) is ported to the real COMAN platform. This requires new adaptations, among which the development of a strategy to provide lateral balance to the robot. Indeed, this reflex-based algorithm was developed in a 2D simulation environment, i.e. without lateral degree of freedom. This artificial constraint must be replicated in the real hardware experiment. To this end, we develop a new upper-body controller to let a human operator provide lateral stability to COMAN, but with the smallest impact possible on the sagittal plane (i.e. the robot can still fall in the sagittal plane). These developments are detailed in Chapter 4, together with the resulting gait analyses.

In Chapter 5, the same reflex-based controller is tested with prosthetic feet, in contrast to the rigid feet used in the rest of the thesis. To this end, real child prostheses are characterized and later modeled in the simulation environment. After re-tuning of the reflex-based parameters (but without other adaptations in the controller rules), three walkers are compared, each one equipped with its own feet: (i) the prostheses model, (ii) rigid feet with the same shape as the prostheses and (iii) rigid flat feet. These comparisons mainly address the trade-off between energy efficiency and robustness on rough terrain.

The remaining chapters introduce a CPG in the neuromuscular model, mainly to control the proximal muscles (i.e. the muscles close to the hip) and to steer the gait with a reduced number of parameters to adapt. All these developments are performed in simulation, but with the possibility to port it to the real hardware. For instance, this restrains the inputs used in the controller to the sensory information available on the real platform.

Chapter 6 studies walking for 2D scenarios. In this chapter, the modulation of a limited number of parameters (most of them being CPG-related) can continuously adapt the biped forward speed in a range close to the human one (once scaled to the size of the robot). This affects both the step length and frequency, resulting in fast speed alterations. Similarly, Chapter 7 also reaches forward speed modulation in a 2D environment, but studies running instead of walking. For this purpose, some reflexes and CPG inputs are adapted to the targeted running gaits.

The final goal of the thesis is to obtain 3D walking gaits, i.e. to achieve autonomous lateral balance, but also to control the robot speed and heading. However, gait initiation is more complex to obtain in 3D environments, due to the possible loss of lateral balance during the first swing. To overcome this problem, the COM is initially moved on top of one foot of the biped before initiating the swing phase with the other foot. In Chapter 8, this is obtained with appropriate muscle stimulations. Interestingly, this chapter also offers to develop new methods to automatically learn the stimulation patterns for a required task. In particular, this is tested during upright posture, with the purpose to move the COM to a target position, when subject to external perturbations.

Using this new 3D initialization, Chapter 9 extends the controller of Chapter 6 to obtain 3D walking. In particular, a new lateral balance control is implemented. Yet, the stimulations acting in the other planes are also updated. Forward speed modulations can then be obtained when walking straight ahead. Importantly, the range of achievable speeds is not reduced compared to the 2D walking gaits obtained in Chapter 6.

The 3D straight walking controller of Chapter 9 is later incremented in Chapter 10 to obtain heading control. This includes the modulation of the steering curvature, while keeping the previously developed forward speed adaptation. In sum, this chapter presents the main task targeted by this thesis: achieving a rich and robust bio-inspired locomotion control for humanoid robots.

Finally, Chapter 11 concludes the thesis. In particular, the questions of Section 1.5.2 are answered in that last chapter. A summary of the thesis achievements and future possible work are also detailed.

2 General methods

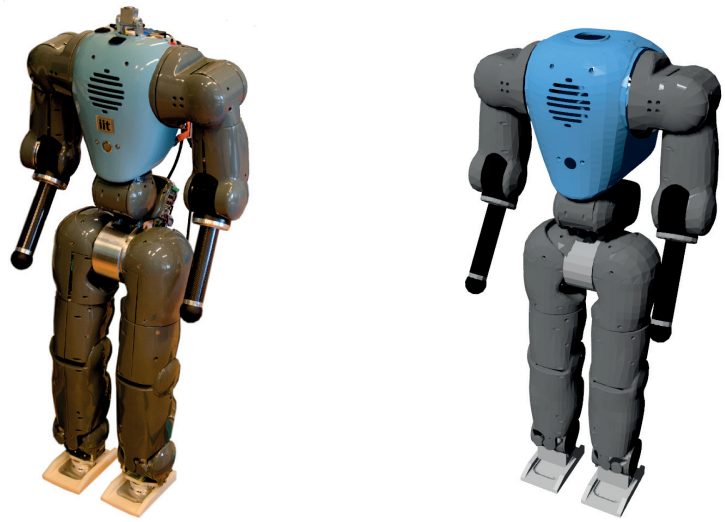
The developments performed in this thesis use the COMAN humanoid platform as embodiment. This robot is first introduced before presenting the simulation environment used to model it. We then give a brief overview of the optimization methods tuning the gaits unknown parameters. Finally, we detail the general structure of the controllers developed in the next chapters.

2.1 COMAN humanoid platform

The COmpliant huMANoid platform (COMAN) is a 25 degrees of freedom (DOFs) full-body humanoid robot, based on the iCub robot (Metta et al., 2008), and developed at the Italian Institute of Technology (IIT), within the AMARSI European project (Tsagarakis et al., 2013). This 95 cm tall robot, weighting 31 kg, is depicted in Figure 2.1a. As illustrated in this figure, COMAN is headless and does not have hands. In fact, among its 25 DOFs, 2 are located in the neck to provide pitch and roll motion to a possible head extension. Because all developments in this thesis are done without head, we consider that COMAN has only 23 DOFs.

In contrast to the actuation system of most humanoid platforms targeting high precision with non-backdrivable, stiff transmission systems, COMAN relies on embodied passive compliance. More precisely, passive compliance is added to 14 of its DOFs through the inclusion of series elastic actuators (SEA) (Pratt and Williamson, 1995).

The main interest of this soft actuation is to better adapt the biped's body to interaction uncertainties, similarly to what is observed on most biological systems. Therefore, these actuators are more suitable for safe human-robot interactions. Another potential benefit is to be able to store and release elastic energy, therefore decreasing the energetic consumption (Tsagarakis et al., 2013). Regarding walking, these added springs can also contribute to largely reduce the contact forces when the feet impact the ground (Dallali, 2011). Finally, compliant joints can improve the robustness of the robot by passively adapting the feet to uneven surfaces. Using compliance at the feet level (i.e. using flexible feet instead of rigid ones) also affects the



(a) real robot (b) simulated robot

Figure 2.1: COMAN humanoid platform in reality (panel (a)) and in the Robotran simulation environment (panel (b)).

biped robustness on uneven grounds, as studied in Chapter 5.

All leg sagittal joints feature SEA compliance. Because these joints are the main ones propelling the robot during walking or running, this added compliance highly impacts the biped locomotion. Other joints including this SEA compliance are the sagittal arms ones, the shoulders lateral ones, as well as most torso joints. Therefore, this SEA compliance is fully modeled in the simulation environment, as detailed in Section 2.2.

Regarding the robot sensors, all joints feature position encoders, as well as torque sensors specifically developed for COMAN. Moreover, custom-made 6 DOFs force/torque sensors are located below the ankle joint to measure the interaction forces between each foot and the ground (Tsagarakis et al., 2013). Finally, an inertial measurement unit (IMU) is attached to the robot waist, in order to measure the body orientation relative to the ground.

Importantly, the controllers developed in the next chapters only use inputs available to the real robot, even when these developments are only performed in simulation. Also, the COMAN on-board processing unit is composed of an embedded dual core Pentium PC104 unit running at 2.5GHz (Tsagarakis et al., 2013), whose controller is called with a time interval of 1 ms. All controller developments are thus performed with a frequency of 1 kHz. Consequently, all these controllers could potentially be tested on the real hardware.

Further information about COMAN can be found in (Dallali et al., 2013) and (Tsagarakis et al., 2013). Besides, (Tsagarakis et al., 2011) details the specific development of the lower body of COMAN with a description of the design of the cCub, the intermediate version of the robot between the iCub and the current version of COMAN.

2.2 COMAN model in Robotran

The direct implementation of the neuromuscular controllers targeted by this thesis on real hardware is not straightforward. Indeed, most of these controllers require an off-line optimization to tune the controllers unknown parameters (see Section 2.3), which is difficult to automate, and could damage the robot. Also, the developments performed before Chapter 8 restrain the biped motion to 2D walking gaits. In other words, the motion of the robot waist is constrained to stay in the sagittal plane, such that lateral falls are not possible. This artificial constraint is difficult to obtain on real hardware without important experimental setups, like a boom constraining the robot on a circular path (Geng et al., 2005).

Therefore, the thesis controller developments are mainly performed in simulation. More precisely, we use the Robotran simulation software (Fisette and Samin, 1993; Samin and Fisette, 2003; Docquier et al., 2013). It is a symbolic environment for multi-body systems developed at the Université catholique de Louvain (UCL). Robotran recruits a symbolic generator, in the sense that it generates the analytical form of the motion equations, for a given multi-body system. The Robotran simulation software is detailed in Appendix A, especially in Section A.2.

Regarding COMAN developments, the Robotran direct dynamics module is used to generate the symbolic equations of the robot dynamics. The COMAN platform in Robotran is depicted in Figure 2.1b. Because the thesis purpose is to develop controllers which could be ported to real hardware, a particular attention was paid to reduce the reality gap. In particular, this controller transfer to the real platform is presented in Chapter 4.

The actuators are implemented in simulation, with the purpose to reproduce the effects of the added SEA compliance. Their full implementation is reported in (Dallali et al., 2013) and in (Zobova et al., 2017). Then, a low-level controller is recruited to track position or torque references, by computing appropriate motor voltages. This is done by using a similar impedance controller as the one implemented on the real COMAN, being summarized in Appendix D and fully detailed in (Mosadeghzad et al., 2012). As revealed by our initial hardware tests, the torque measurements on the real robot are noisy, which impacts the torque tracking. Therefore, a uniform noise with a maximal amplitude of $0.4\,Nm$ is added to the actual torque measurements in the simulation environment. This corresponds to the noise level obtained from our measurements with the real platform. Consequently, the torques references computed by the controllers do not match the ones sent to the simulator, as would happen on a real robotic platform.

The ground contact model highly impacts the resulting gait. In this thesis, a mesh-based contact similar to the one of (Geyer and Herr, 2010) is implemented. In other words, ground contact forces are computed for each point of a mesh representing the feet sole. These forces are then gathered for each foot as resulting forces and torques, which are later sent to the simulator dynamics equations. The corresponding implementation is detailed in Appendix G.5.1. Interestingly, this ground implementation also works for uneven grounds. Finally, some simulation experiments measure the walker robustness to external pushes

coming from flying objects (see Chapter 9). The corresponding forces are computed based on the volume penetration model detailed in Appendix G.5.2.

2.3 Gait optimizations

The neuromuscular controllers developed in the next chapters recruit many open parameters to tune in order to obtain robust gaits with requested features (e.g. reaching a target forward speed). More precisely, these open parameters constitute the search space in which an optimal solution must be found, given some objective criteria. These criteria are quantitatively evaluated by a fitness function measuring the quality of a given set of parameters.

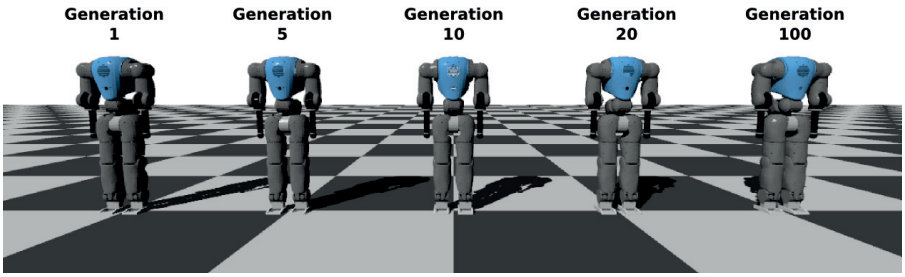
In the next chapters, different fitness functions are used, depending on the requested task. Except in Chapter 7 (targeting running gaits), the optimizer always maximizes a staged fitness function. In other words, different stages are ordered, such that each stage is unlocked once a condition related to the previous one is fulfilled.

In fact, most fitness functions in this thesis share the following template: (i) a first stage prevents the walker from staying in its initial upright position by rewarding the traveled distance; (ii) a second stage rewards the biped robustness proportionally to the walking time before a possible fall; (iii) a third stage constraints the walker to reach a target speed; and (iv) a last stage rewards energy efficiency. This initial fitness template is more detailed (and possibly incremented) in the related chapters.

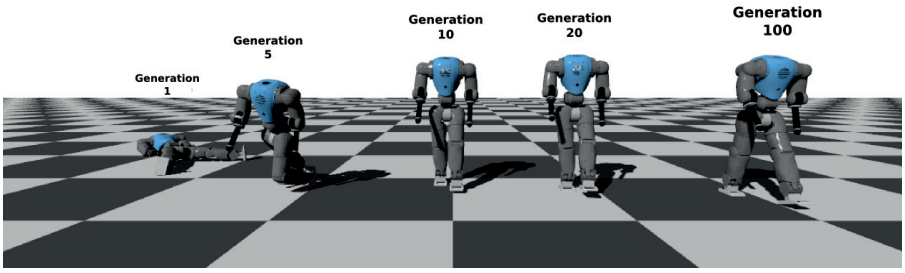
Because the search space is huge and the problem to solve is highly non-linear, these parameters tunings are performed using off-line heuristic optimizations. In contrast to deterministic optimization algorithms (usually requiring convex, continuous and derivable fitness functions), the heuristic algorithms can deal with complex fitness functions, like the ones developed in this thesis, but at the cost of a higher computational cost, and without guaranty to find the global optimum. More precisely, these heuristic optimizations usually recruit random processes, iteratively reproducing a given search scheme. Consequently, the optimizer can be trapped in a local optimum. Interestingly, these heuristic optimization algorithms are easy to parallelize, which is a key advantage given the time needed to run one optimization. Once optimized, the trained controllers can be run in real-time with conventional hardware, such as the PC104 embedded in COMAN (see Section 2.1).

All optimized results presented in this thesis were obtained using the particle swarm optimization (PSO) algorithm (Kennedy and Eberhart, 1995; Clerc and Kennedy, 2002). This heuristic algorithm is detailed in Appendix C. In Figure 2.2, an example of straight walking gait optimization in a 3D environment (see Chapter 9) is achieved using PSO.

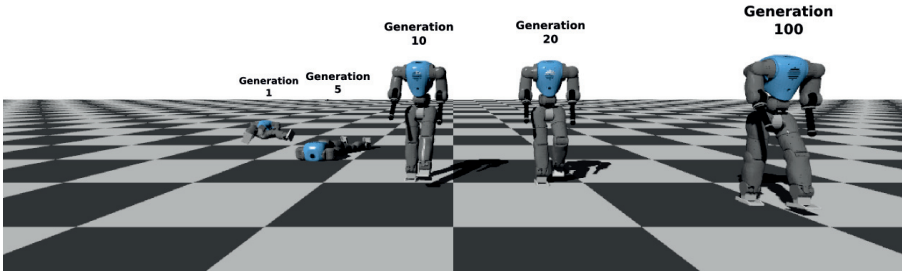
Note however that similar results could be obtained by using different heuristic algorithms, like a genetic algorithm (GA) (Weile and Michielssen, 1997) or an evolution strategy with covariance matrix adaptation (CMA-ES) (Hansen et al., 2003). In our case, PSO was selected



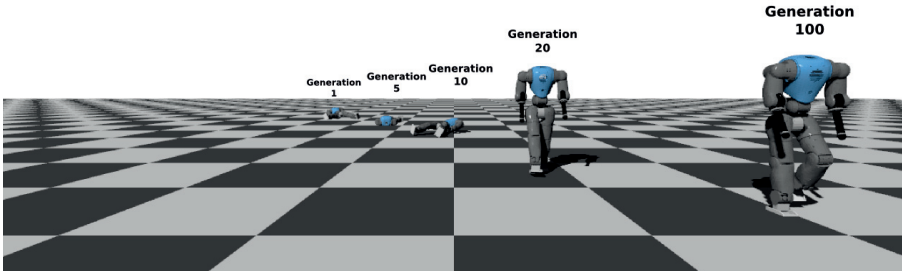
(a) $t = 0\,s$



(b) $t = 5\,s$



(c) $t = 8\,s$



(d) $t = 14\,s$

Figure 2.2: A particle swarm optimization (PSO) is run to tune the 3D walking gait of COMAN, in order to reach a target speed of $0.65\,m/s$. In this off-line optimization, 50 particles are recruited over 100 generations. In each panel, the performance of the best particle (i.e. set of optimized parameters) after 1, 5, 10, 20 and 100 generations is depicted. In panel (a), the simulation is initiated similarly for the different particles. The other panels present the resulting gaits evolution over time. The solution after 20 and 100 generations are both robust (i.e. no fall), but only the solution over 100 generations reaches the target speed of $0.65\,m/s$.

because it was already implemented on the computing cluster we used, and which is located in the BioRob laboratory [1].

2.4 General control framework

In this section, the general control structure being used in most of the thesis is presented. This structure is summarized in Figure 2.3.

The final goal of the controller is to provide appropriate voltages to the robot motors, in order to achieve locomotion. These voltages are either sent to the real robot motors or to their models implemented in simulation. Using this structure, the controllers developed in simulation can be directly transfered to the real hardware, without additional alteration, speeding up the testing process.

The controller is divided into three parts: (i) high-level controller, (ii) middle-level neuromuscular model, and (iii) low-level impedance controller. The low-level impedance controller receives torque and/or position references and produces appropriate voltages. Its implementation is summarized in Appendix D and is detailed in (Mosadeghzad et al., 2012).

The middle-level neuromuscular model lies at the heart of this thesis. More precisely, its purpose is to compute torque references reproducing the behavior of virtual muscles (and possibly some joint angle references). These virtual muscles are implemented as a set of equations called the Hill-type muscle model, as detailed in Chapter 3. In Chapters 4, 5 and 8, these muscles are only commanded by reflexes (i.e. feedback signals).

In Chapters 6, 7, 9 and 10, a central pattern generator (CPG) is added to provide descending (i.e. feed-forward) signals, in order to modulate the biped forward speed. This speed modulation is commanded with the high-level controller, a module introduced in these chapters and adapting a few key parameters as linear or quadratic functions of the speed reference v_{ref}. Finally, Chapter 10 extends the gait modulation to left/right steering, using a new scalar input: the heading reference h_{ref}.

[1] This PSO software is part of the Optimization Framework developed by Dr. Jesse van den Kieboom.

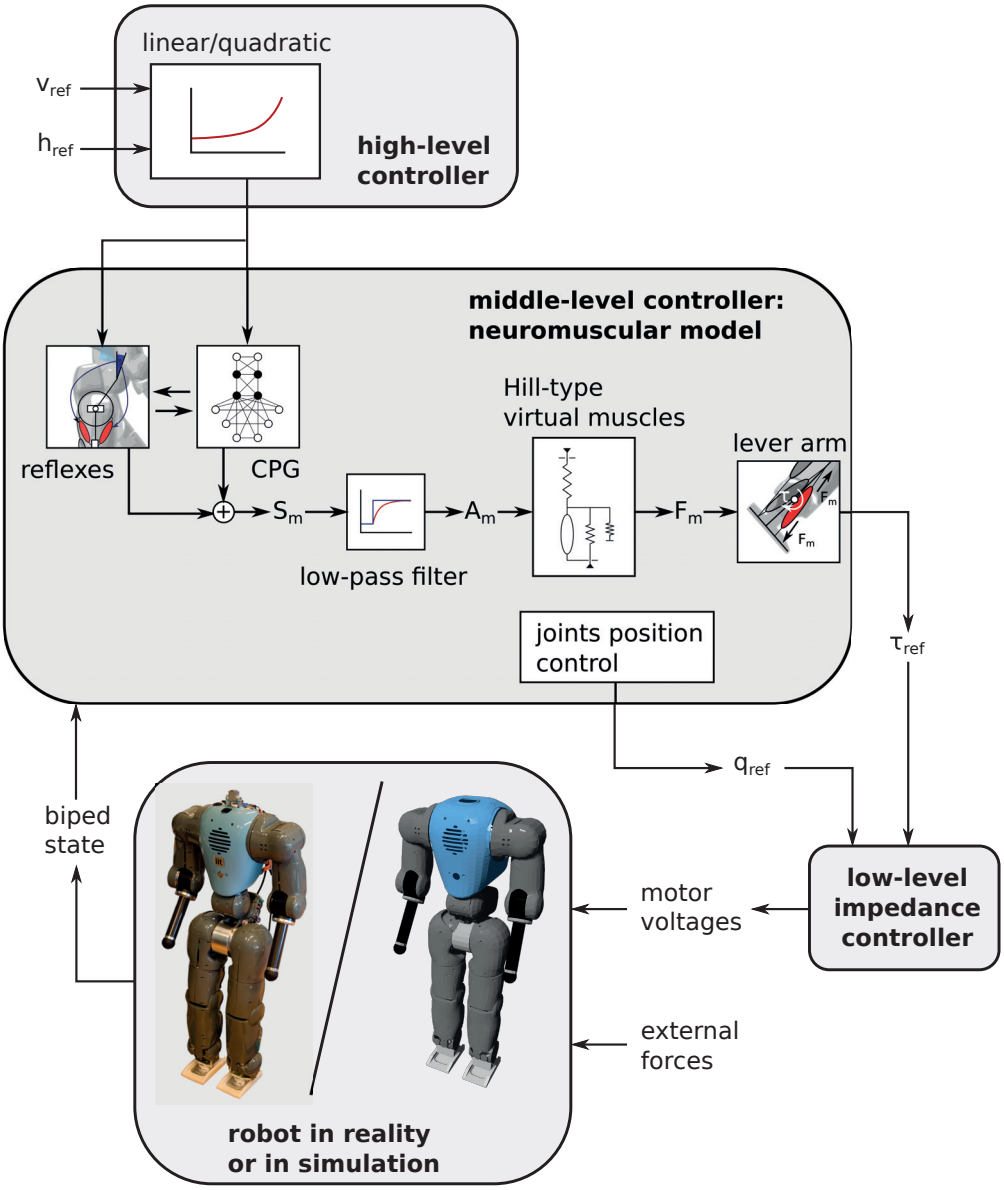


Figure 2.3: The controller final purpose is to compute appropriate voltages to drive the motors of COMAN, either in simulation or on the real hardware. The controller is divided into three parts: (i) the high-level controller provides commands to the middle-level controller by modulating key parameters as linear or quadratic functions of two scalar inputs (i.e. the speed reference v_{ref} and the heading reference h_{ref}), in order to adapt the gait accordingly; (ii) the middle-level controller computes torque references τ_{ref} and position references q_{ref}, while receiving sensory information from the biped state and key parameters modulated by the high-level controller; and (iii) the low-level impedance controller computes the motor voltages. Regarding the neuromuscular model, the interplay between the CPG and the reflexes provides stimulation signals S_m. They are later converted into activations A_m controlling the virtual Hill-type muscles. These muscles finally produce forces F_m, converted to the joint torques via lever arms. In parallel, some position references are computed.

3 Hill muscle model time integration

Publication

The material presented in this chapter is adapted from (abstract and poster):

Van der Noot N, Dzeladini F, Ijspeert AJ and Ronsse R (2014) Simplification of the Hill Muscle Model Computation for Real-Time Walking Controllers with Large Time Steps. In: Dynamic Walking, Zurich, 10-13 June 2014.

As discussed in Chapter 1, we are interested in adapting neuromuscular locomotion controllers to humanoid robots. This requires to deal with the robot controller limited resources and time constraints (dependent on the embedded computation units). In particular, we would like to port the reflex-based controller developed in (Geyer and Herr, 2010) to a real robotic device (see Chapter 4). This model recruits Hill muscle models, whose state integration requires to use small and/or adaptive time steps, which are not always compatible with the controller intrinsic frequency.

This chapter addresses this problem by studying the effects of the integration time step on the muscle state integration. Two solutions are evaluated: (i) replacing the full muscle integration scheme by steady-state approximations and (ii) iterating several times through the muscle dynamics equations during each controller step call.

3.1 Introduction

Bio-inspired controllers are emerging as a promising way to implement dynamic walking on humanoid robots without resorting to full local controllability concepts like the zero-moment point-based methods (Vukobratovic and Borovac, 2004). Among all the bio-inspired approaches, we implemented the one proposed by (Geyer and Herr, 2010), relying on reflex-controlled virtual Hill muscles (Hill, 1938).

Each Hill muscle state time integration is governed by stiff and strongly non-linear state equations. Consequently, this requires to use small (and/or adaptive) integration time steps, which

might lead to computational issues when transferring this model to real-time controllers.

In this contribution, we illustrate that the dynamics induced by this muscle-velocity relationship is actually negligible for fast muscles. It can thus be replaced by steady-state approximations. We compare three methods to compute these steady-state approximations, along with their impact on accuracy and computational cost. For slow muscles, we present a technique mixing both approaches: steady-state computations and full muscle dynamics-based model integrations. Finally, we also evaluate the possibility to iterate several times through the muscle dynamics equations.

The impact of the proposed solutions are evaluated in a forward simulation environment called Robotran (Docquier et al., 2013), modeling the torque-controlled robot COMAN (Dallali et al., 2013; Tsagarakis et al., 2013). This is tested during walking experiments using the neuromuscular controller of (Geyer and Herr, 2010), see Figure 3.2.

3.2 Hill muscle overview

The implementation of the Hill muscle model is fully described in (Geyer et al., 2003) and (Geyer and Herr, 2010). On top of this, the corresponding equations are developed in Section G.1. In short, each muscle tendon unit (MTU) consists of two main elements: the contractile one (CE) and the series elastic one (SE). Two additional elements only engage when the muscle is outside its normal range of operation: the parallel-elastic element (PE) and the buffer elasticity one (BE). Each virtual MTU attachment point to the body is known, such that its length l_{mtu} can be directly computed from the human body kinematics. Finally, the forces generated by these muscles can be determined from the length l_{ce} of the active, contractile element of each MTU. These different elements are depicted in Figure 3.1.

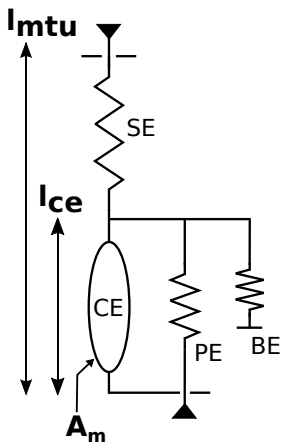


Figure 3.1: Hill muscle model.

The update rate of l_{ce} is governed by the muscle-velocity relationship, as stiff and strongly non-linear state equations. The corresponding dynamics can be computed from three inputs: the muscle lengths l_{ce} and l_{mtu} and the activation A_m, a neural input commanding the CE. The other quantities describing the muscles can be deduced from these three inputs. Because l_{mtu} can be computed from the body current kinematics and A_m is provided by the controller, only the length l_{ce} must be stored to describe the muscle state.

Updating l_{ce} requires to integrate its velocity v_{ce} (= $\dot{l}_{ce}$). However, v_{ce} is strongly impacted by the current value of l_{ce}. Because the value of v_{ce} is supposed to be constant during each integration time step, this can generate numerical issues when the related time step is too large.

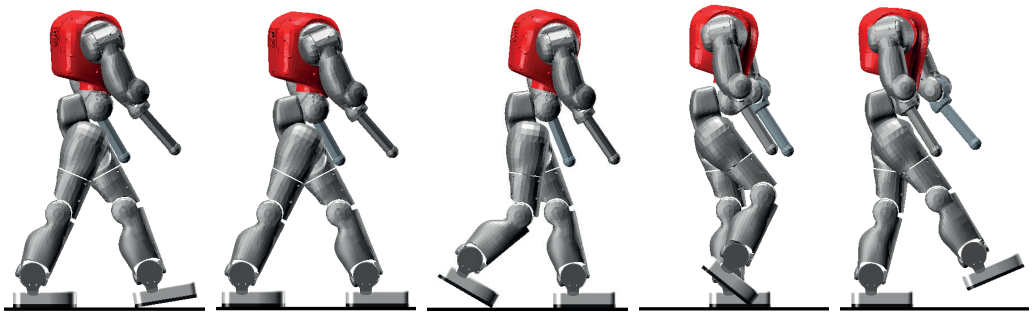


Figure 3.2: Walking gait of the COMAN robot in the Robotran simulator.

This problem is illustrated in Figure 3.3, where the length l_{ce} of three leg muscles are recorded during COMAN locomotion (see Figure 3.2). In this experiment, a Euler explicit integration scheme (known as being conditionally stable) was used. With a controller time step larger than 0.5 ms, numerical oscillations appeared. This is problematic due to the frequency of the real COMAN controller, locked to 1 kHz (i.e. controller called with a 1 ms time interval). Using more advanced integration schemes (like adaptive time steps) can solve the problem, but is usually too greedy or not applicable for real-time controllers, and is therefore not studied here.

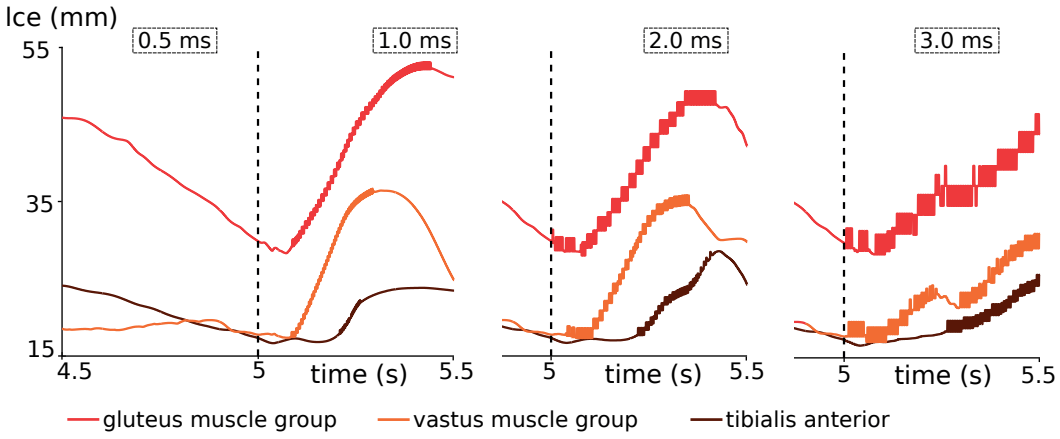


Figure 3.3: Temporal evolution of l_{ce} for three different muscles: gluteus muscle group (GLU), vastus muscle group (VAS) and tibialis anterior (TA). The muscle properties are similar to the ones of (Geyer and Herr, 2010), but they are scaled to the size of COMAN by using dynamic scaling methods, being described in (Bejan and Marden, 2006) and (Schepelmann et al., 2012). Initially, the integrator time step is set to 0.5 ms. At time $t = 5\,s$, it is changed to 1, 2 or 3 ms.

Importantly, the critical time step size strongly depends on the muscle properties. The reflexes of (Geyer and Herr, 2010) directly use l_{ce} in the feedback loop, such that oscillations in l_{ce} propagate and would likely make the robot fall.

3.3 Steady-state approximations

The muscle dynamics is mainly related to the computation of the speed v_{ce} of the CE part. Its computation depends on the three inputs l_{ce}, l_{mtu} and A_m, as presented in Equation (3.1), see Section G.1 for full developments.

$$v_{ce} = \dot{l}_{ce} = f(l_{ce},\ l_{mtu},\ A_m) \tag{3.1}$$

Here, we assume that the values of l_{mtu} and A_m are kept constant during each controller step call. This assumptions relies on the fact that the dynamics and rules governing l_{mtu} and A_m are much slower than the ones acting on l_{ce}. Iterating over Equation (3.1) with l_{mtu} and A_m kept constant results in the convergence of l_{ce} to its steady-state value l^*_{ce}, i.e. $f(l^*_{ce},\ l_{mtu},\ A_m) = 0$.

The problem illustrated in Figure 3.3 can then be explained as follows. Using a too large integration time step, l_{ce} is updated to a new value exceeding its steady-state value l^*_{ce}. Therefore, the sign of v_{ce} is reversed during the next time step, resulting in the oscillations of l_{ce} around its steady-state value l^*_{ce}.

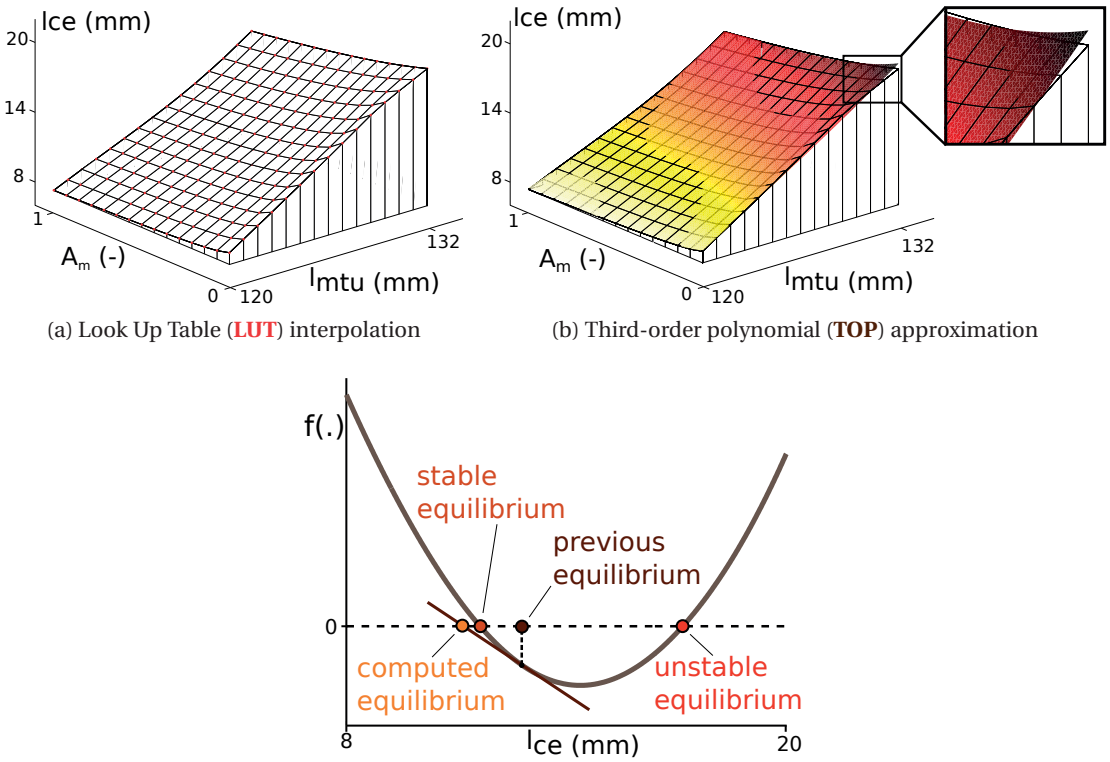
This indicates that the dynamics governing l_{ce} are so fast that it could potentially be neglected. In other words, l_{ce} could be replaced by its steady-state value l^*_{ce} (computed when l_{mtu} and A_m are supposed to be constant). We compare three methods to obtain this steady-state value in real-time (listed below and illustrated in Figure 3.4).

- A Look Up Table (LUT), generated off-line, stores the values of l^*_{ce} for many sets of inputs (i.e. l_{mtu} and A_m). Then, l^*_{ce} is interpolated for any new set of inputs in real-time. While being quite efficient, this method efficiency depends on the inputs mesh resolution. If this is too fine, this can generate problems due to the controller limited memory. In contrast, a too coarse refinement might deteriorate the computation accuracy. This method is illustrated in Figure 3.4a.

- From this LUT, a third-order polynomial (TOP) approximation is computed and later used in real-time to compute l^*_{ce}. While this method is the most computationally efficient, its accuracy strongly depends on the LUT to fit, and so on the muscle properties. This method is illustrated in Figure 3.4b. The zoom presented in this picture illustrates possible deviations with the LUT method.

- A Newton-Raphson scheme (NRS) is applied to solve (3.1) at steady-state (i.e. $f(\bullet) = 0$). Contrary to both previous methods, this one does not require a pre-process computation. Its main drawbacks are that it could converge to an unstable equilibrium point and that more than one iteration might be necessary to reach the desired accuracy. This method is illustrated in Figure 3.4c. Equilibriums are progressively updated to find the

roots of Equation (3.1). In the next section, results are reported when using only a single iteration for the Newton-Raphson scheme.



(a) Look Up Table (**LUT**) interpolation

(b) Third-order polynomial (**TOP**) approximation

(c) Newton-Raphson scheme (**NRS**)

Figure 3.4: The three methods evaluated to get l_{ce}^* without integration are presented. In panel (a), values of l_{ce}^* are evaluated for a grid of input sets (l_{mtu} and A_m). The value of l_{ce}^* for any new input set is then interpolated from the four closest sets from the grid. In panel (b), the grid of panel (a) is used to compute a third-order polynomial approximation, computing l_{ce}^* and depicted by the colored sheet. In panel (c), a Newton-Raphson scheme is applied on the function $f(\bullet)$ detailed in (3.1), in order to finds its stable root (i.e. stable equilibrium), corresponding to l_{ce}^*.

3.4 Steady-state approximation results

The three methods presented in Section 3.3 are compared to reference signals, during COMAN walking experiments (similar to the one depicted in Figure 3.2). These references are obtained using the full dynamics equations (i.e. time integrating over all the stiff and non-linear state equations) with a small time step (i.e. $0.5\,ms$, using Euler explicit). This is illustrated in Figure 3.5, when tested on two muscles with fast dynamics.

When comparing the three steady-state approximation methods, the third-order polynomial (TOP) one was the fastest to compute, but also appeared to be the less accurate. This is

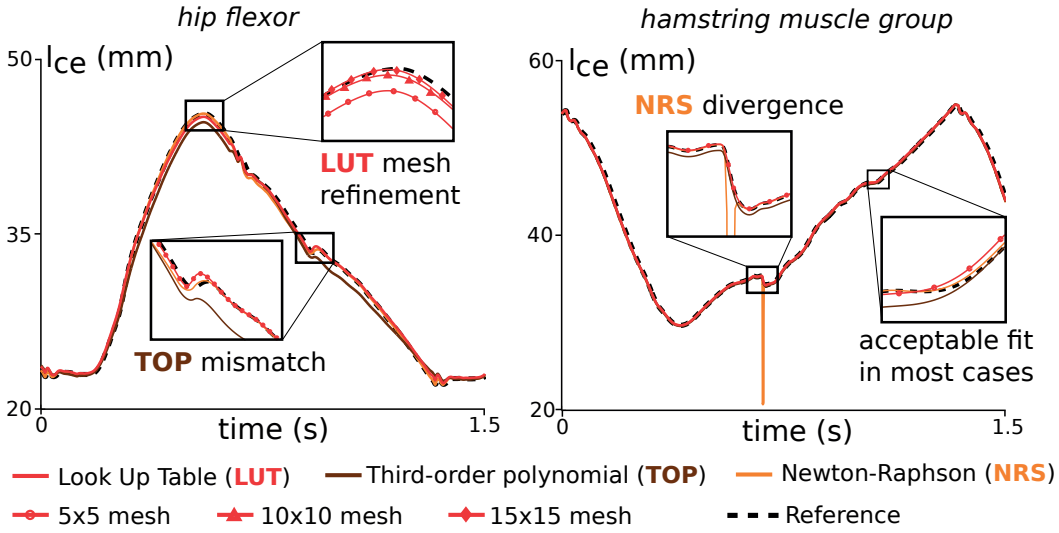


Figure 3.5: The three steady-state methods illustrated in Figure 3.4 are compared to reference signals obtained using the full dynamics equations with time integration. This is measured on the hip flexor (HFL, left panel) and the hamstring (HAM, right panel) muscle groups, during COMAN walking. For the LUT approximations, three different mesh sizes are recruited (for the $l_{mtu} \times A_m$ inputs): 5×5, 10×10 and 15×15.

particularly visible on the left panel of Figure 3.5, where an important mismatch with the reference signal is observed only for this method. This can be explained by the fact that the limited number of polynomial parameters cannot capture the whole evolution of l_{ce}^* with l_{mtu} and A_m. This problem can also be observed on the zoom performed in Figure 3.4b.

The Newton-Raphson scheme (NRS) appeared to perform globally well most of the time (by starting from the previous equilibrium and using a single iteration). However, some short divergences appeared, as illustrated on the right panel of Figure 3.5. Note however that this was only observed for the HFL muscle. Finally, the Look Up Table (LUT) method was the most reliable. Interestingly, correct fits were obtained for coarse meshes, like 10×10 (see left panel of Figure 3.5).

Globally, the curves fitting with the reference signals appeared to be quite accurate, in particular for the LUT method. These results suggested that neglecting the muscle-velocity relationship dynamics has indeed a very limited impact on the l_{ce} profile (for fast muscles). The three proposed methods efficiency depends on the muscle properties and on the controller requirements (intrinsic frequency and computational capabilities).

3.5 Combining full dynamics and steady-state approximations

The muscles studied in Section 3.4 had fast dynamics. Slow dynamics muscles suffer less from the numerical oscillations issue presented in Figure 3.3. Therefore, it is usually more appropriate to use the full muscle dynamics for these slow muscles.

However, we illustrate that combining the full muscle dynamics with the steady-state approximations developed in Section 3.3 is potentially relevant for these muscles with slow dynamics, when being used on controllers with low intrinsic frequencies. Here, we consider the case of the muscle with the slowest dynamics: the soleus muscle (SOL). This is studied for a controller whose intrinsic frequency was set to 100 Hz, i.e. 10 times slower than the one of the COMAN controller.

In Figure 3.6, the SOL muscle l_{ce} length is evaluated during the same COMAN walking experiment as the one depicted in Figure 3.2. The reference is computed using a 0.5 ms time-step (2000 Hz) while the other signals were computed with a 10 ms time step (100 Hz).

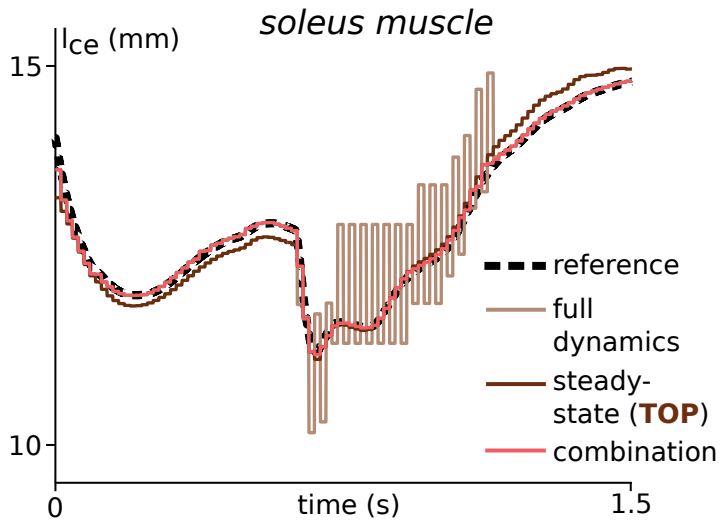


Figure 3.6: Temporal evolution of the SOL muscle l_{ce} length for the COMAN walking with the full dynamics model (i.e. with time integration). The *reference* signal is the one which was used in the controller (computed with full dynamics and 0.5 ms time step). All other signals were computed with a 10 ms time step. The *full dynamics* signal integrated $v_{ce} = \dot{l}_{ce}$, the *steady-state* one used the third-order polynomial (TOP) approximation and the *combination* signal used a mix of these two methods (*full dynamics* and TOP).

Using the full dynamics model integration (Geyer and Herr, 2010) resulted in the same problems as the ones presented in Figure 3.3, due to the large integration time step (10 ms). Using the third-order polynomial approximation (TOP) deviated from the actual reference for two reasons: (i) the fit with the steady-state value was not perfect and (ii) more significantly, the dynamics of this muscle was too slow to be considered as negligible with respect to the sampling frequency of 100 Hz (i.e. similar mismatches would occur with the LUT and NRS methods).

Interestingly, the full dynamics signal correctly fitted the reference when there was a mismatch between this reference and the TOP signal, and inversely. In fact, the TOP mismatch indicated that the muscle dynamics were actually non negligible (i.e. considering steady-state values of l_{ce} was not accurate). This corresponds to the smallest values of v_{ce}, and so to the moment when using the full muscle dynamics did not result in numerical oscillations.

Therefore, the following strategy was developed. Both signals (i.e. full muscle dynamics integration and steady-state approximation) were computed. The full muscle dynamics signal was used, except when the difference of l_{ce} and its steady-state signal l^*_{ce} changed its sign during one time iteration. In that case, the steady-state curve was selected. This resulted in the curve labeled *combination* in Figure 3.6. This curve remained close to the reference during the whole experiment, indicating that this combination strategy is useful when used for slow dynamics muscles implemented on controllers with slow intrinsic frequencies.

3.6 Performing several iterations

As illustrated in Figure 3.3, using the full muscle dynamics works fine, provided the integration time step is small enough. In case the controller intrinsic frequency is too small (i.e. controller time step superior to the critical muscle integration time step), another possibility is to iterate several times the muscle dynamics equations during a single step of the controller.

More precisely, the length l_{mtu} and the activation A_m are still considered to be constant during each controller step call. Two step calls are separated by a time interval Δt (1 ms for COMAN controller). During each controller step call, the muscle equations are iterated N times using an integration time step set to $\Delta t/N$, where N is selected to ensure that $\Delta t/N$ is smaller than the critical muscles integration time step (when using a Euler explicit integration scheme). At each iteration of the muscles dynamics, the new value of l_{ce} is used to compute the muscle quantities, among which v_{ce}. Therefore, the value of v_{ce} is progressively updated, which insures that the update of l_{ce} does not exceed its steady-state value l^*_{ce}.

3.7 Discussion

Interestingly, directly replacing the full muscle dynamics model integration with any of the three steady-state approximation methods (presented in Section 3.3) on the simulated COMAN preserved walking stability, although its gait became more jerky and less robust to perturbations, especially for the third-order polynomial approximation method. Re-optimizing the reflex rules led to retrieve more robust gaits.

On top of that, these walking gaits coped with time steps up to 3 ms, while the full dynamics model integration required a maximal time step of 0.5 ms. Even if the muscle simplification still holds above 3 ms (see Figure 3.6), higher time steps caused issues in the controller reflex rules refreshment. In sum, these steady-state approximations remain a viable solution to run

similar neuromuscular models on controller with small frequencies.

In our case, the COMAN on-board processing unit is composed of an embedded dual core Pentium PC104 unit running at 2.5GHz (Tsagarakis et al., 2013), whose controller call is performed with a time interval of $1\,ms$. This computational process is powerful enough to integrate the full muscle dynamics equations several times during a single controller call. Because the steady-state approximations still induce some small deviations compared to the initial Hill model, the iterative method described in Section 3.6 was selected to update the muscle states in all the remaining contributions of this thesis.

More precisely, during each controller step call, the muscle equations were then iterated 5 times with a Euler-explicit scheme using a $0.2\,ms$ time step. This time step of $0.2\,ms$ is smaller than the critical one ($0.5\,ms$, see Figure 3.3), in order to guarantee a safety margin. Importantly, the time needed to integrate these equations using this method (i.e. Euler explicit with a $0.2\,ms$ time step) is similar to the one needed when using a fourth order Runge-Kutta integration scheme with a $0.8\,ms$ time step. In similar situations, the integration is usually more stable with the fourth order Runge-Kutta. However, preliminary tests with Runge-Kutta did not result in improved stability. This unexpected result remains to be clarified.

4 Experimental validation of a reflex-based walking controller

Publication

The material presented in this chapter is adapted from:

> *Van der Noot N, Colasanto L, Barrea A, van den Kieboom J, Ronsse R and Ijspeert AJ (2015) Experimental validation of a bio-inspired controller for dynamic walking with a humanoid robot. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Sept. 28 2015-Oct. 2 2015, pp. 393-400. DOI: 10.1109/IROS.2015.7353403.*

This chapter aims at porting the neuromuscular reflex-based controller developed in (Geyer and Herr, 2010) to a real robotics platform: the COMAN. In that contribution, a 2D simulated model of an adult human was capable of reproducing human walking kinematics and dynamics, as well as demonstrating some robustness during blind walking when facing ground disturbances and slopes.

Porting this model from simulation to a real robot involves many challenges. One of them was addressed in Chapter 3: the muscle state integration requires a time step smaller than the one available on the controller. As presented in that last chapter, this was solved by iterating several times the muscles equations during each controller step call. Other challenges are addressed here, like the lack of lateral balance in (Geyer and Herr, 2010), due to the 2D simulation being used.

The purpose of this chapter is thus to assess if such neuromuscular controllers are viable alternatives (compared to traditional inverse kinematics/dynamics methods) to achieve stable locomotion with real humanoid robots. Indeed, while they feature valuable properties (human-likeness, energy efficiency, large steps, fast walking speeds...), these neuromuscular models were mainly limited to simulation studies so far.

4.1 Introduction

While opportunities for using mobile robots are steadily expanding, the necessity to adapt the environment to these robots slows down their integration in our everyday life. In a world designed for humans, using humanoid robots could be a solution, since they can cope with our natural environment (Schaal, 2007). However, they are still far from reaching the impressive human performances, e.g. regarding walking. This prevents them from being extensively used in our day-to-day life.

Different methods can be used to achieve dynamic walking with a robot. Likely, the most famous ones are based on the zero-moment point (ZMP) which can be used as an indicator of gait feasibility (Vukobratovic and Borovac, 2004). Many experimental validations were conducted to make humanoid robots walk with ZMP-based methods, for example with ASIMO (Chestnutt et al., 2005) or the HRP-2 platform (Kaneko et al., 2002).

However, there are important drawbacks associated with these control methods like energy inefficiency (Dallali, 2011). Moreover, these robots usually exhibit non human-like features like low waist position, constant knee flexion and foot surface kept parallel to the ground (Kurazume et al., 2005; Sardain and Bessonnet, 2004b). Even if some mechanical factors may explain these discrepancies, the main reasons are related to control strategies like singularity avoidance (Kurazume et al., 2005). In particular, ZMP-based controllers require full local controllability (i.e. local stability is ensured at every instant in time), which is not necessary for stable walking, therefore consuming more energy (Dallali, 2011).

The alternative concept called *limit cycle walking* adopts the perspective of relaxing constraints by considering the gait as a limit cycle and focusing on its global stability (Hobbelen and Wisse, 2007). Bio-inspired controllers are emerging as a promising way to implement limit cycle walking. For instance, (Shimoda et al., 2013) achieved robust reflex-based gaits on a humanoid robot for slow speeds (less than 1 cm/s). Another bio-inspired approach, developed in (Geyer and Herr, 2010), presents a simplified model of human locomotion being exclusively controlled by a chain of reflexes and muscles encoding principles of legged dynamics. This approach converges to a muscle-reflex model producing efficient and realistic walking gaits for the normal human range of speed. Moreover, the simulated viscoelastic properties of these muscles provide robustness to environment perturbations. It was for instance adapted to be the central layer of the control architecture of a powered ankle-foot prosthesis (Eilenberg et al., 2010).

In this chapter, we adopt and extend the approach of (Geyer and Herr, 2010). This bio-inspired reflex-based controller has already been thoroughly studied in simulation (Wang et al., 2012; Van der Noot et al., 2015b). In contrast, no study reports its implementation on a real full-body robot for human-like speed, to the best of our knowledge. The contributions of this chapter are (i) the implementation of this reflex-based bio-inspired controller to a real humanoid robot, namely the COMAN, focusing on the additional steps required to port it to real hardware; (ii) the achievement of a human-like robot walking gait in the sagittal plane, to highlight the

benefits of such a controller for bipedal walking; and (iii) a study of the discrepancies between simulation and reality, along with some clues to fix them.

Implementing this kind of controller on a real robot is not straightforward and new challenges appear with respect to scenarios limited to simulation environments: (i) working on real hardware requires to cope with the non-idealities of the real environment, such as highly non-linear joint friction torques or inaccurate torque tracking; (ii) reflex rules from (Geyer and Herr, 2010) only address the problem of 2D walking gaits, which is straightforward to get in simulation, but not on real hardware; and (iii) the experimental procedure is more likely to damage the robot and more difficult to automate. Our strategy is first to develop and optimize the controller in simulation under realistic assumptions, with the purpose to minimize the reality gap. Then, this controller is transferred to the real robot, without any additional tuning.

This chapter is organized as follows. In Section 4.2, we introduce the COMAN, the robot used for dynamic walking, and the simulation environment. In Section 4.3, we provide an overview of the controller, focusing on its implementation on real hardware. In Section 4.4, we show the results on a 50 steps walk performed by the robot, assessing its qualitative global behavior and comparing it to the simulation results. Finally, Section 4.5 concludes the chapter.

4.2 Hardware and software

An accurate simulation model of the COMAN is needed to develop the controller. We overview the robot hardware before presenting its simulation environment modeling.

4.2.1 COMAN platform

The COmpliant huMANoid (COMAN) is a 23 degrees of freedom (DOFs) full-body humanoid robot. This 95 cm tall robot, weighting 31 kg, was developed by the Italian Institute of Technology (IIT) (Dallali et al., 2013; Tsagarakis et al., 2013). COMAN is pictured in Figure 4.1, along with the inertial base, relative joint frames and the world planes used to describe its kinematics.

The three sagittal joints in each leg (see Figure 4.1) feature series elastic actuators (SEA) (Tsagarakis et al., 2009), while the three remaining leg joints are actuated using traditional, stiff actuators.

Regarding the robot sensors, each joint features position encoders, along with custom-made torque sensors. The torque tracking is then achieved with a PI controller (Mosadeghzad et al., 2012). On top of that, custom-made 6 axis force/torque sensors are placed below the ankle joint to capture the ground interaction forces and torques. Finally, an inertial measurement unit (IMU) is attached to the robot waist.

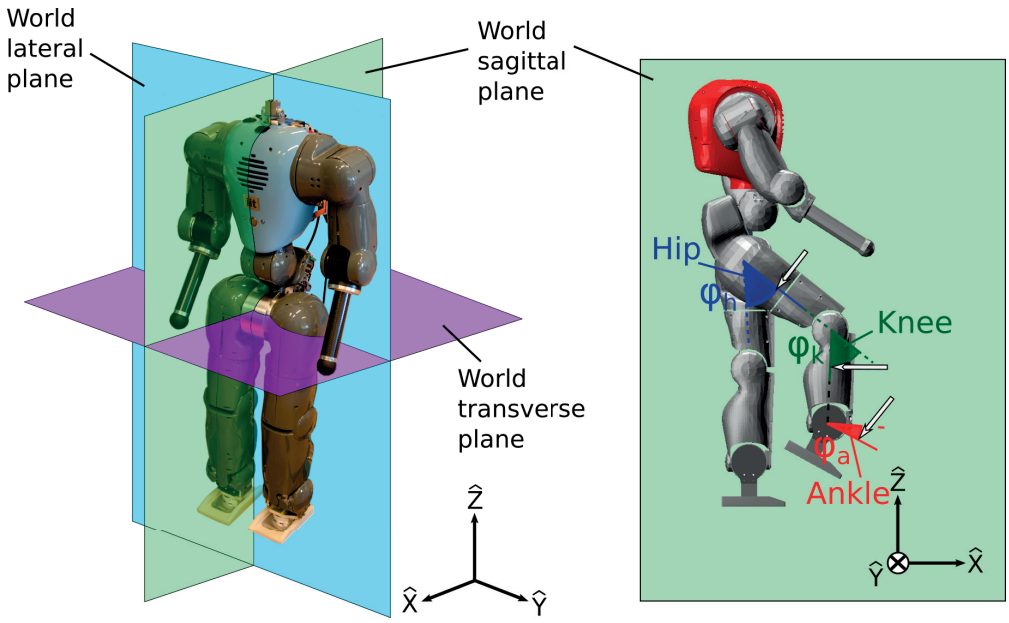


Figure 4.1: Real COMAN along with the world planes and an inertial frame (left panel). Simulated COMAN in the world sagittal plane with an inertial frame and the leg sagittal joints (right panel). On the right side, the arrows indicate the direction of increasing angles. In this case, the hip angle is negative while both other angles are positive.

4.2.2 Simulation environment

Section 4.1 identified several challenges related to the direct development of a bio-inspired controller on real hardware. Consequently, its development and optimization were first performed using a simulator modeling the COMAN in its environment. Next, the same controller can be transferred to the real robot.

The simulation suite, called Robotran (Samin and Fisette, 2003), is an environment for multi-body systems developed within the Université catholique de Louvain. Its direct dynamics module is used to generate the symbolic equations of the robot dynamics. COMAN in Robotran is displayed in the right panel of Figure 4.1. To further minimize the gap between simulation and reality, we reserved particular attention to the ground contact model (GCM), the actuator dynamics and the signals noise.

Regarding the GCM, we adopt nonlinear spring-damper models with realistic friction forces, as described in (Dallali et al., 2013). An accurate and realistic GCM is critical because the resulting forces are important inputs of the controller (see Section 4.3.2). In this chapter, we do not cover the problem of lateral dynamic stability, focusing on the so-called *2D walking* gait. Thus, in simulation, we constrain the waist to stay in the world sagittal plane (see Figure 4.1).

In contrast to the simplified seven-segments model used in (Geyer and Herr, 2010), our simulation purpose is to develop a controller able to run on the real robot. This involves

dealing with motor and sensor noise. The series elastic actuator dynamics significantly affects the robot dynamics and should therefore be carefully modeled. Their implementation in simulation is fully described in (Dallali et al., 2013). Receiving a control voltage signal, each motor generates a torque at the joint level in a similar way for both the real and the simulated COMAN.

Regarding sensors information, we only use inputs available on the real robot (see Section 4.2.1). Preliminary experiments revealed that the torque is the signal being the most affected by noise. Consequently, a white noise with a maximal amplitude of 0.4 Nm was added to the torque reading in simulation. This corresponds to the noise being observed with the actual robot.

4.3 Controller implementation

There are two main independent tasks in the controller: the lower-body and the upper-body ones, each of them sending position or torque references to a low-level controller. We briefly introduce this low-level controller before focusing on the lower-body and upper-body tasks. Then, we present the whole controller optimization.

4.3.1 Joints control

The joints controller is designed to track torque references or zero-position references (see Sections 4.3.2 and 4.3.3). This tracking is implemented on the real robot using a low-level controller described in (Mosadeghzad et al., 2012). The same low-level controller was replicated in simulation. Getting the appropriate voltage then reduces to compute appropriate position or torque references. The noise added in simulation on the torques reading directly impacts this part of the controller.

4.3.2 Lower-body control

Each leg is equipped with three sagittal joints (i.e. whose revolute axes are perpendicular to the sagittal plane) depicted in Figure 4.1 and three non-sagittal joints (two lateral and one transverse). Control of all leg non-sagittal joints consists in tracking zero-position references, so that they can barely move during walking. Indeed, these joints are useful to maintain lateral dynamic stability, a problem which is not directly addressed in this contribution.

The leg sagittal joints propel the body forward during the walking gait. These joints are mainly controlled by the biological approach described in (Geyer and Herr, 2010) and outlined here below. Seven muscle groups are identified within each leg (see Figure 4.2, right panel). Because the COMAN does not have any muscle, we consider here virtual muscles whose state is computed as a set of equations.

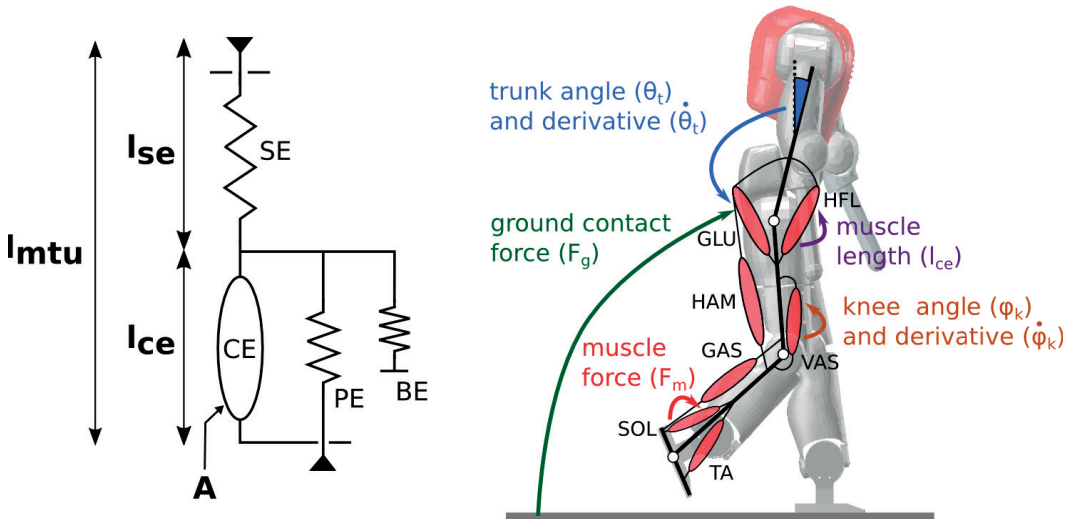


Figure 4.2: Hill-type muscle (left panel). COMAN with the seven muscle groups of the right leg and some examples of reflex rules inputs (right panel). Some of these inputs only affect the stance leg. Muscles: soleus (SOL), tibialis anterior (TA), gastrocnemius (GAS), vasti (VAS), biarticular hamstring (HAM), gluteus (GLU) and hip flexor (HFL).

More precisely, we use the Hill-type model (Hill, 1938), presented in the left panel of Figure 4.2. Each muscle tendon unit (MTU) consists of two main elements: a contractile one (CE) and a series elastic one (SE). On top of that, a parallel-elastic element (PE) and a buffer elasticity element (BE) only affect the muscle state outside its normal range of operation. Each virtual MTU attachment point to the real COMAN body is known, such that its length l_{mtu} can be directly computed from the sagittal joint angles φ (see Figure 4.1). The length l_{ce} of CE depends on an extra input: the muscle activation A, which is detailed later. The length l_{se} of SE is computed as $l_{se} = l_{mtu} - l_{ce}$ and is directly related to the force F_m produced by the muscle. Finally, this force F_m generates a torque contribution τ_m on one or two joints (for the bi-articular muscles HAM and GAS) as $\tau_m = r_m(\varphi)\, F_m$, where $r_m(\varphi)$ is the lever arm. Full muscles implementation details can be found in (Geyer and Herr, 2010) and (Geyer et al., 2003). To scale the muscles parameters to the size of the COMAN, we used dynamic scaling (Schepelmann et al., 2012).

For each joint, the sum of all muscle torque contributions τ_m is sent as torque reference to the corresponding low-level controller presented in Section 4.3.1. The actual robot internal controller has a sampling period limited to 1 millisecond, which is too slow to cope with the muscle state integration, being governed by a stiff and strongly non-linear state equation (Van der Noot et al., 2014). Consequently, we integrate the muscle model several times during each controller time step (see Chapter 3). Interestingly, this controller computation remains very fast (see Section 4.4.4), despite these additional iterations.

The muscle activations A_m are related to neural inputs S_m called muscle stimulations, using a first-order low-pass filter capturing the excitation-contraction coupling, presented in (4.1) where τ is a time constant.

$$\tau \frac{d A_m}{d t} = S_m - A_m \tag{4.1}$$

Some examples highlighting how key stimulation contributions S_i are computed are described in (4.2), (4.3) and (4.4), focusing on their relations with the robot inputs. They are also visible in the right panel of Figure 4.2. Summing these contributions S_i on each muscle produces their corresponding stimulation S_m. All parameters with index *opt* are optimized parameters (see Section 4.3.4). The whole description of stimulation computation is provided in (Geyer and Herr, 2010).

$$S_i = k_{F,opt} F_m \quad ; \quad S_i = k_{l,opt} (l_{ce} - l_{opt}) \tag{4.2}$$

$$S_i = k_{g,opt} F_g \left(k_{\theta,opt} (\theta_t - \theta_{opt}) + k_{\dot{\theta},opt} \dot{\theta}_t \right) \tag{4.3}$$

$$S_i = k_{\varphi,opt} (\varphi_k - \varphi_{k,opt}) [\varphi_k < \varphi_{k,opt}] [\dot{\varphi}_k < 0] \tag{4.4}$$

The stimulations (4.2) capture reflex rules simply governed by the muscle states F_m and l_{ce}. The contribution of (4.3) stabilizes the trunk like an inverted pendulum, using the trunk angle θ_t and its derivative $\dot{\theta}_t$. On the COMAN, computing these two inputs requires to integrate signals provided by the IMU attached to the waist, while adding the trunk angles contribution using forward kinematics. The ground reaction forces F_g are available through the force sensors placed below the ankle joints. They are also used to trigger swing and stance phases. Finally, (4.4) inhibits the VAS muscles to prevent knee hyperextension (i.e. when $\varphi_k < \varphi_{k,opt}$ and $\dot{\varphi}_k < 0$). Its inputs are the knee angle φ_k and derivative $\dot{\varphi}_k$. Its activation is triggered only when the conditions displayed in the brackets are satisfied.

Preliminary tests revealed the prominent impact of non-linear joint friction torques in the real robot. Therefore, the computed muscle stimulations were typically too low to counteract these frictions (especially for the knee and ankle joints during swing motion). Modelling the joint friction is quite challenging and is not addressed in this contribution. However, we present one clue fixing the ankle joint issue.

Friction reduces ankle flexion during swing, hence deteriorating foot clearance with respect to the ground. This can result in an early touch of the foot on the ground. Consequently, the reflex rules from (Geyer and Herr, 2010) were extended with an extra stimulation S^+_{TA} feeding the TA muscle (see Figure 4.2) during the swing phase: $S^+_{TA} = k_{\varphi,a} (\varphi_a - \varphi_{a,th})$ where φ_a is the ankle position (see Figure 4.1) while $k_{\varphi,a}$ (set to 4) and $\varphi_{a,th}$ (set to -0.1 rad) are two parameters

manually tuned. This extra stimulation was added after the optimization process in order to prevent their minimization since they impose an unnecessary cost to the frictionless joint model. This affected the simulation gait, reducing its speed.

The reflex rules governing the knee dynamics were kept similar to those of (Geyer and Herr, 2010), despite unmodeled friction. The impact of this will be discussed in Section 4.4.3.

4.3.3 Upper-body control

The upper-body is made of eleven joints: four joints for each arm (shoulder roll, pitch, yaw and elbow) and three joints for the trunk. In general, the upper-body control is used to provide lateral balance to the walker. In simulation, this is not needed because of the 2D walking constraint (see Section 4.2.2). Consequently, we track constant position references for all these joints. Hence, the pose of the arms does not change.

On the real robot, another strategy was implemented to provide lateral balance, involving the upper-limbs. The main purpose is to let a human operator grab the wrists of the robot to provide lateral balance with a limited effect (ideally null) on the sagittal plane motion. This behavior is achieved using Cartesian space impedance controllers on both arms of the robot (Hogan, 1985) to keep them loosely only in the sagittal plane and stiff in the lateral direction. At the same time, the trunk joints are fixed using zero-position tracking. The same controller is implemented for each arm. The yaw shoulder joint is fixed to a constant position and the control torques of the remaining three joints can be expressed as follows:

$$\tau_{arm} = J^T(q_{arm}) K_C (x_{des} - x) - D_J \dot{q}_{arm} \tag{4.5}$$

where q_{arm}, $\dot{q}_{arm}$, $\tau_{arm} \in \mathbb{R}^3$ are respectively the joint position, velocity and torque vectors and x_{des}, $x \in \mathbb{R}^3$ are respectively the Cartesian desired and actual positions of the wrist (with respect to a base reference frame with the origin at the pelvis of the robot and oriented as the inertial base in Figure 4.1). $J(q_{arm}) \in \mathbb{R}^{3x3}$ is the Jacobian of the wrist position in the same base frame and $K_C = diag(k_x, k_y, k_z) \in \mathbb{R}^{3x3}$ is the Cartesian stiffness matrix. It collects the desired stiffness along the x, y and z directions of the base frame. $D_J = diag(d_{q_1}, d_{q_2}, d_{q_3}) \in \mathbb{R}^{3x3}$ is the joint damping matrix collecting the damping for the three joints of the arm. The stiffness matrix and the damping matrix are positive-definite.

The first term of (4.5) maps into the arm joint space a force being proportional to the wrist position error according to the selected stiffness. The second term adds damping directly at the joint level to limit the joint velocities and stabilize the whole system. The desired behavior is achieved using high stiffness values along the lateral direction ($k_y = 500\,N/m$) and zero stiffness along the sagittal directions ($k_x = k_z = 0\,N/m$). Finally, a low value is used for the damping of all the joints ($d_* = 0.1\,(Nms)/rad$). In this way, the robot wrists can freely move in planes parallel to the sagittal one.

This impedance-based strategy turned to be a more viable solution for a treadmill-based experimental setup in regards to other existing solutions, like e.g. using a boom constraining the robot on a circular path (Geng et al., 2005).

4.3.4 Controller optimization

The lower-body controller design includes many open parameters (see Section 4.3.2 and (Geyer and Herr, 2010)), which must be properly tuned in order to first generate a stable walking gait, and then to optimize the gait efficiency. This tuning is performed in simulation with an extensive off-line optimization process using a heuristic optimization algorithm called *particle swarm optimization* (PSO) (Kennedy and Eberhart, 1995). The optimization process simulates a maximum 60 s walking gait. Sufficient foot clearance with the ground is guaranteed by adding bumps in the simulation environment during the optimization process. These bumps are trapezoidal shapes placed under the swing foot, next to the stance one. Their height linearly increases from 0 cm to 2 cm. This ensures that the swing foot lower extremity is at least 2 cm above the ground when both feet are next to each other (the most critical moment for ground clearance).

Each set of parameters is tested according to a staged objective function, i.e. different objectives are sorted by order of relevance so that the next objective is taken into account only when the previous one is fulfilled. At first, the robot must walk without falling, the objective function being proportional to the walking time. When it is able to walk during the 60 s simulation run, the forward speed is driven towards an arbitrary target speed of 0.5 m/s, which is a reasonable speed according to the robot height. The objective function f is computed as follows:

$$f = \alpha\, e^{-\beta (x - x_{des})^2} \tag{4.6}$$

where x is the forward speed, x_{des} the target speed and α, β two weight parameters. This function provides a result bounded between 0 and α. When the robot speed lies within a range of 0.05 m/s around the target speed, we minimize the metabolic energy consumption in the virtual muscles contraction per unit distance walked (Bhargava et al., 2004), again using (4.6). However, x now represents the metabolic energy consumption per unit distance walked while x_{des} is equal to zero, in order to minimize the absolute energy expenditure.

The noise added in the simulation environment (see Section 4.2.2) helps the optimizer to converge to robust controllers. This is essential for experiments with real robots.

4.4 Results

We transferred the controller presented in Section 4.3 directly to the real COMAN, without any additional tuning of the optimized parameters. Running it on real hardware led to a successful walking experiment where the robot was making 50 steps. We first present the experimental setup and assess the lateral balance controller before comparing the experimental gait with the simulation results. Finally, we detail some specific features of this gait controller implemented on the real hardware.

4.4.1 Experimental setup

The main challenge of the experiments conducted with the real robot was to maintain its waist in the world sagittal plane, in order to reproduce the 2D walking gait generated in simulation. The adopted solution consists in using the impedance controller described in Section 4.3.3 so that a human operator provides the lateral balance with a limited effect on the sagittal motion. The operator grabs the wrists which can freely move in planes parallel to the sagittal one (see Figure 4.3).

The robot was initially suspended above a treadmill, from a hook located in the robot neck and connected to a pulley. Then, the robot was moved down such that an initial contact between the robot feet and the running treadmill initiated the reflex chain, and so the walking gait. In steady-state, the suspension rope did not interfere with the robot motion. The whole experimental setup is illustrated in Figure 4.3.

In simulation, the robot speed was around 0.4 m/s, due to the extra TA stimulation added after the optimization (see Section 4.3.2). To match the real robot speed, the treadmill speed was set to 0.2 m/s, so two times smaller than the one obtained in simulation. This requirement for a lower experimental speed is explained in Section 4.4.3.

4.4.2 Lateral balance

During the experiment, a first trial resulted in a fall of the robot in the sagittal plane after a few steps, due to a contact between the swing leg and the ground. This indicates that the lateral support provided by the operator is only ensuring stability in the lateral plane (as expected) and that the robot is alone in charge of its stability in the sagittal plane.

The effects of the lateral balance support on the sagittal plane can further be quantified by reporting the forces generated by the upper-body impedance controller on the arms during walking. Figure 4.4 shows these forces for the left arm. Opposite forces with the same magnitude are generated at the waist of the robot, which is coherent with the action-reaction chain. The lateral force $\hat{Y}$ (in blue) is two orders of magnitude larger than the frontal $\hat{X}$ and vertical $\hat{Z}$ forces (respectively in green and red). In fact, due to the stiffness values reported in Section 4.3.3, the movement of the hands, operated by the human, induced significant

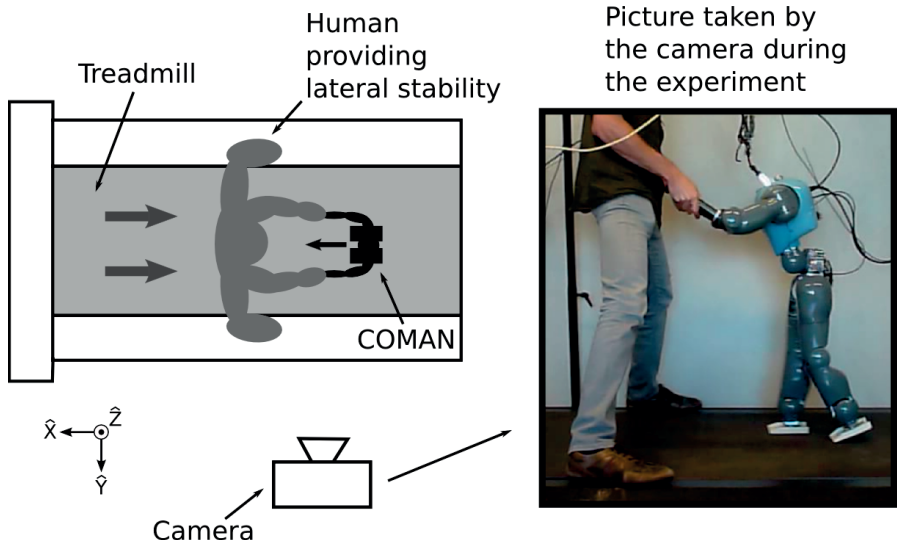


Figure 4.3: Experimental setup for testing the COMAN dynamic walking. The robot was walking on a treadmill while a human operator placed in front of it provided lateral stability. A camera was used to capture the sagittal plane motion presented in Figure 4.6.

forces along the $\hat{Y}$ direction only (see Figure 4.3). The forces along the $\hat{X}$ and $\hat{Z}$ axes were only generated by the joints damping. This result thus validates the fact that the human assistance barely impacted the robot motion in the sagittal plane and so that lateral support did not affect the 2D walking. The robot was thus free to move and to fall in the sagittal plane, as it actually happened from time to time.

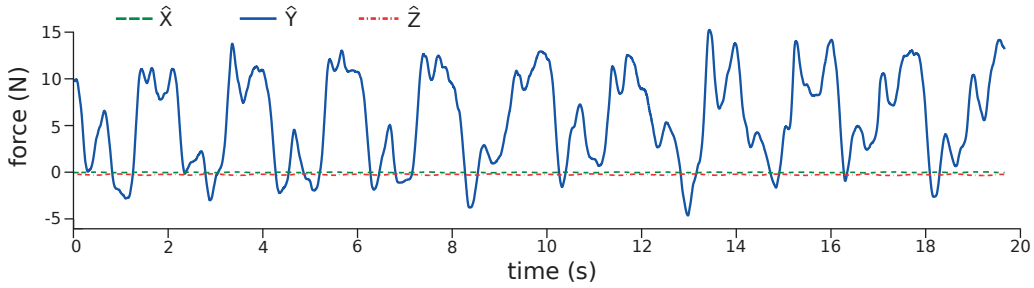
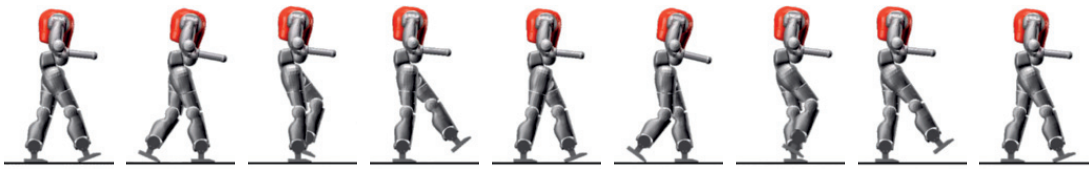


Figure 4.4: The force amplitude along the lateral $\hat{Y}$ direction is much bigger than the forces along the frontal $\hat{X}$ and vertical $\hat{Z}$ directions (measured on the left arm during ten strides).

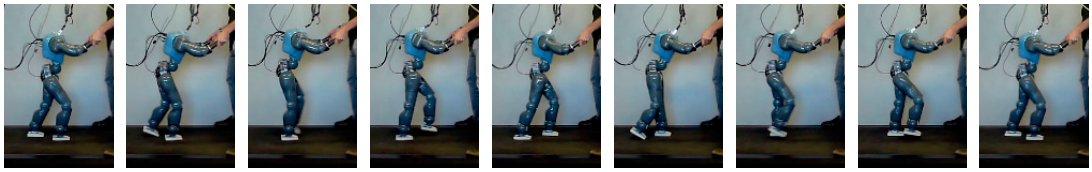
4.4.3 Comparison between simulation and experimental results

Figure 4.5 shows nine snapshots of the simulated COMAN with the selected optimal settings and the extra ankle stimulations presented in Section 4.3.2. These snapshots span a time frame of 1.46 s, corresponding to one stride starting at left foot strike. The same controller transferred to the real COMAN led to the gait shown in the snapshots of Figure 4.6. These snapshots also present one stride starting at left foot strike, although spanning a longer time frame of 2.03 s. The snapshots of these two figures are mirrored to improve legibility (to have a motion from left to right).



(a) t=0 s (b) t=0.2 s (c) t=0.4 s (d) t=0.57 s (e) t=0.73 s (f) t=0.97 s (g) t=1.13 s (h) t=1.33 s (i) t=1.46 s

Figure 4.5: Snapshots of the COMAN in the Robotran simulation environment, corresponding to panels (a) and (c) in Figure 4.7 and (a) in Figure 4.8. Snapshots (a), (e) and (i) are taken at foot strike, (b) and (f) at foot push-off, (c) and (g) when feet are adjacent and (d) and (h) during late swing.



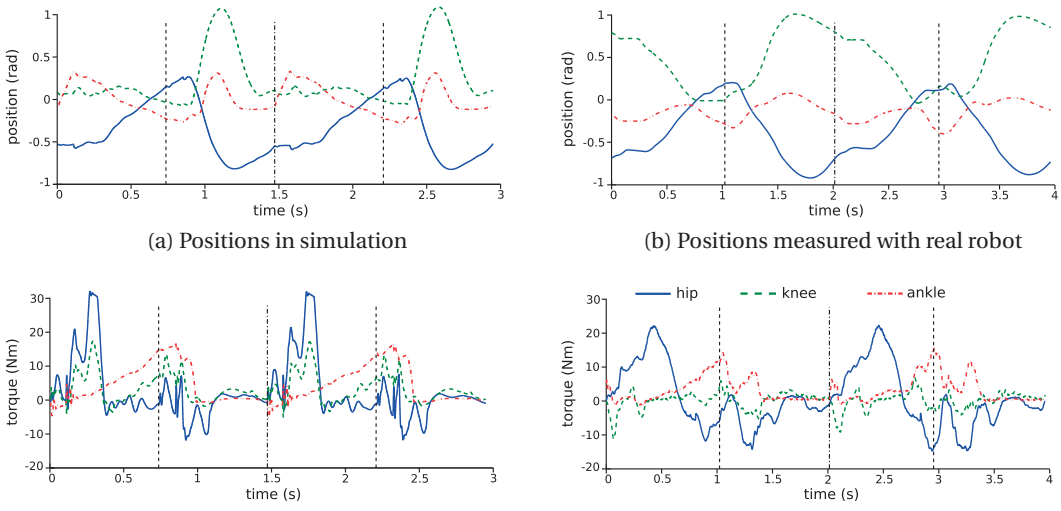
(a) t=0 s (b) t=0.37 s (c) t=0.57 s (d) t=0.8 s (e) t=1.03 s (f) t=1.37 s (g) t=1.6 s (h) t=1.8 s (i) t=2.03 s

Figure 4.6: Snapshots of the real COMAN, corresponding to Figure 4.4, to panels (b) and (d) in Figure 4.7, to panel (b) in Figure 4.8 and to Figure 4.9. These snapshots are consistent with the ones detailed in Figure 4.5 (e.g. panel (a) is also taken at foot strike).

The major difference between the simulated and the real gaits is the forward speed: 0.4 m/s in simulation against 0.2 m/s on the real robot (treadmill speed). Indeed, the real COMAN exhibits shorter steps with a lower frequency, as can be observed when comparing Figs. 4.5 and 4.6. The behavior of the stance and swing legs during swing initiation is qualitatively similar in these two figures. However, this is not the case for the swing leg in late swing phase (compare snapshots (d) and (h) in these two figures). Indeed, while the knee is stretched in simulation, this is not the case with the real COMAN. This is likely due to the joint friction torques that were not modeled in the simulation environment. Consequently, the leg is never fully stretched during the swing phase, resulting in smaller steps. On top of that, a flexed swing leg is shorter than a stretched one. Consequently, impact with the ground happens later and induces a lower gait frequency on the real robot. So, these shorter steps combined with their slower frequency decrease the robot forward speed from 0.4 m/s to 0.2 m/s. The joint friction torques (and possibly other effects like no perfect motor back-EMF compensation)

have thus a large impact on the resulting gait, also preventing heel strikes to appear. However, despite this difference in forward speed, the real robot was able to perform 50 steps, without any additional controller tuning. This demonstrates an impressive level of robustness of this bio-inspired controller, despite the external perturbations and unmodeled dynamics.

Figure 4.7 and Figure 4.8 show the actual positions, torques, and ground reaction forces captured during two strides of these walking trials, for both the simulated and the real COMAN. These graphs follow the conventions depicted in the right panel of Figure 4.1. Despite the significant difference in the cycle duration, the gait kinematics is quite similar between the simulated and the real COMAN (compare panels (a) and (b) in Figure 4.7). In particular, the hip trajectories are barely distinguishable. In both cases, the knee trajectory peaks during swing initiation but lasts longer on the real robot for the reasons phrased above. During a fraction of the stance phase, the knee position lies near zero for the real COMAN, indicating that the leg is stretched. This feature is usually not encountered in most humanoid robots to avoid controller singularities. Regarding the ankle, the pattern is more different. Indeed, this joint is more affected by the ground interactions. In contrast to many traditional ZMP-based walker, the robot feet are not always kept parallel to the ground. This is especially visible during swing initiation in snapshots (b) and (f) of Figure 4.6. On top of that, it is interesting to note that if the leg was stretched during the swing phase, the COMAN would hit the ground with the heel at foot strike, like humans do. Regarding torques, a proximodistal gradient also appears when analyzing simulation and reality matching: the hip matching is slightly better than the ankle one.



(a) Positions in simulation

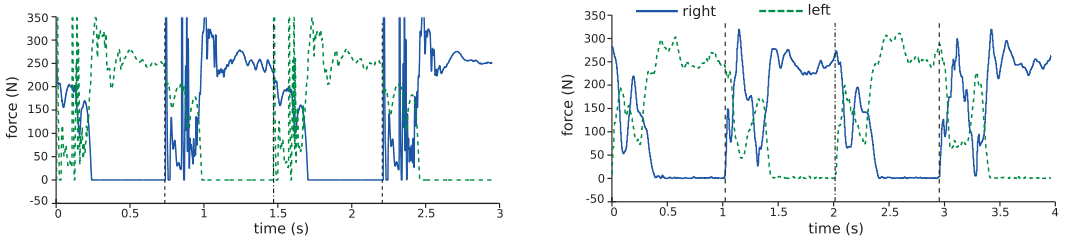
(b) Positions measured with real robot

(c) Torques in simulation

(d) Torques measured with real robot

Figure 4.7: Positions and torques on the COMAN for the left leg sagittal joints, both in simulation (left) and on the real robot (right), according to the conventions depicted in Figure 4.1. The joint frames are depicted in Figure 4.1. These graphs start at a left leg foot strike and span over two strides. Right strikes are indicated with dashed lines while the second left strike is indicated with a dashed line integrating dots. The time references are consistent with the ones presented in Figure 4.5 (simulated robot) and Figure 4.6 (real robot).

The global pattern of the ground reaction forces is again similar (see Figure 4.8), except at foot strike where sharp variations happen in simulation, but not on real hardware. The reason is that the simulated ground contact model involves high and noisy peaks at foot strike, due to the stiff spring-damper contact model used (Dallali et al., 2013).



(a) Vertical feet forces in simulation

(b) Vertical feet forces measured with real robot

Figure 4.8: Vertical feet forces, both in simulation (left) and on the real robot (right). The time references are consistent with the ones in Figure 4.7.

4.4.4 Bio-inspired controller features

Some of the specific gait features emerging from this reflex-based bio-inspired controller were already identified in Section 4.4.3, like foot roll and stretched knees during stance phase. Additionally, conventional ZMP-based walkers tend to walk conservatively by lowering the waist height (Kurazume et al., 2005). This is not the case with our controller (see Figure 4.6), due to the stretched stance leg. These human-like features also enable more energy-efficient walkers by taking more advantage from the inertia effects, but this remains to be quantified. Finally, the real COMAN gait showed a remarkable reproducibility over successive strides. This can be observed in Figure 4.9 where positions and torques are displayed over ten strides.

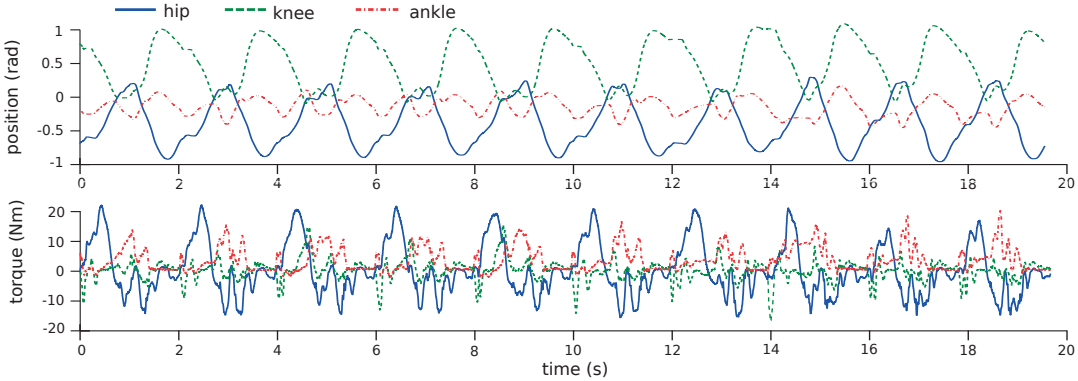


Figure 4.9: Positions and torques on the real COMAN left leg sagittal joints, according to the conventions depicted in Figure 4.1. These graphs start at a left leg foot strike and span over ten strides. The time references are consistent with the one in Figure 4.6.

Classical controllers, relying on inverse dynamics computation, require heavy computational process (Dallali, 2011). Consequently, fulfilling real time constraints is challenging. In this experiment, running one iteration of the whole lower-body bio-inspired controller was performed on average in 14.8 μs. This was tested on a computer with dual-core Intel(R) Core(TM) i7-4600U CPU, 2.1 GHz and 8 Go RAM. This is more than 67 times faster than the controller sampling rate, namely 1 ms, which is another key advantage of this bio-inspired approach.

4.5 Conclusion

In this contribution, we presented an experimental gait on a humanoid robot walking with a bio-inspired controller, based on reflex rules. While this controller was already extensively studied in simulations, we brought it to a real full-body humanoid robot: the COMAN. We presented some extra steps to port it on real hardware, like actuator dynamics modeling, robustness to noise or real-time issues. Running experiments on the real robot highlighted the non-idealities of the real world, stressing the necessity to drive the controller design according to them. The major non-idealities were related to joint friction, especially at the knee and ankle levels. We focused on the impacts of friction on the gait pattern and we presented some clues to fix them.

Regarding the walking gait, we pointed out interesting controller features and compared them to classical approaches: fast computational rate, and similarities to key human gait features (stretched leg during stance phase, foot roll and higher waist position), which could lead to more energy-efficient robots. Moreover, this controller demonstrated some robustness when transferred from a frictionless joint model simulator to real hardware, without any controller re-tuning (robustness to external perturbations demonstrated in (Geyer and Herr, 2010) still needs to be tested on the real robot). This was illustrated on a 50 steps trial where lateral stability was provided by a human operator.

The reported results call for further developments. A first improvement would be to reduce the gap between simulation and reality by implementing joint friction effects in the simulation environment. Nonetheless, friction modeling is not trivial and would not solve the knee flexion issue during swing phase, due to the lack of corresponding stimulation control. A precise timing is required to stretch the leg, which is difficult to get on a pure reflex-based controller. Therefore, we are also exploring the addition of new muscles control principles, like the introduction of a central pattern generator (CPG) to predict the current gait cycle phase (Van der Noot et al., 2015b). A CPG is a neural circuit found in both invertebrate and vertebrate animals, capable of producing rhythmic patterns of neural activity, while receiving only tonic inputs (Ijspeert, 2008).

Finally, the controller implemented only 2D walking gaits. We presented a new method to test these gaits on real robots without deploying complex boom structures like in (Geng et al., 2005). We assessed the effectiveness of this approach by reporting almost zero interaction forces between the human operator and the robot in the sagittal plane, and by showing that human

interventions did not prevent the robot from falling in this sagittal plane. Future developments should however focus on full 3D control (like the simulations results from (Wang et al., 2012)), rather than developing new experimental setups for 2D walking (see Chapters 9 and 10).

Interestingly, even if the resulting gait appeared to be quite different from the the one optimized in simulation, the robot still managed to walk, demonstrating some robustness inherent to these neuromuscular controllers. This motivates the fact that these bio-inspired controllers could possibly pave the way for new generations of humanoid robots. With proper re-tuning and/or with the aforementioned improvement suggestions, it might even be possible to obtain similar gaits between simulation and real hardware.

The next developments of this thesis are performed in simulation, mainly in order to improve the biped robustness and the steering capabilities. Indeed, the final goal of this thesis is to obtain rich locomotions being controlled by an external operator. In contrast, the neuromuscular model implemented here was optimized to reach a single gait, so that its speed and heading could not be adapted during the experiment.

Importantly, all developments performed in the next chapters are done with the purpose to target possible transfers to real robotic devices. Similarly to what was done in this chapter, this involves developing controllers capable of running while fulfilling the real-time constraints, introducing noise in the simulation environment and only using inputs available on the real robot.

While bipedal robots are currently far from reaching the walking capabilities of real humans in terms of robustness and energy-efficiency, this chapter shows that it is possible to take advantage of motor control mechanisms identified in humans to reproduce them on robotic devices, and so to get bipedal robot behaviors closer to human ones.

5 Feet with human-like compliance

Publication

The material presented in this chapter is adapted from:

> *Colasanto L, Van der Noot N and Ijspeert AJ (2015) Bio-inspired walking for humanoid robots using feet with human-like compliance and neuromuscular control. In: 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids), Seoul, 3-5 Nov. 2015, pp. 26-32. DOI: 10.1109/HUMANOIDS.2015.7363518.*

My contributions in this publication include the full design of the simulation environment (together with the implementation of the simulated flexible feet models), the processing of the experiments and the results extractions. I also wrote the related parts in the paper.

In Chapter 4, we ported the neuromuscular reflex-based model of (Geyer and Herr, 2010) to the real COMAN platform, and managed to obtain gaits displaying human-like features (stretched stance leg, foot roll and high waist position). However, the rigid feet used in the related experiments were highly different from the human ones.

In this chapter, we test how changes in the embodiment can affect the walking gait. More precisely, we study the effects of replacing the COMAN rigid feet with flexible prosthetic feet designed for children. This is studied in simulation while controlling the locomotion with the model of (Geyer and Herr, 2010).

Real prosthetic feet are first characterized, in order to implement a faithful model of these flexible prostheses in the simulation environment. The gaits obtained while using the flexible feet model are then compared to the gaits obtained with the rigid one. Interestingly, this added compliance can be handled by the bio-inspired controllers studied in this thesis, without changing the controller rules (i.e. only some parameters must be re-tuned). This allows to directly compare both embodiments (i.e. flexible and rigid feet). This straightforward adaptation is usually not possible for most robotic gaits, requiring controller rules modifications.

Importantly, the ground contact model used in all the developments of this thesis (being detailed in Section G.5.1) is compliant. Yet, other chapters only consider rigid feet impacting this compliant ground. In this chapter, an extra compliance is added to the flexible feet developed here (i.e. on top of the compliance inherent to the ground contact model).

5.1 Introduction

Comparing human and humanoid robot locomotion rises many questions about what is missing in these machines to achieve a proper human-like walk. In fact, most of the humanoid robots walk with bended knees and feet kept parallel to the ground. Hence, they are limited in step length and they lack in robustness when subject to environmental disturbances. These questions have many possible answers concerning the differences in cognition capabilities, control strategies and mechanical properties. In this study, we focus on the big gap existing between the human foot and the traditional solutions used in humanoid robotics.

The human foot has a very complex structure composed of more than 100 muscles and tendons interconnecting 26 bones and a total number of almost 30 joints (Hicks, 1954). Such a rich bio-mechanical design gives to the foot many interesting properties and functionalities. For instance, the shape and the mobility of the bones of the foot play a key role in absorbing internal and external impacts (Chung et al., 2009). Moreover, the soft tissues of the plantar arch contribute to absorb external impacts and damp out vibrations. Finally, the Windlass mechanism (Hicks, 1954) (a passive mechanism embedded in the foot) prevents the plantar arch to collapse under the body weight. Moreover, it kinematically constrains the toes-flexion and plantar-flexion. Therefore, it contracts the foot arch when the foot rolls over the toes.

On the contrary, most humanoid robots, existing nowadays, are equipped with rigid and flat feet. As a consequence, the control efforts to achieve stable walking lie entirely on the upper body. This results, generally, in a very unnatural walk. Nevertheless, more advanced feet for humanoid robots already exist. A common practice to increase the shock absorbance at the lower-limb extremities consists in adding soft material (for instance rubber) under the sole of the foot of the robot (Li et al., 2008). Example of multi-body foot design are, among others, reported in (Yang et al., 2008), (Seo and Yi, 2009), and (Davis and Caldwell, 2010). Unfortunately, these more complex designs are still not explored on real humanoid robots. In fact, the most used walking strategies rely on the computation of dynamic stability indicators (such as the zero-moment point (Vukobratovic and Borovac, 2004)) that suppose to have the foot flat on the ground during the whole stance phase.

Passive prosthetic feet for amputees are developed to allow the user to walk comfortably at a specific nominal speed. In fact, they are designed to replicate the functionalities of the human foot introduced beforehand and to reproduce a similar behaviour. Figure 5.1a depicts the Flex-Foot® Junior prosthesis. It is composed of two carbon fiber elements (a long element and a smaller one) connected in the middle by two screws. The overall shape appears similar to the shape of the human foot having the heel, forefoot, foot tomb and the plantar

arch. However, the similarity is also functional. In fact, pressing on the foot tomb, the longer element bends pushing down the heel. It is similar to what happens on the human foot due to the Windlass mechanism. Moreover, the round shape and the stiffness of the heel and the foot toes are selected to have a proper foot-roll movement during the walk. The cosmetic cover, in Figure 5.1b, is used to allow the prosthesis to wear normal shoes. However, it also adds extra compliance at the heel and the foot toes and increases the impulse absorption, similarly to the soft tissues of the human foot.

(a) (b)

Figure 5.1: (a) the Flex-Foot® Junior prosthesis; (b) the cosmetic cover

The motivation of this study comes from the wish to quantify the benefits that can be obtained by using more human-like foot designs in humanoid robotics. To this end, we extract the physical characteristics of a prosthetic foot to develop a human-like foot model. In (Ker et al., 1987), the authors characterized the physical properties of the human foot measuring them directly on an amputated foot. More recently, in (Ito et al., 2014), the authors measured the kinematic of the bones in an amputated foot while performing a walking gait. In our approach, we use a commercial prosthesis instead of directly analyzing the human foot. This solution has many advantages. Among others, we inherit the knowledge of prosthetic manufacture in the design of a device reproducing the main features of a human foot and we produce data which are reproducible in any other lab. However, the characteristics of the passive prosthesis are fixed and cannot be adapted to different gaits.

In (Van der Noot et al., 2015a), we achieved human-like walking using neuromechanical primitives. More specifically, the walking controller is based on a set of virtual muscles activated by reflexes. Exploiting the principles of legged mechanics and muscle activations, the robot was able to walk on flat ground, exhibiting some human-like features as stretched stance leg and rolling feet. In this study, we use a similar muscle-reflex based controller together with the human-like foot. We optimize several walking gaits on a wide range of walking speeds. Respect to previous attempts to achieve human like walking using flexible feet, such as (Ogura et al., 2006) and (Hashimoto et al., 2010), we are able to reach higher walking speeds. Our walking strategy is systematically tested on flat ground and uneven terrain to evaluate the walking efficiency and robustness. In (Song and Geyer, 2011), the authors conduct a similar study producing interesting results on the energy consumption of a compliant foot. Our analysis focuses not only on the energy but also on other gait features and fundamental aspects of walking such as terrain adaptation. Finally, we compare the results obtained using

human-like compliant feet with the ones obtained using rigid feet.

In the following section, the foot designs are presented. In Section 5.3, the principles of the muscle-reflex controller are explained. More details on the optimization process and the different scenarios used to evaluate the foot designs are collected in Section 5.4. Finally, the results are reported and discussed in Sections 5.5 and 5.6.

5.2 Foot models

For the purposes of this study, we extract the physical characteristic of a prosthetic foot to develop a human-like foot model. Beside, two variations of the rigid foot are proposed and used as benchmark in the performance evaluation. In this section, the three foot designs are described.

5.2.1 The Human-Like Foot

The dual-profile prosthetic design of the selected prosthesis is one of the most common among passive prosthetic devices. It is well known in the prosthetic scientific community as well. Hence, it allowed us to study the design strategy behind it. More specifically, the Flex-Foot® Junior, produced by Össur ehf, is available in three different versions according to the weight of the patient and in different sizes according to the body of the patient. Considering the 30 kg of mass and the 5-years-old body proportion of our robot, COMAN (Tsagarakis et al., 2013), the Flex-Foot® Junior is the most appropriate commercial prosthesis available on the market.

The prosthesis interacts with the ground in three different configurations during a step. At the heel-strike, just the hindfoot is touching the ground, hence, the heel profile bends and adapts to the terrain. The force (F_h) that the hindfoot exhibits, strictly depends on this deflection. At the toe-off, just the forefoot is in contact with the ground, and then it bends. The force exerted (Ff) mainly depends on the corresponding deflection. During the stance phase, both parts (the hindfoot and the forefoot) are in contact. Hence, the total force exerted is equal to $F_h + F_f$.

A schematic of the model used to represent this behavior is depicted in Figure 5.2a. K_h and K_f are respectively the vertical heel stiffness and the vertical forefoot stiffness assumed non linear. Hence, the vertical forces exerted at the hindfoot and forefoot can be expressed as follows:

$$F_h(\delta_z) = K_h(\delta_h)\delta_h \qquad F_f(\delta_z) = K_f(\delta_f)\delta_f \tag{5.1}$$

where δ_h and δ_f are respectively the heel deflection and the forefoot deflection of the two vertical springs. The red plates, attached to the springs (see Figure 5.2a), are shaped as the real sole of the prosthetic foot. They are constrained horizontally, hence they can only move

vertically according to the springs deflection (δ_h and δ_f). The sole profile is very important because it affects the motion of the center of pressure during the step. Hence, it affects the motion dynamics of the walker too. For instance, the curvature of the heel and the forefoot affects the foot roll movement during heel strike and toe off respectively. Moreover, having different values of stiffness for the hindfoot and the forefoot, affects the force distribution below the sole and moment at the ankle joint. The energy storage capability directly depends from those values as well.

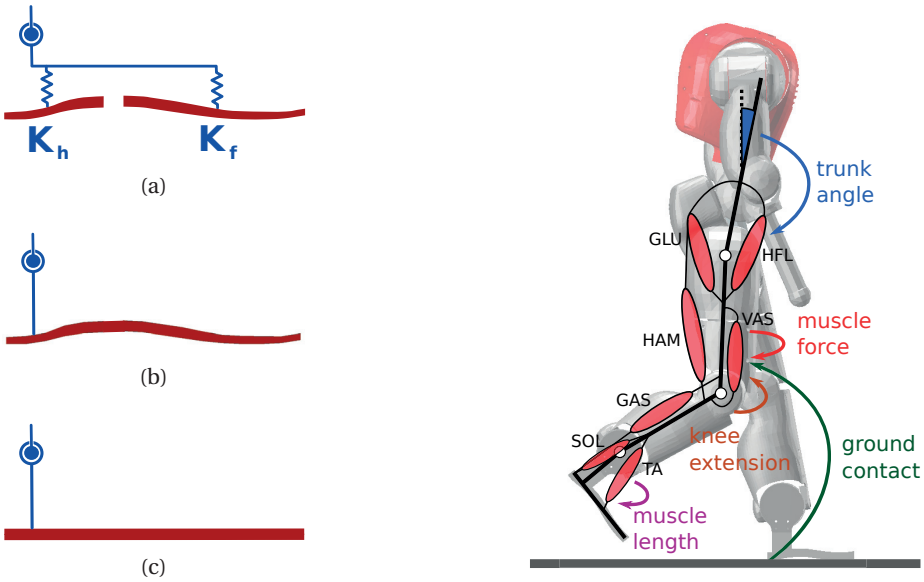


Figure 5.2: Left panel: Three foot designs: (a) the human-like foot; (b) the rigid-shaped foot; (c) the rigid-flat foot. Right panel: COMAN robot with the seven muscle groups of the right leg and some inputs of the controller reflex rules. Muscles: soleus (SOL), tibialis anterior (TA), gastrocnemius (GAS), vasti (VAS), biarticular hamstring (HAM), gluteus (GLU) and hip flexor (HFL).

In order to compute Equation (5.1), we experimentally compute the functions $K_h(\delta_h)$ and $K_f(\delta_f)$. The plot in Figure 5.3 collects the data relative to the heel. The quasi-static force-displacement characteristic of the heel is represented with a dotted black line. As expected, the characteristic is not linear. Moreover the compression characteristic differs from the extension characteristic, mostly due to the hysteresis properties of the cosmetic cover (van Jaarsveld et al., 1990). For the purposes of this study, we do not model the hysteresis of the prosthetic foot. In fact, rather than an highly accurate identification of this specific prosthesis, we are interested in capturing the general behaviour. Hence, we approximate this curve with a second-order polynomial function (represented with a solid blue line in the plot) using the least-squares algorithm. The heel stiffness $K_h(\delta_h)$ is approximated with the derivative of this function. In a similar way we compute the forefoot stiffness function $K_f(\delta_f)$. In Figure 5.4, the quasi-static force-displacement characteristic of the forefoot is represented with a dotted black line. This curve is similar to the one in Figure 5.3. The second-order approximation of the characteristic is represented with a solid blue line.

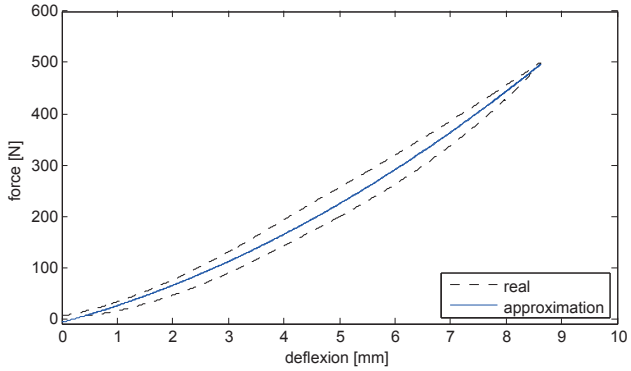


Figure 5.3: Force-displacement characteristic of the hindfoot

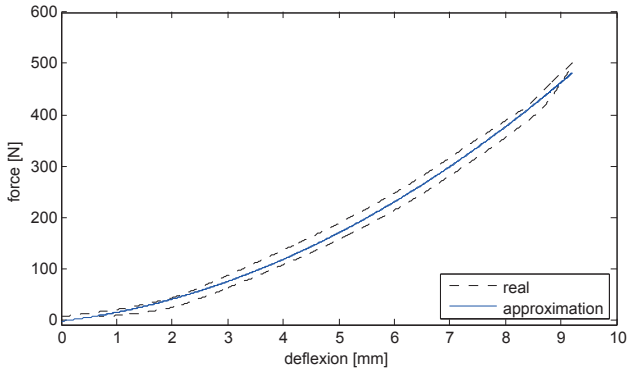


Figure 5.4: Force-displacement characteristic of the forefoot

In this study, we did not characterize the damping of the prosthetic foot. However, we apply low damping forces parallel to f_h and f_f and proportional to $-\dot{\delta}_h$ and $-\dot{\delta}_f$. The main reason is to guarantee the convergence of the sole plates to the equilibrium positions.

5.2.2 The Rigid feet

In Figure 5.2c the first variant of the rigid foot is depicted: the Rigid Flat Foot (RFF). The sole of the foot (in red) is a single element completely rigid and its shape is flat. The total length and the position of the ankle joint are the same as the HLF.

The Rigid Shaped Foot (RSF) is depicted in Figure 5.2b. It is an intermediate version between the HLF and the RFF. In fact, its sole is completely rigid but it has the same shape as the HLF. Proportions are the same as the other two feet.

5.3 The Muscle-Reflex based Controller

The bio-inspired controller presented in (Geyer and Herr, 2010), achieves limit cycle walking on a simplified model of a human. The locomotion is then entirely controlled by a chain of reflexes commanding human muscle groups. We implement a similar controller to get stable locomotion gaits on the COMAN. We focus on 2D walking gaits, hence, no motion is permitted in the lateral and transverse plane. Consequently, all robot non-sagittal degrees of freedom (DOFs) are set to a fixed position. Compared to the simplified human model used in (Geyer and Herr, 2010), we have extra DOFs for the upper body (controlled using the rules described in (Wang et al., 2012)). Moreover, we include the robot series-elastic-actuator dynamics (Mosadeghzad et al., 2012) because we want to derive a controller that could be directly plugged into the real robot.

Seven muscle groups are identified within each leg, as depicted on the right side of Figure 5.2. Each muscle group is represented by a Hill-type muscle, a set of equations developed to fit the behaviour of a real muscle. They are controlled by scalar signals: the neural stimulation signals, being generated by some reflex rules. These rules are a set of equations using different inputs: the trunk absolute angle, the ground contact forces, the knees position and the muscles length and force. These reflex rules require an optimization phase to find a set of unknown parameters, as described in Section 5.4.1.

The different virtual muscles react to the activation signals by contracting. The forces generated are mapped into the joint space considering the segments free-body diagram. Hence, the desired torques are sent as references to a low-level torque controller implemented as in (Mosadeghzad et al., 2012). Finally, the outputs of this low-level controller are used in the motor equations, generating the actual torques.

5.4 Simulation Environment

The dynamic model of the COMAN robot is generated in Robotran (Dallali et al., 2013). It is a software able to model and analyze multibody systems. On top of the robot body dynamics, its actuator dynamics is implemented as referred in Section 5.3 and different feet designs are implemented as presented in Section 5.2.

To fully present the simulation environment, we first detail the controller optimization phase and finally, we introduce the different scenarios used to compare the different feet design performances.

5.4.1 Gait Optimization

The muscle-reflex based controller involves a set of unknown parameters that can be tuned by an optimization process phase. These parameters directly impact the robot gait features, among others, its speed and energy consumption. We aim to compare different foot designs.

To this end, we only compare gaits with the same speed. Consequently, getting a final target speed is one of the requirements of the optimization. We also want the controllers to be energetically efficient. To get more human-like walkers, we aim to minimize the metabolic energy consumption in the virtual muscles per unit distance walked (Bhargava et al., 2004).

The selected optimization algorithm is a heuristic one called Particle Swarm Optimization (Clerc and Kennedy, 2002; Kennedy and Eberhart, 1995). To achieve all the above-mentioned requirements, each set of parameters is tested according to a staged objective function. This means that the different objectives are sorted by order of relevance so that the next objective is taken into account (in the objective function) only when the previous one is fulfilled.

The first stage rewards the gait robustness, by assigning an objective function proportional to the walking time (before a possible fall). A fitness of 100 is assigned to a non falling walk of 60 s. To further improve the gait robustness, we want the robot to keep a foot clearance of at least 1 cm above the ground when walking on a flat surface.

When the COMAN is able to walk without falling during the entire simulation time (i.e. 60 s), the second stage of the objective function is unlocked. This one constrains the gait to achieve a target speed. Different target speeds are tested ranging from 0.4 m/s to 0.9 m/s, which covers the normal walking speed range for a five-years-old child. The objective function f is computed as follows:

$$f = \alpha\, e^{-\beta (x - x^*)^2} \tag{5.2}$$

where x is the forward speed, x^* the target speed and α, β two weight parameters ($\alpha = \beta = 100$). Hence, this function is bounded between 0 and α.

If the robot speed lies within an interval of 0.05 m/s around the target speed, the last stage is unlocked, i.e. the energy minimization. The objective function is formulated as Equation (5.2). However, x^* is set to zero while x is now the metabolic energy consumption per unit distance walked and per mass unit ($\alpha = 100$ and $\beta = 5 \cdot 10^{-6}$). This helps minimizing the absolute energy expenditure.

5.4.2 Optimization and Evaluation Scenarios

To evaluate the feet designs or to optimize their respective controllers, we use different types of ground. The property of the contact model are the same , however, the ground profile changes to fulfill different requirements. Referring to Figure 5.5, we define the following ground profiles:

(a) *Flat ground:* This ground is totally flat with no obstacles and is used to evaluate the different gait features: energy consumption, stride length and stride frequency.

(b) *Uneven terrain:* In order to evaluate the contribution of the different foot designs to the walking stability, we use a flat ground with small bumps. Referring to Figure 5.5, the shape of the bumps can be computed as follows:

$$h(x) = \frac{h_{obs}}{2}\left(1 + cos\left(\frac{\pi}{l_{obs}}x\right)\right) \tag{5.3}$$

where h_{obs} and $l_{obs} = 15\,mm$ are respectively the total height and the half of the base length of the obstacle. Using this sinusoidal shape, there is no slope discontinuity in the ground contact model. The distance between two consecutive obstacles (l_{rnd}) is randomly computed with a flat distribution bounded between 100 mm and 200 mm. Consequently, the feet (150 mm long) can land on a perfectly flat terrain or on a location with one or two obstacles.

Finally, we prevent the robot from falling due to a collision between the toes of the swing foot and a bump. Hence, we make these bumps to only interfere with a landing foot (and

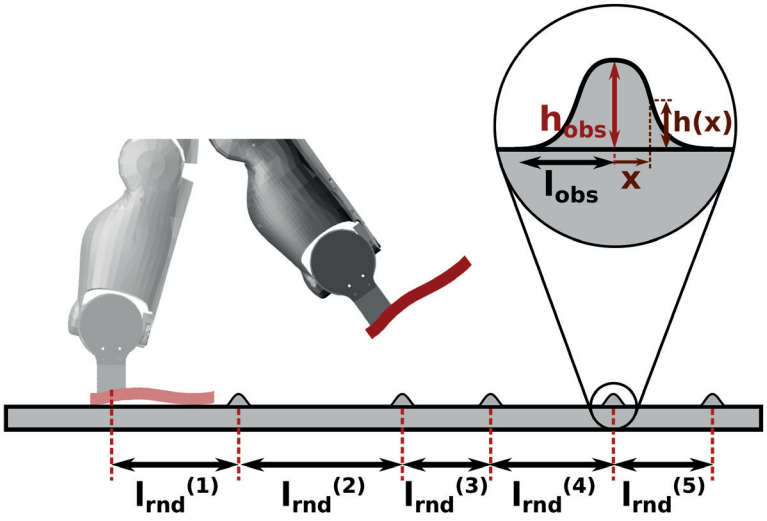


Figure 5.5: Uneven terrain profile.

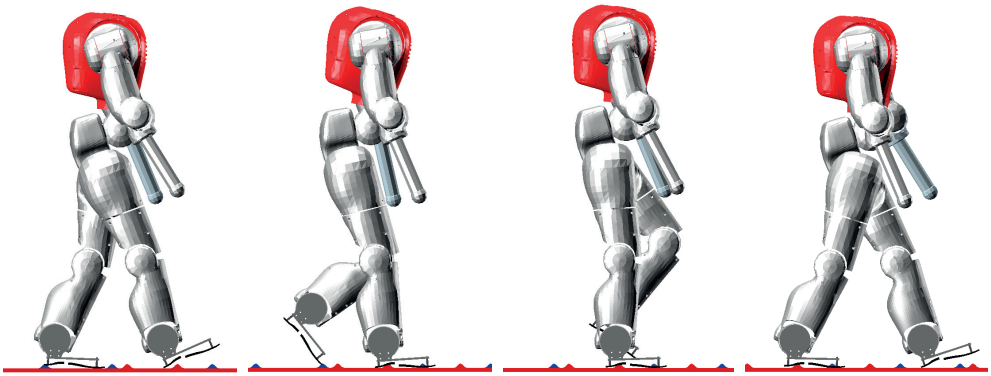
during the stance phase). This does not constitute a loss of generality, for the purposes of this study (see Section 5.6).

5.5 Results

For each foot design and for each target speed, we run five optimizations (with different random initial populations) that converge to as many sets of controller parameters that allow the COMAN to walk at the requested speed on the flat ground for at least 60s.

Snapshots of the resulting walking gait are presented in Figure 5.6. The COMAN performs the heel-strike with its right foot. Meanwhile the left foot rolls over the toe and swings in front of the opposite leg. The supporting leg propels the body forward and the corresponding foot adapts to the uneven ground. The next step starts with the heel-strike of the left foot.

The gait features and walking stability of these controllers with the corresponding foot designs are evaluated in the next section.



(a) heel-strike (b) toe-off (c) single-support (d) heel-strike

Figure 5.6: Snapshots of the COMAN walking on uneven terrain. Blue bumps affect the left foot and red bumps affect the right one.

5.5.1 Gait features

The plot in Figure 5.7a collects the values of the energy consumption measured (as in (Bhargava et al., 2004)) during an eight-meter steady-state walk on flat ground. The data corresponding to the HLF are represented in red and the data corresponding to the RSF and RFF are represented respectively in blue and green. For each target speed and each foot design, we summarize the five optimized controllers by presenting the mean and the standard deviations of their characteristics. The energy values for the two variants of the rigid foot are very close to one another at low speed and slowly diverge at high speed maintaining the same trend. The minimum energy value corresponds to speeds of 0.8154m/s and 0.8126m/s respectively for

RSF and RFF. The rigid human-like shape of the sole lower the energy efficiency of the RSF with respect to RFF. The HLF has higher energy consumption for the whole speed range, and has its lowest values between $0.5073 m/s$ and $0.706 m/s$. Moreover, its standard deviation regularly increases for high speed, demonstrating a higher variability of the controller parameters.

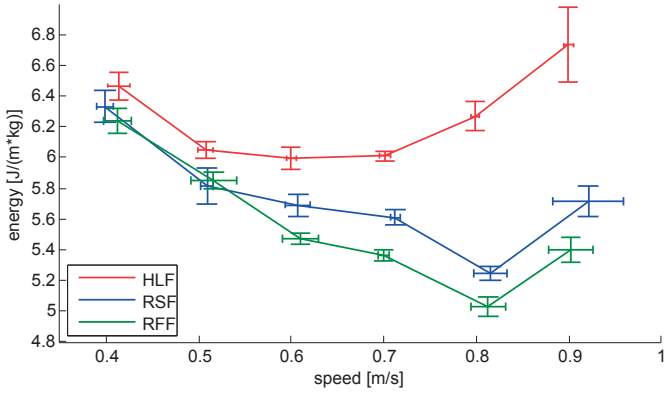
The plots in 5.7b and 5.7c collect respectively the values of the stride duration and the stride length during the walk. As for Figure 5.7a, we present their mean values and the standard deviations. Their trends are mostly monotonic with respect to the walking speed and they are slightly affected by the foot design. The rigid human-like shape of the sole of RSF induces a lower walking frequency and longer stride compared to RFF at any speed. The trend of the walking frequency for the HLF is more flat along the whole speed range. Hence, its stride length compared with the two rigid designs is slightly smaller at lower speed and bigger at high speed.

5.5.2 Walking Stability

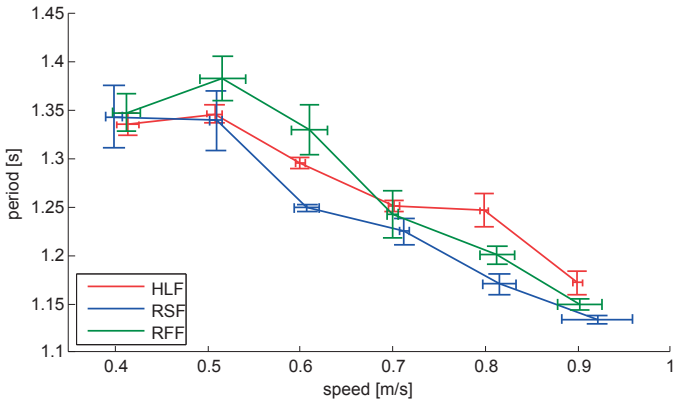
The influence of the foot designs to the robustness of walking is evaluated by an endurance test on uneven terrains. More specifically, the COMAN has to blindly walk on different uneven terrains with different obstacle heights (see Figure 5.5). Hence, the three foot designs are tested on the same terrain conditions and the number of steps performed before falling are considered as index of walking stability. It is important to remark that the controllers are optimized on flat ground (see Section 5.4.2). Therefore, the difference in the number of steps performed are related to the different features of the feet rather than the controller.

The chart in Figure 5.8a collects the results of the endurance tests performed using the controllers optimized for a speed equal to $0.5 m/s$. This is the limit value of the speed range (between $0.5 m/s$ and $0.7 m/s$) having the most efficient gaits on flat ground for the HLF (see Figure 5.7a). For each bump height, the number of steps performed before falling are represented with a red, blue and green bar respectively for the HLF, the RSF and the RFF. The standard deviations are reported too. The HLF results to be the most stable along the whole observed range. The RSF and RFF present very similar results for bump heights bigger than $1 cm$.

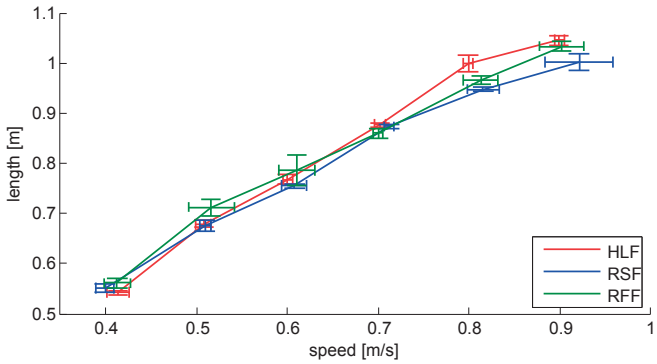
The chart in Figure 5.8b, collects the results of the endurance tests performed at the most efficient gait on flat ground for the rigid feet : $0.8 m/s$ (see Figure 5.7a). The advantage of HLF on the rigid feet is much reduced respect to results of Figure 5.8a.



(a) Energy consumption per mass unit and per unit distance walked

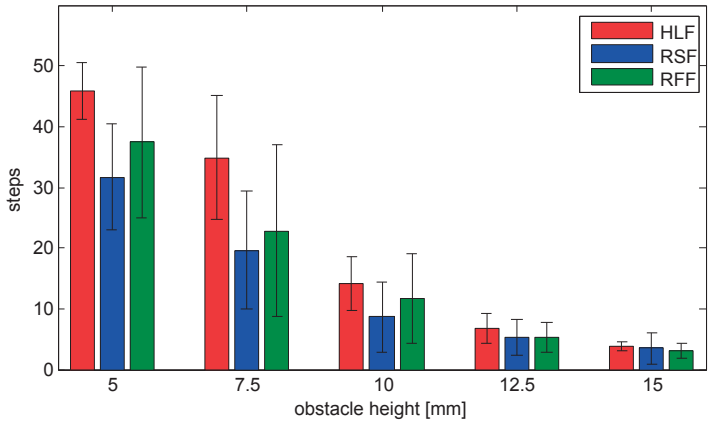


(b) Stride period

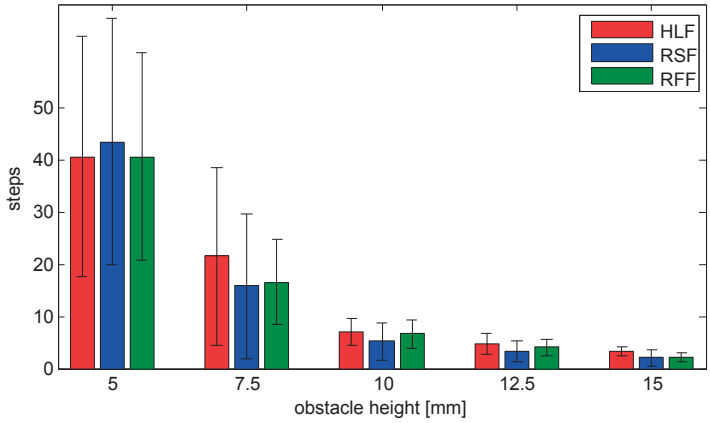


(c) Stride length

Figure 5.7: Mean value and standard deviation of the gait features for the different walking speeds and the different foot designs : Human-Like Foot (HLF), Rigid Shaped Foot (RSF), Rigid Flat Foot (RFF).



(a)



(b)

Figure 5.8: Endurance test results for different walking speed : (a) 0.5 m/s ; (b) 0.8 m/s

5.6 Discussion

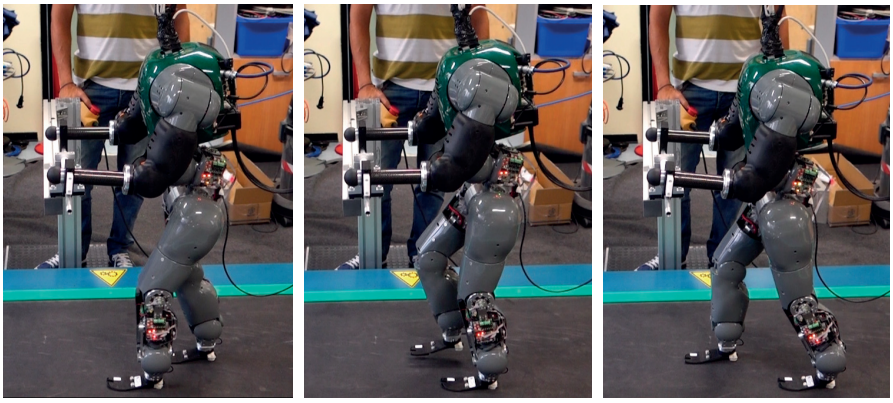
In this work, we investigated the advantages and the drawbacks of implementing human-like compliant feet to a humanoid robot driven by a neuromuscular controller.

The rigid foot results in more energy efficient gaits than the human-like foot. This characteristic is in line with the results presented in other work such as (Song and Geyer, 2011). It has its minimum energy consumption for walking speed close to 0.8 m/s and increases around it. The trend of the human-like design is similar and has its minimum energy between 0.5 m/s and 0.7 m/s. In fact, this model is based on a passive prosthesis that is designed to match the human behavior around a specific range of speeds (between 0.52 m/s and 0.83 m/s for a normal child (Rose-Jacobs, 1983)). Other gait features slightly differ. The shape of the sole affects the walking frequency at slow speed and the compliance mainly affects the straight

length at high speed.

Despite its high energy consumption, the human-like foot has the best performance in terms of stability of the walking. The shape of the sole and the different compliance between the hindfoot and the forefoot probably increase the foot adaptability to the ground uncertainties and reject more efficiently the disturbances with respect to a rigid foot. The data reported in Figure 5.8, refer to a blind walk and no strategies are implemented to guarantee robustness to ground perturbation. This leads to a minimization of the influence of the controller to the walking robustness, and therefore an unbiased evaluation of the performances of the different feet. By implementing extra stabilization strategies (such as stumbling reflexes) or optimizing the control parameters directly on rough terrains, the robustness of the walk could certainly be increased. However, the aim of this study is different.

The results of this study are promising. In fact, using human-like compliant feet together with the neuromuscular controller, we achieved walking gaits in a wide range of speed. Moreover, we showed an increase of robustness of the walk using soft feet respect to rigid ones. The obstacle sizes and the walking speeds, considered in this study, were selecting according to the kid-size proportions of the COMAN robot. In order to compare these results with an adult-size robot, a factor of 2, at least, should be considered.



(a) (b) (c)

Figure 5.9: Snapshots of the real COMAN walk. The robot is constrained on the sagittal plane by its hands.

Future possible work might be to collect experimental data with the real robot implementing the prosthetic feet. Snapshots of a preliminary test performed on the real COMAN are presented in Figure 5.9. This result is obtained using the exact same controller parameters optimized in simulation. In fact, the simulation included full model of the robot dynamics in the sagittal plane, the actuator dynamics and a realistic contact model. Another possible avenue for future developments is to further extend the model of the human-like compliance foot and to deeper study its effects on the walking gait. For instance, the effect of using stiffer prosthesis on the gait characteristics could be investigated.

The remaining chapters of this thesis are centered around controllers developments, in contrast to this chapter (centered around embodiment adaptation). For these next chapters, results are obtained when using rigid feet, therefore favoring energy efficiency over rough grounds robustness. However, all the developments of the next chapters should also work when using prosthetic feet, but this was not investigated.

6 Forward speed modulation during 2D walking gaits

Publication

The material presented in this chapter is adapted from:

Van der Noot N, Ijspeert AJ and Ronsse R (2015) Biped gait controller for large speed variations, combining reflexes and a central pattern generator in a neuromuscular model. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, 26-30 May 2015, pp. 6267-6274. DOI: 10.1109/ICRA.2015.7140079.

As mentioned in Chapter 4, the neuromuscular reflex-based controller of (Geyer and Herr, 2010) could be optimized to reach only a single forward speed. In other words, it was not possible to control the gait after the optimization process. Other publications extended this work with new reflex-based rules, for instance in order to control speed or swing leg placement (Song and Geyer, 2012; Desai and Geyer, 2013).

In this thesis, we explore another avenue to control steering behaviors: the introduction of a central pattern generator (CPG). CPGs are neural circuits capable of producing rhythmic patterns of neural activity without receiving rhythmic inputs (Ijspeert, 2008). Interestingly, they feature interesting properties, among which the possibility to adapt the limit cycle behavior while modulating simple control signals. This makes the use of CPGs extremely appealing to obtain the rich locomotion behaviors we target in this thesis.

In this chapter, the 2D walking gait is extended by the introduction of a CPG. Forward speed can then be controlled by the modulation of a small set of parameters (most of them being related to the CPG) as linear functions of the target speed, resulting in the combined modulation of the step length and frequency.

6.1 Introduction

Dynamic walking gaits with a robot can be achieved by using many approaches. Among them, those relying on the zero-moment point (ZMP), an indicator of dynamic stability (Vukobra-

tovic and Borovac, 2004), are likely the most famous ones. Using this framework, humanoid robots like ASIMO (Chestnutt et al., 2005) or HRP-2 (Kaneko et al., 2002) perform robust gaits. However, controllers based on these approaches usually present some limitations like energy-inefficiency, high computational cost and non human-like features like continuous knee bending (Kurazume et al., 2005; Dallali, 2011).

In parallel, some models consider the human gait as a limit cycle and focus on global stability. This leads to the concept of *limit cycle walking* (Hobbelen and Wisse, 2007). Among these, the neuro-musculo-skeletal model developed in (Geyer and Herr, 2010) relies on reflex-based controlled muscles generating torques at the joint level. Interestingly, this approach can generate robust and energy-efficient gaits similar to the human ones in terms of muscles activities, joint angles and torques.

The reflex rules developed in (Geyer and Herr, 2010) require an optimization phase (or manual tuning) of the many open parameters governing the contribution of each local reflex. This approach makes the optimized parameters set working for a single gait speed. When the robot is walking, it is thus not easy to change its speed or its stride length.

A first strategy to overcome this limitation was developed in (Song and Geyer, 2012). It consists in optimizing different gaits and then modulating some key control parameters to change the forward speed during the walking gait. However, this approach can only cope with small speed changes after a first optimization. High speed variations require then running extra optimizations to find new parameter modulations between pre-optimized walking gaits.

Another approach developed in (Dzeladini et al., 2014) introduces a central pattern generator (CPG), a neural circuit capable of producing rhythmic neural activity patterns without receiving rhythmic inputs (Ijspeert, 2008). The CPG is used as a feedback predictor of the reflex rules from (Geyer and Herr, 2010). This CPG can thus be used as a feed-forward component, reducing the complexity in the speed control strategy of (Song and Geyer, 2012). Nevertheless, this approach requires to capture (with third order spline interpolations) the reflex rules outputs that were optimized for one precise walking speed with no feed-forward contribution. Consequently, the gait is not optimal (regarding energy-efficiency) for the whole range of speeds. On top of that, this leads to a speed transition range being smaller than the one from (Song and Geyer, 2012).

In this contribution, we also propose a controller mixing reflexes and a CPG to control the leg muscles. Our CPG is designed as a six-neurons network of Matsuoka oscillators (Matsuoka, 1985, 1987) sending feed-forward signals to the proximal muscles controlling the hip. These bio-inspired artificial oscillators, capturing the mutual inhibition between half-centers located in the spinal chord, are widely used to model the firing rate of mutually inhibiting neurons, in both the upper and the lower extremities (Ronsse et al., 2009).

The controller can then be optimized for a large range of walking speeds, co-optimizing reflexes and CPG parameters at the same time within a single optimization. We applied this approach to a simulation of the COMAN, a humanoid robot presented in Section 6.2. Then, in Section 6.3, we detail the controller itself and the associated optimization process, while Section 6.4 presents the strategy used to adapt the robot speed during the walking gait. Section 6.5 analyses the resulting gaits, comparing them to the ones obtained with the original model of (Geyer and Herr, 2010). Results about speed transitions, strike prediction and holes stepping techniques are also presented. Finally, we conclude the chapter in Section 6.6.

6.2 COMAN platform

We use a simulation model of the 95 cm tall COmpliant HuMANoid platform (COMAN). This robot, developed by the Italian Institute of Technology (IIT), has 23 actuated degrees of freedom (DOFs), most of them being equipped with series elastic actuators (Pratt and Williamson, 1995). Each joint is equipped with position, velocity and torque sensors. The robot also features an inertial measurement unit (IMU) and 6-DOF feet force and torque sensors measuring ground reactions. Our controller only uses sensory inputs available on this robot. Further information can be found in (Dallali et al., 2013) and (Tsagarakis et al., 2013).

The COMAN (visible in Figure 6.1b) is modelled in a simulation environment called Robotran (Samin and Fisette, 2003). Its actuators implementation is described in (Dallali et al., 2013). In this contribution, we artificially constrain the waist to stay in the world sagittal plane to study 2D walking gaits only.

6.3 Controller design

The purpose of our controller is to produce position or torque references for each of the 23 DOFs of the robot. We briefly describe the control rules for the main of them before focussing on the three sagittal joints of each leg.

6.3.1 Joints control

The COMAN has eleven DOFs for the upper body: three in the torso and four per arm. All torso joints are controlled to track zero position. For arms control, we use similar rules as the ones presented in (Wang et al., 2012). In short, constant position references are tracked for the elbow sagittal DOF (0.25 rad), the shoulder lateral DOF (0.09 rad) and the shoulder transverse DOF (0.14 rad). Finally, the shoulder sagittal DOF ϕ_s^s tracks a linear function of the hip angles difference as $\phi_s^s = 0.3 * (\theta_h^R - \theta_h^L) - 0.3$ where θ_h^R and θ_h^L are the right and left sagittal hip positions (θ_h^R and θ_h^l are inverted for the left shoulder), all expressed in radians. Such control leads to balancing arm trajectories reducing the total energy consumption during the walking gait (Wang et al., 2012).

Regarding the lower body, each leg has three sagittal DOFs, two lateral DOFs and one transverse DOF. All leg non-sagittal joints track zero position, to comply with the 2D walking. Finally, the three sagittal joints, being the main focus of this contribution, rely on the neuromuscular model controller described in Sections 6.3.2 to 6.3.5. The initial posture of the robot constraints all joints to track zero position which corresponds to an upright posture, except the sagittal ankle tracking a position κ to be optimized (see Table E.1 in Appendix E.2).

6.3.2 Musculo-skeletal model

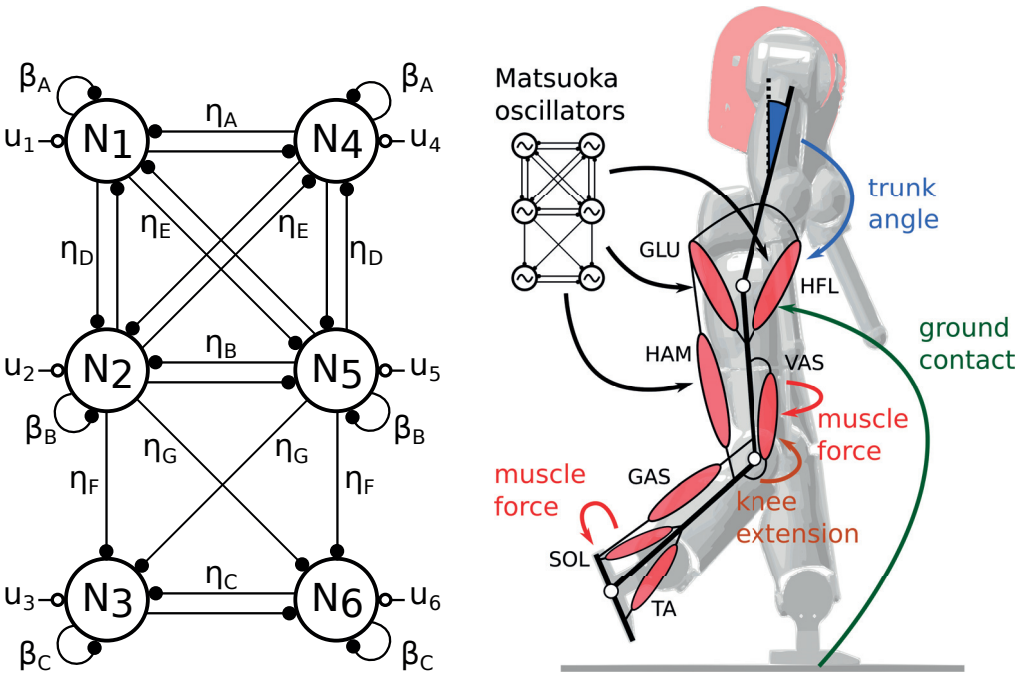
We focus here on the leg sagittal DOFs which are the most important joints propelling the body forward. The model proposed by Geyer and Herr (Geyer and Herr, 2010) actuates each leg with seven Hill-type muscles, capturing the contribution of the main muscle groups of the human leg. For our COMAN model, these virtual muscles are depicted in red in Figure 6.1b, producing torque references in a way similar to (Geyer and Herr, 2010). The key idea is the following: muscles react by contracting and apply forces on the body. Therefore, the equivalent torques applied by the seven muscles on the leg sagittal joints are computed from the segments free-body diagrams. These torque references are sent to a PI controller feeding the actuators (implemented like in (Dallali et al., 2013)). Their state computation is fully described in (Geyer et al., 2003) and (Geyer and Herr, 2010). The main muscle properties are scaled to fit the size and the weight of the COMAN, using dynamic scaling (Schepelmann et al., 2012).

Each muscle is then controlled by its activation $A_m(t)$, capturing the neural signal provided by motoneurons. This signal is related to a neural input $S_m(t)$, the muscle stimulation, using a first-order low-pass filter capturing the excitation-contraction coupling (Geyer et al., 2003). Controlling the muscle model thus reduces to designing control rules for the stimulations $S_m(t)$ driving the seven muscle groups of each leg.

6.3.3 Central pattern generator design

Central pattern generators (CPGs) are neural circuits capable of producing rhythmic patterns of neural activity without receiving rhythmic inputs (Ijspeert, 2008). They present attractive properties like distributed control, redundancies handling, and locomotion modulation using simple control signals (Ijspeert, 2008). While locomotor CPGs have been identified in many vertebrates, their recruitment for human locomotion is still a matter open to discussion (Dimitrijevic et al., 1998). Yet, computational models show that CPGs could play a major role in human locomotion. For instance, (Taga, 1994) demonstrated bipedal locomotion ability to adapt to a changing environment using CPG modulations. The work of (Paul et al., 2005) proposed a neuro-musculo-skeletal model studying the effects of spinal cord injury on locomotor abilities, again with a CPG as central element.

In this contribution, a CPG structure is used to send descending feed-forward signals to proximal muscles, i.e. muscles driving the hip joint. This is coherent with the *proximo-distal*



(a) Oscillators network

(b) COMAN stimulations

Figure 6.1: The six-neurons oscillators network sends stimulations to the proximal muscles, while the distal ones are only driven by reflexes. The hip flexor muscles (HFL) are stimulated by both the CPG and the reflexes.

gradient hypothesis postulating that CPGs mostly drive the proximal muscles while the distal ones should be driven by reflexes (Dzeladini et al., 2014). Indeed, distal muscles are more impacted by external perturbations like ground interactions (Daley et al., 2007). The three proximal muscle groups controlled by the CPG are the hip flexors (HFL), the gluteus muscle group (GLU) and the biarticular hamstring muscle group (HAM), presented in Figure 6.1b.

The firing rate x_i of each neuron N_i is computed according to Equation (6.1) where v_i is the self-inhibition modulated by an adaptation constant β_j, u_i the external excitation and τ is a time constant. The connexion strengths η_k tune the mutual inhibitions. The $[\bullet]^+$ function takes the positive part of its argument (it saturates to zero when the argument is negative) and thus captures the fact that the activation of a given neuron decreases when another is active (mutual inhibition). Figure 6.1a depicts the Matsuoka network with six neurons N_i that is used to drive these virtual muscles, along with the parameters β_j, u_i and η_k.

$$\dot{x_i} = \frac{1}{\tau}(-x_i - \beta_j v_i - \sum_{1}^{3} \eta_k [x_l]^+ + u_i) \tag{6.1}$$

The self-inhibition computation is captured by Equation (6.2) where γ_j is a constant multiplying τ. The index i corresponds to the neuron index, while the indexes k and l in Equation (6.1)

are replaced according to Figure 6.1a. Index j equals A for neurons N_1 and N_4, B for N_2 and N_5 and C for N_3 and N_6. These rules are fully developed in Appendix E.1.

$$\dot{v}_i = \frac{1}{\gamma_j \tau}(-v_i + [x_i]^+) \tag{6.2}$$

This network obeys a mirror symmetry due to the symmetry of the right and left legs. This symmetry between neurons N_1, N_2, N_3 and neurons N_4, N_5, N_6 can be observed in the mutual inhibition connexions strength η_k in Figure 6.1a and also in the full equations development provided in Appendix E.1.

Neurons N_1, N_2, N_4 and N_5 form a fully-connected network where each neuron fires alternatively over the cycle. These neurons will stimulate the HAM and GLU muscles of each leg. Neurons N_3 and N_6 receive inputs from them but do not interfere on this first fully-connected network. In this way, their respective own parameters β_j, γ_j and η_k provide more flexibility to stimulate the HFL muscles.

Similarly to (Paul et al., 2005), this CPG can also be modulated by the interactions between the robot body and its environment. This is done via short excitations modulations at foot strike. The input excitations u_i of the neurons first consist in a tonic excitation equal to u. For simplicity, this tonic contribution is kept equal to 1. Modulations of the oscillators output will rather be governed by external gains (see Equation (6.5)). Some terms are further added to the excitation component in order to achieve synchronization between the oscillators and the walking gait. In particular, the firing rate x_1 is expected to switch from a negative to a positive value at the moment of right foot strike. Similarly, x_4 is expected to become positive at left foot strike.

This results in the equations presented in (6.3). The function $[\bullet]^-$ takes the absolute value of its argument if it is negative, and saturates to zero otherwise. On top of that, the function $[\bullet]_{SR}$ keeps its argument intact during the right leg supporting phase, while saturating it to zero otherwise (similar for the left leg with $[\bullet]_{SL}$). Then, $[\bullet]_{Str,R}$ always saturates its argument to zero, except after the right foot strike if the firing rate x_1 is still negative. In this case, it keeps its argument intact as long as x_1 is negative (similar for $[\bullet]_{Str,L}$ with the left leg and x_4). Finally, the excitation u_i of all neurons is forced to zero when x_1 becomes positive before right strike or when x_4 becomes positive before left strike, again in order to achieve the desired synchronization.

$$\begin{aligned}
u_1 &= u - [x_1]^+_{SL} + [x_1]^-_{Str,R} & u_4 &= u - [x_4]^+_{SR} + [x_4]^-_{Str,L} \\
u_2 &= u - [x_2]^+_{SL} - [x_2]^+_{Str,L} & u_5 &= u - [x_5]^+_{SR} - [x_5]^+_{Str,R} \\
u_3 &= u - [x_3]^+_{SL} & u_6 &= u - [x_6]^+_{SR}
\end{aligned} \tag{6.3}$$

The terms $-[\bullet]^{+}_{SR/SL}$ are used to make each neuron firing rate synchronizing with the appropriate leg. In steady-state, this term is thus always zero. The terms $\pm[\bullet]^{\mp}_{Str,R/L}$ are used when the oscillators are too slow. Then, a burst is provided to the late neurons while others are partially inhibited. On the contrary, if the oscillators are faster than requested, all excitations are forced to zero so that all firing rates will slowly converge to zero. Again, in steady-state, the contribution of these synchronization terms is very limited. These mechanisms achieve the synchronization between the different neurons. Interestingly, these synchronization mechanisms make the oscillators able to predict when the next strike will happen. Some associated results are presented in Section 6.5.4. Walk initiation is simply achieved by sending an excitation of 1 to two neurons (N1, N3 or N4, N6) while the other excitations are set to zero. After 0.2 s, all excitations are activated, as previously explained.

Finally, the oscillators produce four outputs y_i, taken as the difference between the positive part of the firing rates x_i of two adjacent neurons, see Equation (6.4). This arrangement is designed to feed the appropriate signals to the different muscles during the different walk phases (e.g. high stimulations to the HFL muscles during early swing to flex the corresponding hip), see the equations in (6.5).

$$\begin{aligned} y_1 &= [x_1]^+ - [x_2]^+ \qquad y_3 = [x_4]^+ - [x_5]^+ \\ y_2 &= [x_3]^+ - [x_2]^+ \qquad y_4 = [x_6]^+ - [x_5]^+ \end{aligned} \tag{6.4}$$

6.3.4 Muscle stimulations

Muscles stimulations are computed as combinations of CPG output signals y_i, reflex rules and prestimulations S_0. This combination is presented in Figure 6.1b. The stimulations are all bounded between 0.01 and 1. All the reflex rules are adapted from (Geyer and Herr, 2010). However, using oscillators to feed the proximal muscles allows to drastically reduce the number of reflex rules.

The stimulations of the three proximal muscle groups HFL, GLU and HAM, respectively $S_{HFL,R/L}$, $S_{GLU,R/L}$ and $S_{HAM,R/L}$ for the right/left leg are linear combinations of the CPG output signals y_i positive part, see (6.5). k_{HFL}, k_{GLU}, $k_{HAM,1}$ and $k_{HAM,2}$ are four gains presented in Appendix E.3.

$$\begin{aligned} S_{HFL,R} &= k_{HFL}\,[y4]^+ & S_{HFL,L} &= k_{HFL}\,[y2]^+ \\ S_{GLU,R} &= k_{GLU}\,[y1]^+ & S_{GLU,L} &= k_{GLU}\,[y3]^+ \\ S_{HAM,R} &= k_{HAM,1}\,[y1]^+ + k_{HAM,2}\,[y3]^+ \\ S_{HAM,L} &= k_{HAM,1}\,[y3]^+ + k_{HAM,2}\,[y1]^+ \end{aligned} \tag{6.5}$$

On top of that, the HFL muscles receive an extra stimulation S^{ext}_{HFL} coming from (Geyer and Herr, 2010) to help maintaining the trunk to a desired reference position θ_{ref} (in radians):

$$S^{ext}_{HFL} = \xi_1 \frac{F_z}{w} (\theta_{ref} - \theta_t - \xi_2 \dot{\theta}_t) \tag{6.6}$$

where θ_{ref}, ξ_1 and ξ_2 are three parameters to be optimized, θ_t is the trunk absolute angle, $\dot{\theta}_t$ its derivative, w the whole robot weight and F_z the vertical force under the foot of the corresponding leg. This extra stimulation is a reflex similar to a PD controller stabilizing an inverted pendulum.

Two muscle groups only receive a constant prestimulation S_0: the gastrocnemius (GAS) and the tibialis anterior (TA). S_0 is put to 0.01 (the minimal stimulation). Finally, the vasti muscles (VAS) and the soleus muscles (SOL) receive only the prestimulation during swing, while receiving positive force feedback reflexes during stance (Geyer and Herr, 2010):

$$\begin{aligned} S_{VAS} &= S_{0,VAS} + G_{VAS}(F_{VAS}/F_{VAS,max}) \\ S_{SOL} &= S_0 + G_{SOL}(F_{SOL}/F_{SOL,max}) \end{aligned} \tag{6.7}$$

where G_{VAS} and G_{SOL} are two parameters to be optimized, $F_{VAS}/F_{VAS,max}$ and $F_{SOL}/F_{SOL,max}$ are the normalized forces produced by these two muscles and $S_{0,VAS}$ is the prestimulation of the VAS muscle, which is optimized and can exceed 0.01. Finally, S_{VAS} is set to $S_{0,VAS}$ when the corresponding leg is the trailing leg during the double support phase or if the corresponding knee ankle ϕ_k exceeds an over-extension threshold ϕ_{off} (to be optimized) while $\dot{\phi}_k$ is positive (see (Geyer and Herr, 2010)). This prevents knee over-extension.

6.3.5 Optimization of the gait controller

We use a particle swarm optimization (PSO) algorithm to optimize the open parameters (Kennedy and Eberhart, 1995). These parameters are listed in Appendix E.2, Table E.1, along with their bounds. All optimization runs simulate a 60 s walking gait. To encourage solutions with enough foot clearance with the ground, obstacles are added below the swing foot during optimization. These bumps are trapezoidal shapes placed next to the foot in contact with the ground. Their height linearly increases with the simulation time from 0 cm to 3 cm. The objective function used to evaluate each set of parameters is staged in the sense that different objectives are sorted by order of relevance, such that the next objective is taken into account only when the previous one is fulfilled.

The first stage requires the robot to walk without falling during the 60 s simulation time, the fitness being proportional to the walking time. When this objective is reached, the speed is later optimized to match a target speed. The corresponding objective function is given

in Equation (6.8), where f is the objective function, x the parameter to be constrained (the speed here), x^* is the target parameter and α, β are two weight parameters. This output is then bounded between 0 and α. When the robot speed is in a range of 0.05 m/s around the target speed, the last fitness stage is triggered. There, we minimize the metabolic energy consumption in muscle contraction per unit distance walked (Bhargava et al., 2004) and the oscillators prediction error, summing their corresponding objective functions. Indeed, a good oscillator phase prediction is potentially relevant to develop other mechanisms requiring gait synchronization, see Section 6.5.4. In both cases, we also use Equation (6.8) where x now represents the metabolic energy consumption per unit distance walked or the time error between the oscillator strike prediction and the actual one. In both cases, the target x^* is equal to zero, since the objective is to minimize both the energy consumption and the prediction error.

$$f = \alpha\, e^{-\beta(x-x^*)^2} \tag{6.8}$$

6.4 Speed adaptation

In this section, the controller is further extended with the objective to optimize several gaits (corresponding to a range of different speeds) within a single optimization. This will allow the modulation of speed when the robot is walking. This speed modulation is mainly performed by adapting two features of the CPG: frequency and amplitude. Indeed, faster walking speeds usually correspond to faster walking frequencies and longer step lengths (Murray et al., 1966). Moreover, faster speeds result in larger trunk tilt, as identified in (Song and Geyer, 2012). Consequently, the trunk angle reference θ_{ref} acting on the HFL muscles also needs to be adapted as a function of the desired speed, θ_{ref} increasing for faster gaits. The oscillators frequency is tuned with the time constant τ, decreasing with higher speeds. To modify the oscillators outputs amplitude, we adapt the gains k_{HFL}, k_{GLU}, $k_{HAM,1}$ and $k_{HAM,2}$, using different adaptation laws for each of them. Indeed, an increase in speed does not necessary result in an uniform scaling of the muscles stimulations. In this case, it is less trivial to predict how these parameters will evolve with the robot speed.

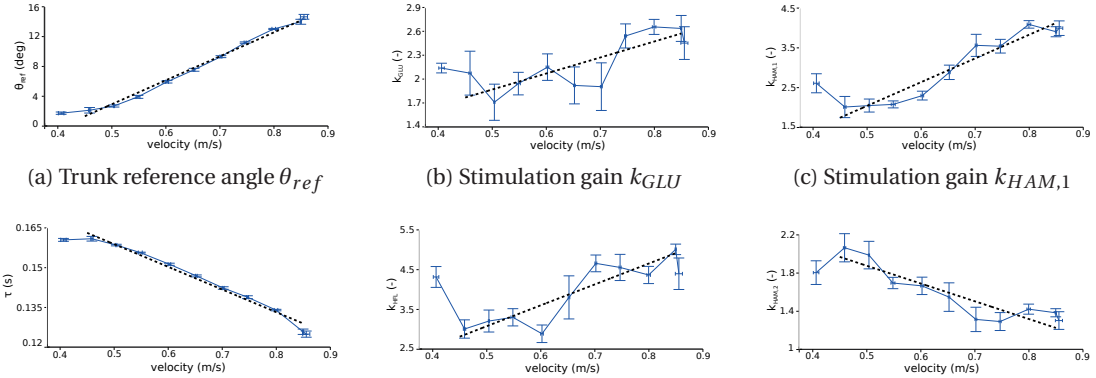
The evolutions of these three types of parameters is studied in Section 6.5.1. It appears that they can be approximated by linear functions of the target speed, according to the rules described in Appendix E.3, which are used to extend our controller to speed adaptation. So, the strategy is the following. The optimization process always targets an arbitrary speed of 0.6 m/s for initiation. After four steps, the speed parameters are updated for a new target speed according to the rules described in Appendix E.3. Each set of optimized parameters is then tested for a large range of different speeds, and the objective function is set as the average of each trial. In this way, we co-optimize all parameters for the largest possible range of speeds (from 0.4 m/s to 0.9 m/s).

6.5 Results

The CPG-based controller presented in this contribution is compared with the reflex-based approach from (Geyer and Herr, 2010). To this end, we first study the evolution of the parameters presented in Section 6.4. We also present the ability of our CPG-based controller to track a speed reference, to predict when the next strike will happen and to step over holes.

6.5.1 Speed parameters

Six parameters were identified in Section 6.4 to be adapted as a function of the walking speed (θ_{ref}, τ, k_{GLU}, k_{HFL}, $k_{HAM,1}$ and $k_{HAM,2}$). First, an optimization with an arbitrary target speed of 0.6 m/s is launched. Then, all the optimized parameters are frozen, except the six parameters being left for speed adaptation. New optimizations were then performed, allowing only these six parameters to change across the different target speeds. Our target speed experiments run from 0.4 m/s to 0.9 m/s with a 0.05 m/s step. We run five optimizations for each target speed and report the evolution of the six parameters in Figure 6.2.



(a) Trunk reference angle θ_{ref} (b) Stimulation gain k_{GLU} (c) Stimulation gain $k_{HAM,1}$

(d) Oscillator time constant τ (e) Stimulation gain k_{HFL} (f) Stimulation gain $k_{HAM,2}$

Figure 6.2: We run five optimizations for each target speed and we measure the actual speed of each solution, along with the optimized value of the six open parameters. For each target speed, we gather the five optimization final results, presenting their mean and standard deviations. For graph legibility, the errorbars represent half of the standard deviations. Dashed lines present the linear approximations for the range of speeds between 0.45 m/s and 0.85 m/s, using the minimum mean square error method.

Let's first take a look at the speeds ranging from 0.45 m/s to 0.85 m/s. There, the evolution of θ_{ref} and τ can be captured by linear functions of the speed. On top of that, this matches the expectations: θ_{ref} increases with speed while τ decreases. Similar observations can be performed for the amplitude gains: k_{GLU} and $k_{HAM,1}$ impact the stance phase while k_{HFL} and $k_{HAM,2}$ impact the swing phase. During stance phase, k_{GLU} and $k_{HAM,1}$ are both used to bring the trunk back after foot strike. This requires higher stimulations at high speeds where inertia effects and strike impacts are more important. While this increase trend is clearly visible for $k_{HAM,1}$, it is less obvious for k_{GLU}. On top of that, the linear approximation

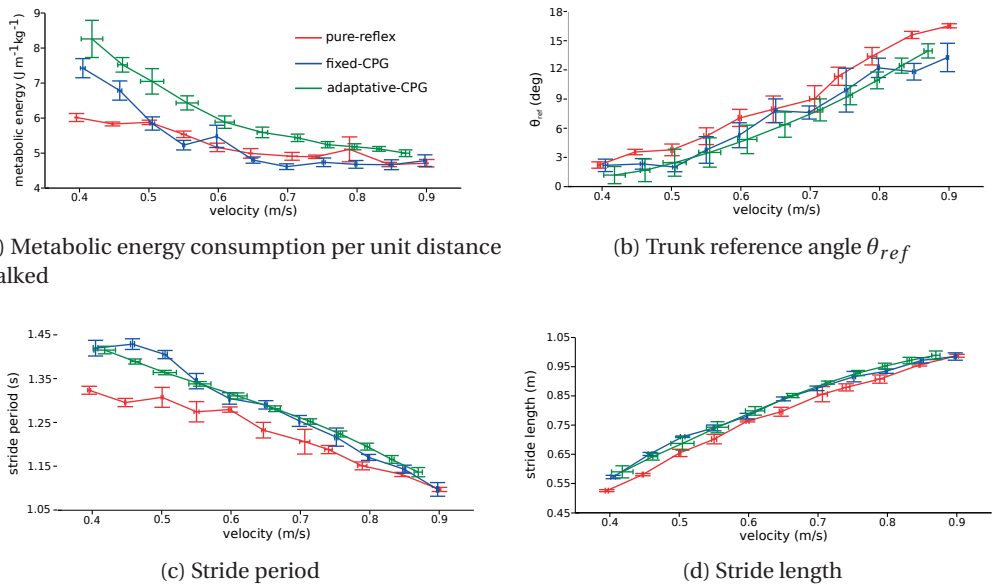
slope is much higher for $k_{HAM,1}$ than for k_{GLU}. This suggests that modulating k_{GLU} is not necessary to achieve gait modulation because its effect after foot strike is largely dominated by the HAM muscles. Consequently, k_{GLU} is finally kept constant for all speeds, see Table E.1 in Appendix E.2. During the swing phase, hip flexion increases for higher speeds. So, the HFL muscles get higher stimulations (with k_{HFL} increasing) while their antagonist muscles HAM get lower stimulations (with $k_{HAM,2}$ decreasing). Under 0.45 m/s, we get a stagnation of θ_{ref} and τ and the evolution of the six parameters of interest is less obvious. Over 0.85 m/s, the optimizer did not manage to find appropriate and robust solutions. However, co-optimizing all parameters according to the strategy described in Section 6.4 (i.e. simultaneous optimization of all parameters) could increase this range from 0.4 m/s to 0.9 m/s. These results support our linear speed control rules presented in the equations (E.3) in Appendix E.3.

6.5.2 Gaits comparison

Three main controllers are compared in terms of energy efficiency, trunk angle reference, stride period and stride length. These three controllers are (i) the *pure-reflex* model from (Geyer and Herr, 2010), (ii) our CPG-based controller optimized for a single fixed speed called *fixed-CPG* and (iii) our CPG-based controller optimized to adapt the gait to a wide range of speeds called *adaptive-CPG*. All of them were optimized for target speeds ranging from 0.4 m/s to 0.9 m/s with a step of 0.05 m/s. For each target speed, five independent optimizations were performed. Contrary to the two first controllers, the *adaptive-CPG* one was optimized in a single optimization run for the whole range of speeds. These three controllers resulted in human-like gaits for the whole range of tested speeds with leg stretching and heel strikes.

All the results are presented in Figure 6.3. Regarding energy efficiency (Figure 6.3a), the *pure-reflex* controller and the *fixed-CPG* one perform in a similar way for speeds higher or equal to 0.5 m/, with the *fixed-CPG* one being slightly more efficient. However, for speeds under 0.5 m/s, the *pure-reflex* controller is clearly more efficient. The main reason is that the objective function of the *fixed-CPG* also targets a good strike prediction, a constraint that is not taken into account in the *pure-reflex* optimizations. Finally, the *adaptive-CPG* model is the less energy-efficient. However, given that this model is optimized for a wide range of speeds in a single shot and not tuned for a precise gait, the small increase regarding energy-efficiency seems a reasonable price to pay.

In terms of stride analysis (Figure 6.3c and 6.3d), the *fixed-CPG* and the *adaptive-CPG* controllers have similar stride periods and lengths. The *pure-reflex* model, however, features lower stride periods and lengths, so favoring smaller steps with a faster frequency. Finally, another difference in gait analysis being visible in Figure 6.3b is that the *pure-reflex* model also favors larger trunk tilt θ_{ref}.



(a) Metabolic energy consumption per unit distance walked

(b) Trunk reference angle θ_{ref}

(c) Stride period

(d) Stride length

Figure 6.3: We run five optimizations for each target speed with different controllers. For each set of five optimizations, the mean and the standard deviation are depicted. For graph legibility, errorbars correspond to the half of the standard deviation.

6.5.3 Target speed tracking

We now focus only on the *adaptive-CPG* controllers where a wide range of speeds is optimized in a single optimization. Figure 6.4 presents snapshots of the COMAN walking with its target speed increasing from 0.4 m/s to 0.9 m/s. After a few steps, it gets backs its initial speed of 0.4 m/s.

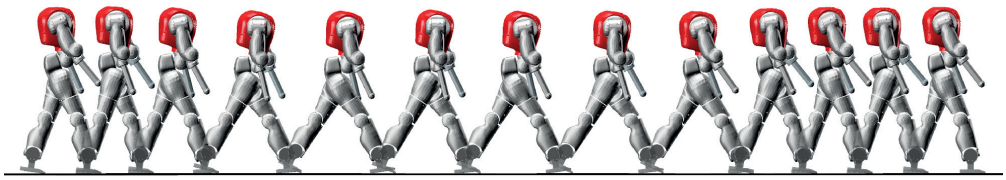


Figure 6.4: COMAN is walking with its target speed increasing from 0.4 m/s to 0.9 m/s before going back to 0.4 m/s. Snapshots of the gait are taken during each double support phase to show the stride length evolution.

Figure 6.5 shows a gait where we modulate the robot speed. The target speed is modified in the range from 0.4 m/s to 0.9 m/s with constant accelerations of $\pm 0.25\, m/s^2$. We measure the speed and post-process it with a 0.5 s running average, visible in green in Figure 6.5. We observe that the robot is able to accelerate from 0.4 m/s to 0.9 m/s in less than 2.3 s (acceleration of $0.22\, m/s^2$), so corresponding to less than two strides. In comparison, (Song and Geyer, 2012) requires four strides for similar accelerations while (Dzeladini et al., 2014) targets a range of speeds nearly two times smaller once scaled to the robot height. Finally,

decelerating from 0.9 m/s to 0.4 m/s is also performed in 2.3 s. Higher target accelerations do not result in higher real accelerations and might make the robot fall. During this experiment, the leg sagittal torques never exceed 30 Nm, except for the hip at high speeds (just after strike). However, these short torque peaks (less than 10% of the stride period in the worst case) never exceed the COMAN maximum hip torque of 55 Nm (Tsagarakis et al., 2013). The reference torque signals are thus within the robot actuators capabilities.

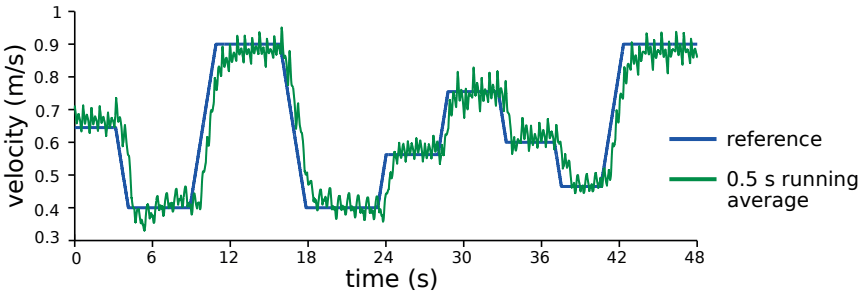


Figure 6.5: The controller can track a speed reference (blue). The robot speed computed in post-process is presented in green.

6.5.4 Stride period prediction

The oscillators network is synchronized with feet strike using short excitation modulations (see Equation (6.3)) when the oscillators prediction is too slow or too fast. During the optimization process, the objective function rewards solutions minimizing these synchronization mechanisms duration, thus when the CPG correctly predicts the step period. So, these oscillators can be used to predict when the next strike will take place. This is potentially relevant to develop other mechanisms requiring synchronization with the walking gait. In Figure 6.6, the robot walks at different speeds from 0.4 m/s to 0.9 m/s. For each speed, we get the stride period predicted with the oscillators structure. Then, we compare this prediction with the actual stride period. The global prediction is rather accurate. Some small differences still appear with slowest speeds: the predicted stride period is slightly higher than the actual one, revealing that the oscillators are too slow. This prediction is also presented in the multimedia attachment.

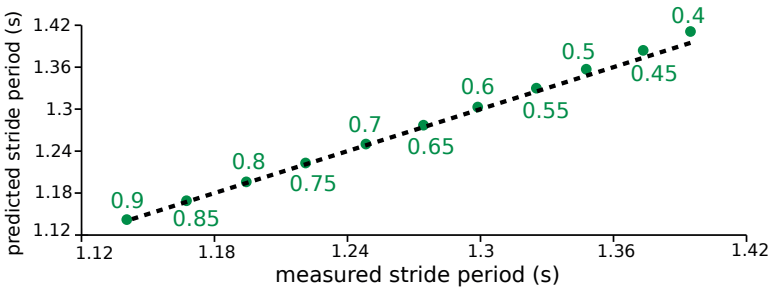


Figure 6.6: The stride period predicted by the CPG is compared to the actual one for different speeds expressed in m/s (green). The dashed line corresponds to correct predictions.

6.5.5 Stepping over a hole

Since our speed modulation algorithm directly impacts the step length, it features another nice and potentially very useful property. Indeed, it can be used to temporarily alter the gait and avoid landing the foot on an undesired place like a hole. An example is provided in Figure 6.7 where a short-time speed target increase can alter the gait to perform a smaller step (likely due to the predominant frequency increase effects during the first step) followed by a longer one to cross a hole. Once this is done, the COMAN recovers its previous gait.

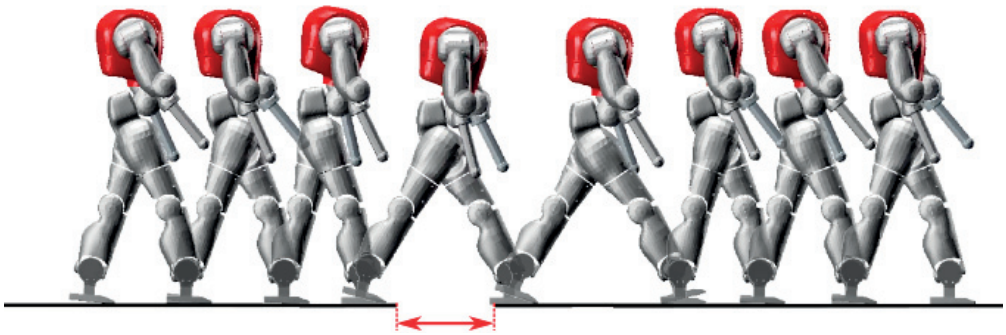


Figure 6.7: COMAN gait is adapted to cross a hole before going back to its previous gait (snapshots taken during each double support phase).

6.6 Conclusion

In this contribution, we presented a bio-inspired controller able to make a humanoid robot walk over a wide range of speeds, allowing fast speed variations during the walking gait. This speed modulation was achieved using simple rules where all parameters to adapt were expressed as linear functions of the target speed. Moreover, this controller, combining reflexes and a CPG in a neuromuscular model, could be tuned in one single optimization. Consequently, reflexes and CPG parameters were co-optimized to achieve a good energy-efficiency over the whole range of speeds. On top of that, the CPG could be used to predict the stride period and to modulate the gait to avoid landing the foot on unwanted locations like holes.

While the main focus of this contribution is to provide efficient walking algorithms for robots with human-like gait features, it might also help to get a better insight on human locomotion, where the existence of CPGs is still a matter open to debate. Our controller relies on Hill-type muscle models controlled by reflexes and Matsuoka oscillators, both being developed on a solid biological background. In this contribution, we demonstrated that simple modulations of the CPG frequency and amplitude, together with a trunk reference angle adaptation, could lead to large gait speeds variations and step modulation. So, like the works of (Taga, 1994) or (Paul et al., 2005), this contribution also argues that CPGs could play a major role in human locomotion, at least to modulate the gait.

However, there is still room for improvement with this controller. In particular, results are deteriorated for slow speeds in terms of energy efficiency, strike prediction and gait modulation, which is worth being investigated. This controller could also be tested on a real platform, as we did for the reflex-based model of (Geyer and Herr, 2010) in Chapter 4. Finally, the natural extensions of this controller is the development of 3D walking gaits, which is addressed in Chapters 9 and 10.

Importantly, the controller developed in this contribution follows the proximo-distal gradient hypothesis (Daley et al., 2007; Dzeladini et al., 2014). In other words, the CPG mainly controls proximal muscles, while distal ones are commanded by reflexes. Similar proximo-distal structures will also be recruited to achieve running gaits (see Chapter 7) and to extend the walking controller to 3D scenarios (see Chapters 9 and 10).

7 Forward speed modulation during 2D running gaits

The material presented in this chapter was obtained during the master thesis of Mr. Bruno Somers (UCL), later extended in the semester project of Mr. Matthew Harding (EPFL).

More precisely, Mr. Bruno Somers managed to obtain running gaits at a single speed (similarly to (Wang et al., 2012)), while Mr. Matthew Harding extended it by including a CPG, resulting in speed adaptation.

I directly supervised these two projects, providing help through guidance, advice and strategy discussions. I also wrote most of the material of this chapter.

In Chapter 6, a controller was presented to modulate the step length and frequency of a walking biped, resulting in its speed adaptation. This speed modulation was performed by adapting a few parameters as functions of the target speed. The leg neuromuscular controller mainly followed a proximo-distal gradient structure, in the sense that the proximal muscles (i.e. close to the hip) were mainly driven by the CPG, while the distal ones (i.e. close to the ankle) were commanded by reflexes.

In this chapter, we target the same purpose (i.e. speed modulation by means of a few parameters modulation), but for running gaits. The proximo-distal gradient hypothesis is maintained, but the stimulations controlling the leg muscles are adapted to the running motion.

Importantly, this contribution adds some flexibility to each foot through the introduction of a new rigid plate connected to the main foot plate by a torsion spring (to simulate the toes). However, this added compliance differs from the prosthetic feet studied in Chapter 5. Indeed, the new feet are made of two interconnected rigid plates, rather than separate parts connected to the body by different springs. Therefore, they are closer to the rigid feet used in most chapters of this thesis.

7.1 Introduction

Reproducing human locomotion in simulation has a variety of applications from aiding prosthetic and rehabilitation medicine to generating stable and human-like robot and animated character movement (Eilenberg et al., 2010; Wang et al., 2012; Van der Noot et al., 2015a). Biologically-inspired neuromuscular controllers are currently gaining attraction regarding speed modulation within walking or running (Song and Geyer, 2012; Dzeladini et al., 2014; Van der Noot et al., 2015b; Song and Geyer, 2015b).

Interestingly, human locomotion is far from being understood. In particular, the recruitment of central pattern generators (CPGs) during human walking or running is still a matter open to discussion (Dimitrijevic et al., 1998). CPGs are neural circuits capable of producing rhythmic patterns of neural activity without receiving rhythmic inputs. These oscillators feature valuable properties among which the possibility to modulate locomotion with simple low-dimensional control signals (Ijspeert, 2008).

Computational models could achieve successful locomotions, both with and without recruiting CPGs. For instance, the work of (Geyer and Herr, 2010), lather extended in (Song and Geyer, 2015b) achieved locomotion modulation on a simplified model of a human, only by recruiting reflexes (i.e. sensory feedback signals). In contrast, (Taga, 1994) could adapt the locomotion of a biped model on uneven terrains using CPG modulation, while (Paul et al., 2005) developed a neuromuscular model recruiting a CPG as central element, in order to investigate the effects of a spinal cord injury on locomotor abilities.

In (Van der Noot et al., 2015b), we developed a neuromuscular walking controller recruiting both reflexes and a CPG to drive the locomotion of COMAN, a 95 cm tall humanoid robot. This is coherent with Kuo's framework, suggesting to combine feedback (i.e. reflexes) and feed-forward (i.e. CPG) pathways in the control of a periodic task (Kuo, 2002). Using this approach, the modulation of a few high-level parameters (as linear functions of the target speed) resulted in the continuous adaptation of the walking speed of the COMAN platform, from 0.4 to 0.9 m/s. This range is similar to the normal human one, once scaled to the robot height. These results were obtained in a 2D simulation environment (i.e. with the waist constrained to stay in the sagittal plane).

In (Song and Geyer, 2015b), running gaits could be obtained on a 2D model of an adult human, resulting in speed adaptation from 2.4 to 4 m/s. This was obtained without CPG (i.e. only using reflexes). Motivated by our previous work on walking locomotion (Van der Noot et al., 2015b), we present a mixed neuromuscular controller combining CPG and reflexes to develop running gaits on a robotic device. In particular, velocity tracking is achieved by modulating a few parameters as linear or quadratic functions of the target speed.

This chapter is divided as follows. In Section 7.2, the COMAN platform is presented in its simulation environment. The neuromuscular controller is developed in Section 7.3, in order to achieve running gaits at a single speed. This controller is later incremented in Section 7.4 to

develop the velocity tracking policy, before analyzing the resulting gait features in Section 7.5. Finally, Section 7.6 concludes the chapter.

7.2 COMAN model

The embodiment being used in this contribution is the COMAN platform, a 23 degrees of freedom (DOFs) full-body humanoid robot. This 95 cm tall robot, weighting 31 kg, was developed at the Italian Institute of Technology (IIT) (Dallali et al., 2013; Tsagarakis et al., 2013), and is pictured in Figure 7.1a.

COMAN features both series elastic actuators (SEA) (Tsagarakis et al., 2009) (mainly for sagittal joints) and traditional, stiff actuators (for the other joints). The torque tracking is then mainly achieved with a PI controller, as presented in (Mosadeghzad et al., 2012). Regarding the robot sensors, each joint features position encoders and custom-made torque sensors. On top of that, an inertial measurement unit (IMU) is attached to the robot waist, while force/torque sensors at the ankle level measure the ground interaction forces and torques.

The results presented in this contribution were obtained using the Robotran simulation software (Samin and Fisette, 2003; Docquier et al., 2013), a symbolic environment for multi-body systems developed within the Université catholique de Louvain (UCL). The ground contact model (GCM) was implemented as detailed in (Geyer and Herr, 2010), while the actuators dynamics were modeled as explained in (Zobova et al., 2017). In particular, a uniform noise with a maximal amplitude of $0.4\,Nm$ was added to the torque measurement, in order to reproduce the one measured with the real platform. Therefore, the resulting torques differ from their references, as would happen on the real platform. The integration was performed using a Runge-Kutta integration scheme with a $125\,\mu s$ time step. More information about the robot and its simulator is provided in (Zobova et al., 2017).

The feet model highly impacts the resulting motion. While many biped robots recruit rigid feet, this is not optimal and differs from the extraordinarily rich bio-mechanical design of the human feet (Colasanto et al., 2015). This is particularly important to initiate the flying phase of running gaits through proper foot push-off. Therefore, a passive compliance was added to the feet model. More precisely, each foot is composed of two rigid plates (of respectively 105 and 35 mm, see Figure 7.1a), connected by a torsion spring. The passive compliance of this spring is modeled as $\tau_{toe} = -k_{toe}\varphi_{toe} - d_{toe}\dot{\varphi_{toe}}$, where τ_{toe} is the resulting torque, φ_{toe} the toe joint angle and $\dot{\varphi_{toe}}$ its time derivative. The torsion spring stiffness k_{toe} is set to $30\,Nm/rad$ and its damping coefficient d_{toe} is set to $1\,Nms/rad$.

7.3 Neuromuscular controller

All the leg and arm sagittal joints are controlled to track torques references, while the remaining joints are maintained to their static home position. The purpose of the controller is therefore to

produce torque references for these leg and arm sagittal joints. These references are obtained using virtual muscles commanded by the combined action of reflexes (feedback signals) and a CPG (feed-forward signals).

7.3.1 Musculo-skeletal model

Similarly to (Geyer and Herr, 2010), the locomotion is achieved by recruiting virtual muscles generating torques at the joint level. In short, each group of muscles is represented by a set of equations called the Hill muscle model (Hill, 1938). When excited, these muscle react by contracting and applying forces on the body. Then, the equivalent torques applied at the joint level can be deduced from the lever arm joining the muscle virtual attachment points to the joints. These torques are sent as torque references, in turn producing voltages at the motor level using the PI controller described in (Mosadeghzad et al., 2012).

In our case, eight Hill-type muscles are recruited for each leg, and four muscles actuate each arm. They are depicted in Figure 7.1a. Their characteristics are summarized in Appendix F.2, while their implementation can be found in (Geyer et al., 2003) and (Geyer and Herr, 2010).

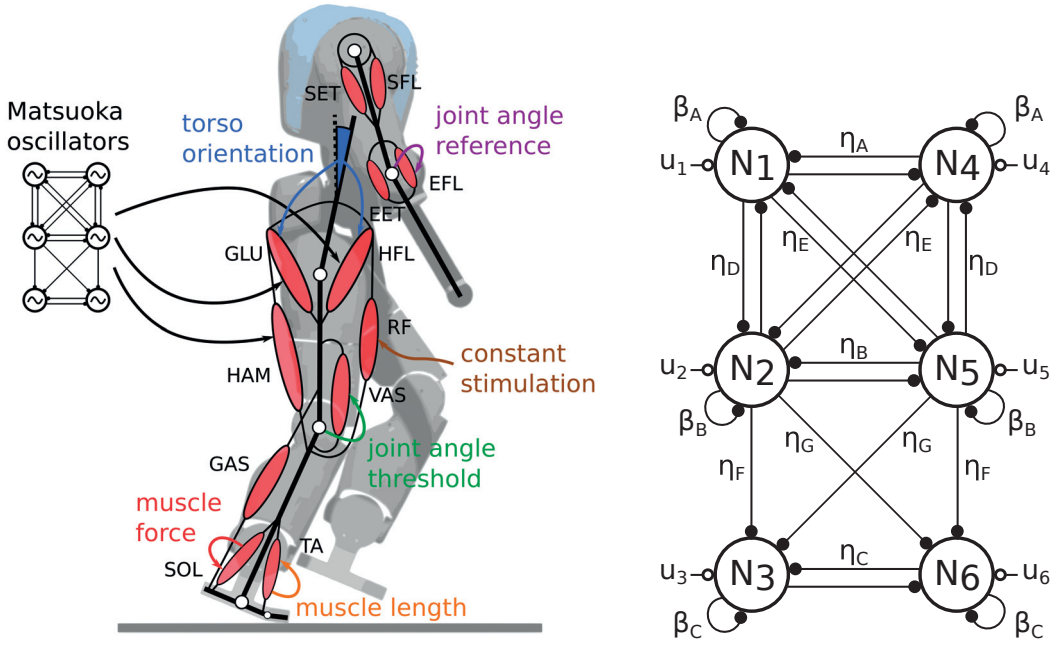
Each muscle is controlled by its activation A_m, capturing the neural signal provided by motoneurons. This signal is related to a neural input S_m, the muscle stimulation, using a first-order low-pass filter capturing the excitation-contraction coupling (Geyer and Herr, 2010). Controlling each muscle thus reduces to design appropriate control rules for these neural stimulations S_m.

7.3.2 Central pattern generator design

In this contribution, a central pattern generator (CPG) network is recruited to drive the proximal muscles, i.e. the muscles acting at the hip level. This is coherent with the proximo-distal gradient hypothesis, postulating that distal muscles (i.e. close to the ankle), being more impacted by external perturbations, are mainly controlled by reflexes, while proximal ones are mainly driven by CPG signals (Daley et al., 2007; Dzeladini et al., 2014).

Here, the CPG network affects the hip flexors (HFL), gluteus (GLU) and biarticular hamstring muscle groups (HAM), i.e. the muscles having a large impact on the hip joint. The other leg muscles, i.e. the soleus (SOL), tibialis anterior (TA), gastrocnemius (GAS), vasti (VAS) and rectus femoris muscle groups (RF) are only commanded with reflexes. These virtual muscles are depicted in Figure 7.1a.

The CPG structure is designed as a six-neurons network of Mastusoka oscillators (Matsuoka, 1985, 1987). These bio-inspired neurons structures capture the mutual inhibitions between half-centers located in the spinal chord and are widely used to model the firing rate of mutually inhibiting neurons, in both the upper and the lower extremities (Ronsse et al., 2009).



(a) COMAN stimulations

(b) CPG central unit

Figure 7.1: To actuate the leg and arm sagittal joints, the controller recruits 12 different Hill-type muscle models, visible in panel (a). These muscles are commanded by a combination of reflex signals and the CPG central unit descending signals (Matsuoka oscillators). Some examples of these stimulation signals are presented in panel (a). In panel (b), the full CPG network is presented. The inter-neuron excitations are indicated with an empty circle, while plain circles represent inhibitions. The corresponding CPG equations are provided in Appendix F.1.

In this network, each neuron N_i state is represented by its firing rate x_i whose evolution with time is governed by Equation (7.1), where v_i is the self-inhibition modulated by an adaptation constant β_j, u_i the external excitation and τ is a time constant. The connexion strengths η_k tune the mutual inhibitions (with $[\bullet]^+ = max(0, \bullet)$). Figure 7.1b depicts the Matsuoka network developed in this contribution. Its six neurons N_i driving the virtual muscles are depicted, along with the parameters β_j, u_i and η_k.

$$\dot{x_i} = \frac{1}{\tau}(-x_i - \beta_j v_i - \sum_{1}^{3} \eta_k [x_l]^+ + u_i) \tag{7.1}$$

The self-inhibition v_i dynamics are captured by Equation (7.2), whose time constant is related to the one of Equation (7.1) through the adimensional parameter γ_j. In Equations (7.1) and (7.2), the index i corresponds to the neuron index, while the indexes k and l are replaced according to Figure 7.1b. Index j equals A for neurons N_1 and N_4, B for N_2 and N_5 and C for

N_3 and N_6. The full development of this network is detailed in Appendix F.1.

$$\dot{v}_i = \frac{1}{\gamma_j \tau}(-v_i + [x_i]^+) \tag{7.2}$$

Interestingly, CPGs present attractive properties like distributed control, redundancies handling, and locomotion modulation using simple control signals (Ijspeert, 2008). In particular, they provide frequency and phasing signals, if properly synchronized with the locomotion gait. This is potentially relevant to activate reflex signals at the right moment or to send feed-forward signals coming from the CPG to the virtual muscles.

The full CPG network (see Figure 7.1b) is composed of four fully connected neurons called the *primary oscillators* (N_1, N_2, N_4 and N_5), and two extra neurons called the *secondary oscillators* (N_3 and N_6). The secondary oscillators are driven by the primary ones but do not impact them.

More precisely, the primary neurons fulfill two main functions: triggering the proximal muscle reflexes at the right timing and sending descending stimulations to the HFL muscles. In particular, neuron N_2 is supposed to fire just after the left strike, while N_5 is supposed to fire just after the right strike. This synchronization mechanism is achieved by modulating the CPG excitations, as described in Appendix F.3. The secondary neurons N_3 and N_6 are mainly aligned with neurons N_1 and N_4, although their firing rates differ, due to the different parameters β_j, γ_j and η_k being recruited. These secondary neurons are in charge of stimulating the HAM muscle before strike impact.

7.3.3 Muscle stimulations

The primary neurons firing rates are used to segment the running gait into four distinct phases for each leg, requiring different computation rules for the muscle stimulations. This is done by detecting when the corresponding firing rates x_i are positive.

In the following rules, most reflexes are adapted from (Geyer and Herr, 2010). The neural signal delays introduced in that contribution are also implemented here. Regarding feed-forward control through CPG signals, the stimulations contributions are computed proportionally to the CPG outputs, defined as $y_i = [x_i]^+$. At then end, all stimulations are bounded in the $[0.01; 1]$ interval.

The first phase is triggered when x_5 is positive for the right leg, or when x_2 is positive for the left leg. This corresponds to the phase directly following the corresponding leg strike impact. During that phase, the torso orientation angle θ_t is maintained to a given reference θ_{ref}, using the feedback rules detailed in Equation (7.3) (with $[\bullet]^- = -min(\bullet, 0)$), where $\dot{\theta}_t$ is the derivative of θ_t and the $S_{0,st}$, k_p and k_d parameters require proper tuning. In particular, the

$S_{0,st}$ parameters represent positive constant prestimulations acting during the stance phase (similarly with $S_{0,sw}$ during the swing phase, see later).

$$\begin{aligned} S_{HFL} &= S_{0,HFL,st} + [k_{p,HFL}\,(\theta_t - \theta_{ref}) + k_{d,HFL}\,\dot{\theta}_t]^- \\ S_{GLU} &= S_{0,GLU,st} + [k_{p,GLU}\,(\theta_t - \theta_{ref}) + k_{d,GLU}\,\dot{\theta}_t]^+ \\ S_{HAM} &= S_{0,HAM,st} + [k_{p,HAM}\,(\theta_t - \theta_{ref}) + k_{d,HAM}\,\dot{\theta}_t]^+ \end{aligned} \tag{7.3}$$

Then, the next phase mainly covers the swing initiation and the flying phase, happening before the contralateral leg strike impact. This is detected when x_1 is positive for the right leg, or when x_4 is positive for the left leg. During that phase, the hip is propelled with the CPG control of the HFL muscle detailed in (7.4), where $k_{HFL,1}$ and $k_{HFL,2}$ are two gains (R and L stand for right and left). Its antagonist muscles GLU and HAM only receive the minimal prestimulation $S_{MIN} = 0.01$.

$$\begin{aligned} S_{HFL,R} &= k_{HFL,1}\,y_1 + k_{HFL,2}\,y_2 \\ S_{HFL,L} &= k_{HFL,1}\,y_4 + k_{HFL,2}\,y_5 \end{aligned} \tag{7.4}$$

After the other leg strike (being detected when x_2 is positive for the right leg, or when x_5 is positive for the left leg), swing-leg retraction is enforced by using the reflex rules of Equation (7.5). $\tilde{F}$ is the muscle current force, normalized by its maximal force F_{max} (see Table F.1 in Appendix F.2). G_{GLU} and G_{HAM} are two gains. During that phase, the HFL muscle still receives the CPG control detailed in (7.4).

$$\begin{aligned} S_{GLU} &= S_{0,GLU,sw} + G_{GLU}\,\tilde{F}_{GLU} \\ S_{HAM} &= S_{0,HAM,sw} + G_{HAM}\,\tilde{F}_{HAM} \end{aligned} \tag{7.5}$$

In the last phase (preceding the ipsilateral leg strike, and corresponding to x_4 positive for the right leg or x_1 positive for the left leg), the stance preparation is improved by adjusting the hip angle φ_h to a given reference $\varphi_{h,ref}$, as suggested in (Yin et al., 2007) and (Wang et al., 2012). This is done with the reflex rules presented in (7.6), where $\dot{\varphi_h}$ is the time derivative of φ_h and the $k_{p,sp}$ and $k_{d,sp}$ are stance preparation gains. At the same time, the knee flexion is controlled by the CPG, using Equation (7.7).

$$\begin{aligned} S_{HFL} &= S_{0,HFL,sw} + [k_{p,sp,HFL}\,(\varphi_h - \varphi_{h,ref}) + k_{d,sp,HFL}\,\dot{\varphi_h}]^+ \\ S_{GLU} &= S_{0,GLU,sw} + [k_{p,sp,GLU}\,(\varphi_h - \varphi_{h,ref}) + k_{d,sp,GLU}\,\dot{\varphi_h}]^- \end{aligned} \tag{7.6}$$

$$S_{HAM,R} = k_{HAM}\, y_6 \qquad S_{HAM,L} = k_{HAM}\, y_3 \tag{7.7}$$

Knee stretching is mainly achieved using the RF and VAS muscles. Similarly to (Wang et al., 2012), the swing initiation during the stance phase is detected when the signed horizontal distance between the COM and the ankle normalized by the leg length ($\tilde{d}$) is larger than a fixed threshold d_{si}. Similarly, the stance preparation during the swing phase is detected when $\tilde{d} < d_{sp}$, where d_{sp} is another fixed threshold.

During stance phase, the RF muscle only receives a basic prestimulation $S_{0,RF,st}$. At the same time, the VAS muscle is activated with a positive force feedback, which can be inhibited when the knee angle φ_k exceeds a given threshold $\varphi_{k,th,st}$. This is performed with Equation (7.8) where G_{VAS} and $k_{\varphi,k}$ are gains and the function $[\bullet]^-_{(\dot{\varphi}_k<0)}$ is set to $-min(\bullet, 0)$ when the condition $\dot{\varphi}_k < 0$ is met, and to 0 otherwise ($\dot{\varphi}_k$ being the time derivative of φ_k).

$$S_{VAS} = S_{0,VAS,st} + G_{VAS}\,\tilde{F}_{VAS} - k_{\varphi,k}\,[\varphi_k - \varphi_{k,th,st}]^-_{(\dot{\varphi}_k<0)} \tag{7.8}$$

During swing initiation, the efforts are being transfered from the VAS to the RF muscles. This is done by incrementing the RF stimulation by a fixed amount si_{RF} and by decreasing the VAS one by a fixed amount si_{VAS}.

The early swing phase keeps the RF and VAS muscles nearly silent by sending fixed prestimulations $S_{0,RF,sw}$ to RF and $S_{0,VAS,sw}$ to VAS. However, the knee joint angle φ_k is expected to reach a given threshold $\varphi_{k,th,sw}$ before the next strike. This is done with the VAS muscle during the stance prepartation, by using Equation (7.9), where $k_{p,sp,VAS}$ and $k_{d,sp,VAS}$ are stance preparation gains.

$$S_{VAS} = S_{0,VAS,sw} + [k_{p,sp,VAS}\,(\varphi_k - \varphi_{k,th,sw}) + k_{d,sp,VAS}\,\dot{\varphi}_k]^+ \tag{7.9}$$

Finally, the control of the remaining leg muscles (i.e. SOL, TA and GAS, acting on the ankle) is done similarly as in (Geyer and Herr, 2010). More precisely, force and length feedback are used using (7.10) during the stance phase and (7.11) during the swing phase. $\tilde{l}_{ce}$ is the length of the contractile part of the muscle, normalized by its optimal length l_{opt} (see Table F.1 in

Appendix F.2). The parameters S_0, G and l_{off} require proper tuning.

$$\begin{aligned} S_{SOL} &= S_{0,SOL,st} + G_{SOL}\,\tilde{F}_{SOL} \\ S_{TA} &= S_{0,TA,st} - G_{S-T}\,\tilde{F}_{SOL} + G_{TA,st}\,[\tilde{l}_{ce,TA} - l_{off,TA,st}]^{+} \\ S_{GAS} &= S_{0,GAS,st} + G_{GAS}\,\tilde{F}_{GAS} \end{aligned} \tag{7.10}$$

$$\begin{aligned} S_{SOL} &= S_{0,SOL,sw} \\ S_{TA} &= S_{0,TA,sw} + G_{TA,sw}\,[\tilde{l}_{ce,TA} - l_{off,TA,sw}]^{+} \\ S_{GAS} &= S_{0,GAS,sw} \end{aligned} \tag{7.11}$$

Regarding the upper-body, the arms swinging motion is controlled by the shoulder flexion (SFL) and extension (SET) muscles at the shoulder level and by the elbow extension (EET) and flexion (EFL) muscles for the elbow (see Figure 7.1a). All these muscles rely on the following feedback rules tracking desired joint positions:

$$\begin{aligned} S_{SFL} &= k_{arm}\,[\varphi_{sh,ref} - \varphi_{sh}]^{-} \qquad & S_{EFL} &= k_{arm}\,[\varphi_{elb,ref} - \varphi_{elb}]^{-} \\ S_{SET} &= k_{arm}\,[\varphi_{sh,ref} - \varphi_{sh}]^{+} \qquad & S_{EET} &= k_{arm}\,[\varphi_{elb,ref} - \varphi_{elb}]^{+} \end{aligned} \tag{7.12}$$

where φ_{sh} and φ_{elb} are respectively the shoulder and elbow joint positions, $\varphi_{sh,ref}$ and $\varphi_{elb,ref}$ are their references and k_{arm} is a constant gain arbitrarily set to 5.

Similarly to (Wang et al., 2012), the arm swing motion is obtained by relating the shoulder target angle position to the difference in the sagittal hip joint positions $\varphi_{hip,R}$ and $\varphi_{hip,L}$, respectively for the right and left legs. This is captured by the following equations:

$$\begin{aligned} \varphi_{sh,ref,R} &= k_{sh}\,(\varphi_{hip,L} - \varphi_{hip,R}) - \Theta_{sh} \\ \varphi_{sh,ref,L} &= k_{sh}\,(\varphi_{hip,R} - \varphi_{hip,L}) - \Theta_{sh} \end{aligned} \tag{7.13}$$

where Θ_{sh} and k_{sh} are both set to 0.3. Finally, $\varphi_{elb,ref}$ is set to a constant reference of $-0.25\,rad$ to keep a constant position for the elbow.

7.3.4 Gait initiation

The controller initialization only requires specific control rules for the initial CPG excitations, as described in Appendix F.3. On top of this, the biped initial dynamics (i.e. initial position and speed) is optimized, so that the simulation starts with both legs in swing (i.e. flying phase). The related parameters (joint initial angles and speeds...) are reported in Table F.3 (see Appendix F.5).

The initial forward position of the floating base (represented by the initial coordinate along the X axis (i.e. forward direction) of the waist x_w) is irrelevant, and is therefore set to 0. However, its time derivative $\dot{x}_w$ highly impacts the initial gait. Letting the optimizer tune this parameter usually results in its maximization, in order to propel the body as far as possible (therefore getting a larger fitness score), even if it directly collapses after the first step. To solve this issue, $\dot{x}_w$ is arbitrarily set to a value of 1.5 m/s, i.e. in the middle of the targeted speed range.

Regarding the arms, their initial positions are set to the initial position references tracked by both the elbows and the shoulders (see Section 7.3.3). The elbow speeds are set to 0, while the right shoulder speed is set to $\dot{\varphi}_{sh}$, and the left shoulder speed is set to $-\dot{\varphi}_{sh}$ (see Table F.3).

7.3.5 Gait controller optimization

Many unknown parameters were introduced in the controller development and in the simulation initial dynamics. They are listed respectively in Tables F.2 and F.3 (see Appendix F.5), together with their respective bounds. These parameters require a proper tuning, in order to optimize the resulting running gaits. Here, this tuning was performed using a particle swarm optimization (PSO) algorithm (Kennedy and Eberhart, 1995).

More precisely, a fitness function was computed as the aggregate of different stages, computed in parallel. The algorithm purpose was then to maximize the global fitness function. More precisely, each set of optimized parameters was tested with the biped running during a maximal time of 70 s. After this duration (or earlier if the robot fell), all stages were computed and their respective contributions were added to the global fitness function. The computation of this global fitness function is fully described in Appendix F.4, while the corresponding optimized parameters (and their bounds) are listed in Appendix F.5.

7.4 Speed adaptation

In this section, the controller is further incremented to achieve speed adaptation through the modulation of a small set of parameters. In (Van der Noot et al., 2015b) (i.e. Chapter 6), three types of key control parameters were identified to adapt the biped speed during walking: the CPG time constant τ, the CPG output gains and the torso reference angle θ_{ref}. Their modulation resulted in continuous speed adaptation in a range from 0.4 to 0.9 m/s.

Similarly to that contribution, the parameters τ, $k_{HFL,1}$, $k_{HFL,2}$, k_{HAM} and θ_{ref} were selected to study their impact on the running speed. The hip target $\varphi_{h,ref}$ has a similar function as θ_{ref}, in the sense that it is also recruited to track a desired position through hip muscles control. This parameter was therefore the last hip muscle parameter studied.

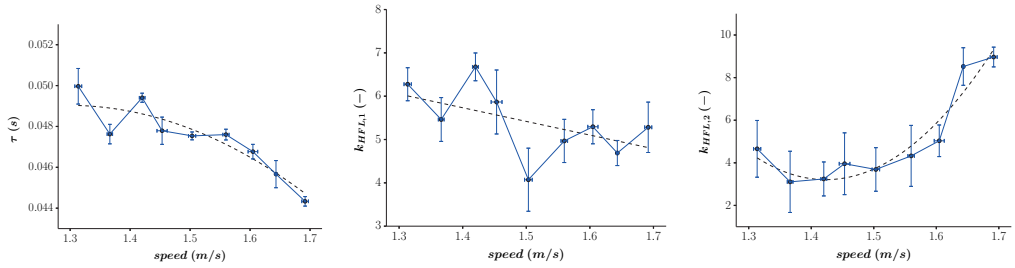
For running gaits, the push-off phase and the related leg stretching are crucial because of their impact on the flying phase. At the knee level, the leg stretching is mainly obtained with the VAS muscle during stance. Therefore, the gains G_{VAS} and $k_{\varphi,k}$ were both studied. The ankle push-off is mainly obtained (also during stance) with the SOL and GAS action, together with the TA inhibition. Thus, parameters G_{SOL}, G_{GAS} and G_{S-T} were the last selected parameters.

To study the impact of these parameters on the running speed, the following experiment was performed. A first optimization was carried out as described in Section 7.3.5, with a target speed of $1.5\,m/s$. The resulting controller was called the *initial controller*. Then, all the optimized parameters were frozen, except the eleven key parameters selected for gait adaptation. New optimizations were then performed, only on these eleven key parameters. More precisely, the controller was kept intact during the first six steps (i.e. same parameters as the initial controller). After the sixth step, the eleven key parameters were replaced with their new optimized values. At the end, the steady-state speed was measured. These reduced optimizations were performed for target speeds from 1.3 to $1.7\,m/s$ ($1.5\,m/s$ included) with a step of $0.05\,m/s$. For each target speed, five similar optimizations were performed.

The results are visible in Figure 7.2, where the evolution of these parameters with the measured speed can be observed. Intuitively, these parameters evolution with speed can be approximated using polynomial functions. In order to select appropriate polynomial orders capturing the essence of the curve without over-fitting, a model goodness-of-fit analysis using the sum of squared values of the prediction errors was used, as described in (Smith and Rose, 1995). In short, for each pair of parameter and polynomial order, the corresponding p-value measures the likeliness that the selected order is appropriate to represent the parameter evolution. These p-values are gathered in Table 7.1, for orders 0, 1 and 2. The order with the highest p-value was then selected. The only exception is the parameter $\varphi_{h,ref}$, whose p-values for orders 0 and 2 are very close. Therefore, we arbitrarily selected the order 0 for this parameter.

The polynomial approximations for the selected orders are depicted with dashed lines in Figure 7.2. The time constant τ decreases for faster speeds, thus favoring higher step frequencies. Interestingly, this decrease is more important for faster speeds, indicating that the robot favors step length adaptation for slow speeds, and step frequency adaptation for faster speeds.

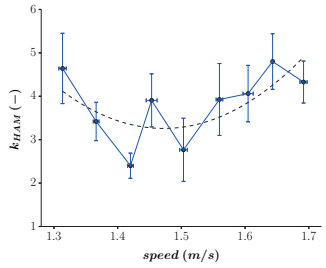
The hip flexor is stimulated by two CPG gains during the swing phase: the swing initiation gain $k_{HFL,1}$ and the mid-swing gain $k_{HFL,2}$. In particular, $k_{HFL,2}$ shows an increasing trend with speed (except for the slowest speeds), resulting in larger hip flexion, and so in longer step lengths for the fastest speeds. In contrast, the negative trend of the swing-initiation stimulation gain $k_{HFL,1}$ remains unclear, but its relative variation is small (compared to the one of $k_{HFL,2}$), and should thus not significantly impact the gait.



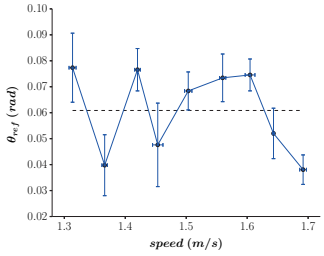
(a) CPG time constant τ

(b) CPG gain $k_{HFL,1}$

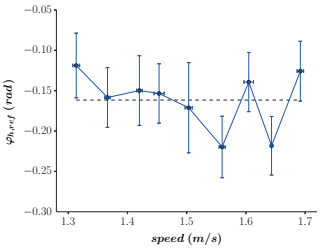
(c) CPG gain $k_{HFL,2}$



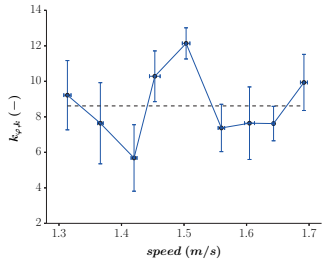
(d) CPG gain k_{HAM}



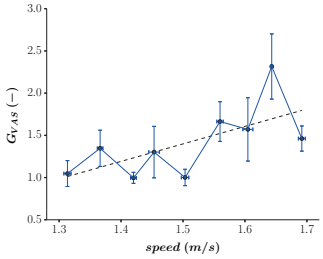
(e) Torso orientation reference θ_{ref}



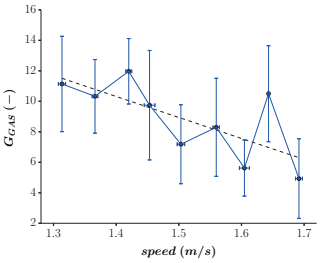
(f) Hip angle reference $\varphi_{h,ref}$



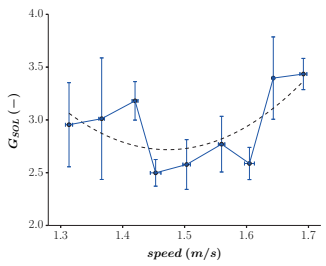
(g) Knee angle gain $k_{\varphi,k}$



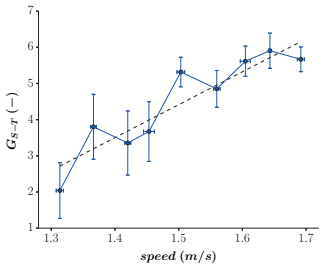
(h) reflex gain G_{VAS}



(i) reflex gain G_{GAS}



(j) reflex gain G_{SOL}



(k) reflex gain G_{S-T}

Figure 7.2: Five optimizations are performed for each target speed (from 1.3 m/s to 1.7 m/s with an interval of 0.05 m/s).The actual speed of each solution is measured, along with the optimized value of the eleven open key parameters. For each target speed, we gather the five optimization final results, reporting their mean and standard deviations. For graph legibility, the error bars represent half of the standard deviations. Dashed lines correspond to the polynomial approximations whose order is computed in Table 7.1, using the minimum mean square error method.

Table 7.1: This table reports the p-values associated with the polynomial approximations of orders 0, 1 and 2 of the data provided in Figure 7.2, based on the least square errors. Each p-value is computed using the sum of squares due to lack of fit analysis (Smith and Rose, 1995). The selected order is presented with grey cells.

	order 0	order 1	order 2	selected
τ	0	0.017	**0.026**	2
$k_{HFL,1}$	0.036	**0.102**	0.093	1
$k_{HFL,2}$	0.002	0.117	**0.825**	2
k_{HAM}	0.146	0.146	**0.356**	2
θ_{ref}	**0.026**	0.022	0.021	0
$\varphi_{h,ref}$	**0.608**	0.577	0.642	0
$k_{\varphi,k}$	**0.239**	0.175	0.117	0
G_{VAS}	0.014	**0.104**	0.071	1
G_{GAS}	0.612	**0.86**	0.781	1
G_{SOL}	0.275	0.247	**0.517**	2
G_{S-T}	0.002	**0.653**	0.647	1

The HAM gain k_{HAM} obtained the largest values for the speed extrema. This increases the knee flexion before stance using the HAM muscle, therefore preventing hyper-extension and enabling greater shock-absorption to occur. At high speed, this is useful to counter the effects of the increased step length. At low speeds, this increase of the k_{HAM} gain helps reducing the biped speed.

No trend is observed for the torso reference angle θ_{ref}, indicating that this value can be kept constant. This might be due to the optimization stage which aimed at maintaining an upright posture (see Appendix F.4). This stage was inspired from (Wang et al., 2012), in order to achieve more robust running gaits. However, (Song and Geyer, 2015b) argued that modulating the torso orientation is a dominant contributor for running speed acceleration and deceleration, such that, intuitively, a forward lean can help to realize faster speeds. Similarly to θ_{ref}, no clear trend is observed for the hip reference angle $\varphi_{h,ref}$.

Regarding knee control during stance, the increase of the G_{VAS} gain favors legs stretching for faster speeds. In contrast, the parameter $k_{\varphi,k}$, in charge of preventing knee overextension, is not significantly affected by the running speed.

Finally, both the SOL and GA muscles act on the ankle to create forward thrust (via plantarflexion through the G_{SOL} and G_{VAS} gains), while the antagonist TA muscle is inhibited using the G_{S-T} gain, to avoid unnecessarily fighting against the SOL and GAS actions. Intuitively, these three parameters were expected to increase with speed, in order to favor larger push-offs for the fastest speeds. This trend is clearly visible for the G_{S-T} gain and for the major part of the G_{SOL} evolution. However, for slow speeds, the opposite trend is observed for G_{SOL}. Interestingly, the GAS muscle parameter displays a negative trend, thus favoring larger thrust for slow speeds. A possible explanation is that a significant part of the ankle propulsion efforts is transfered from the SOL to the GAS muscle for slow speeds.

Based on these results, new optimizations can be performed, in order to co-optimize all the controller parameters for the whole range of speeds. The strategy is the following. Among the elven key parameters studied, eight presented linear or quadratic evolution trends with speed: τ, $k_{HFL,1}$, $k_{HFL,2}$, k_{HAM}, G_{VAS}, G_{SOL}, G_{GAS}, G_{S-T} (see Table 7.1). These parameters were replaced by their corresponding linear or quadratic functions of the target speed, as presented in Appendix F.5. All the other parameters were optimized as constant values (see Tables F.2 and F.3). Then, each set of optimized parameters was evaluated successively with different target speeds, according to the requested speed range. The fitness function was finally obtained as the average of each fitness function computed for each run, as detailed in Appendix F.4.

7.5 Results

The results presented in this section were obtained after a single optimization on the range of speeds from 1.25 to 1.7 m/s, i.e. on a range similar to the one of Figure 7.2, but slightly extended towards the lowest speeds.

The resulting gait is first presented during a speed tracking experiment. The gait main features and their evolution with speed are then analyzed.

7.5.1 Speed tracking

First, the robot forward speed reference v_{ref} was modulated by an operator in the $[1.25; 1.7]$ m/s range, during 70 s. Its evolution is displayed in Figure 7.3. The modulation of this parameter directly impacted the eight selected key parameters, according to the rules summarized in Table F.2 (see Appendix F.5). The evolution of the robot actual speed is also displayed in Figure 7.3. In Figure 7.4, snapshots of this experiment are presented when COMAN was running at a speed close to the middle of the speed range (1.42 m/s).

Globally, the lowest commands (i.e. $v_{ref} = 1.25\,m/s$) resulted in stable running. However, the actual (measured) speed was a bit higher and stayed close to 1.3 m/s. For the fastest speed references (i.e. v_{ref} close to 1.7 m/s), the gait was generally less stable, usually resulting in the biped fall in the forward direction. However, it is possible to reach these fastest speeds during a short time period, before reducing v_{ref} to stabilize the gait. This behavior was obtained in this experiment, as can be seen in Figure 7.3, especially during the $[6; 7]$ s and the $[44; 48]$ s time intervals. For speed references over 1.55 m/s, the gait actual speed started oscillating, as can be see in Figure 7.3 during the $[25; 30]$ s and the $[44; 48]$ s time intervals. Finally, speed accelerations and decelerations were both around 0.145 m/s^2 (in absolute values).

The snapshots of Figure 7.4 illustrate the running gait motion. In particular, feet strikes are visible in panels (a), (f) and (k), push-off phases in panels (c) and (h) and flying phases in panels (e) and (j).

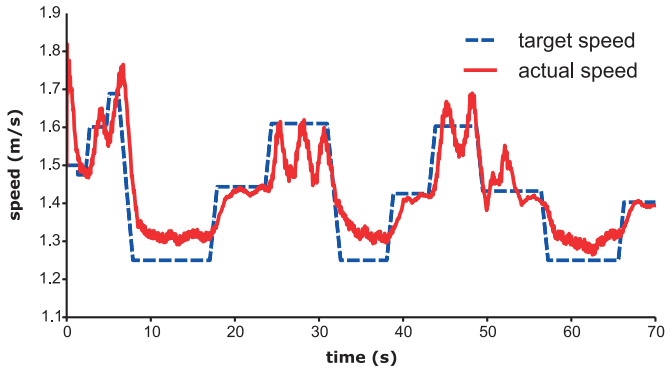
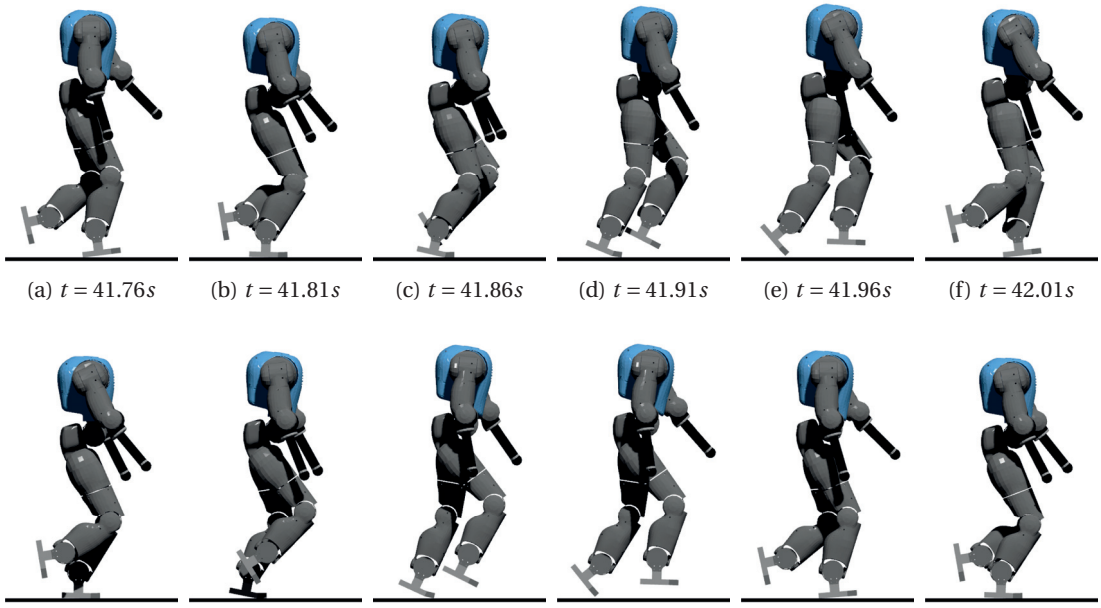


Figure 7.3: In this experiment, the target speed v_{ref} is indicated with the dashed line, while the robot actual forward speed is indicated with the solid line. This actual speed is post-processed with a running average of 1 s.



(a) $t = 41.76s$ (b) $t = 41.81s$ (c) $t = 41.86s$ (d) $t = 41.91s$ (e) $t = 41.96s$ (f) $t = 42.01s$

(g) $t = 42.06s$ (h) $t = 42.11s$ (i) $t = 42.16s$ (j) $t = 42.21s$ (k) $t = 42.26s$ (l) $t = 42.31s$

Figure 7.4: Snapshots of the COMAN robot during the experiment displayed in Figure 7.3. The times indicated are coherent with the abscissa of Figure 7.3 and correspond to an average speed of 1.42 m/s. The first snapshot was taken at a right foot strike, while the next ones were taken with a 50 ms time interval.

7.5.2 Gait features

Using the same controller, the following gait features were studied: stride frequency and length, flying phase ratio and metabolic energy consumption (evaluated as presented in (Bhargava et al., 2004), and normaliyed to the biped mass and to the traveled distance). This was studied for speed references v_{ref} evolving in the $[1.25; 1.6]\, m/s$ range. Indeed, as mentioned in Section 7.5.1, faster speeds could be reached during short periods, while we were interested here in measuring steady-state values.

The results are presented in Figure 7.5. Firstly, it can be seen that there is a gap between the measured features of most tested speed references and the last one (i.e. $v_{ref} = 1.6\, m/s$). As previously mentioned, this is due to the oscillations in the measured speed observed for speed references over $1.55\, m/s$.

The stride frequency and length are both shown to increase proportionally to speed (see Figures 7.5a and 7.5b). This matches the intuition that human running prefers fewer steps and higher frequencies to run distances in a shorter amount of time. However, the ranges of variations of the stride frequency and length are both around 9 % of their maximal values, while it was observed that the increase of speeds in real human running was mainly related to stride length adaptation, i.e. with few variations of the stride frequency (Cavanagh and Kram, 1989).

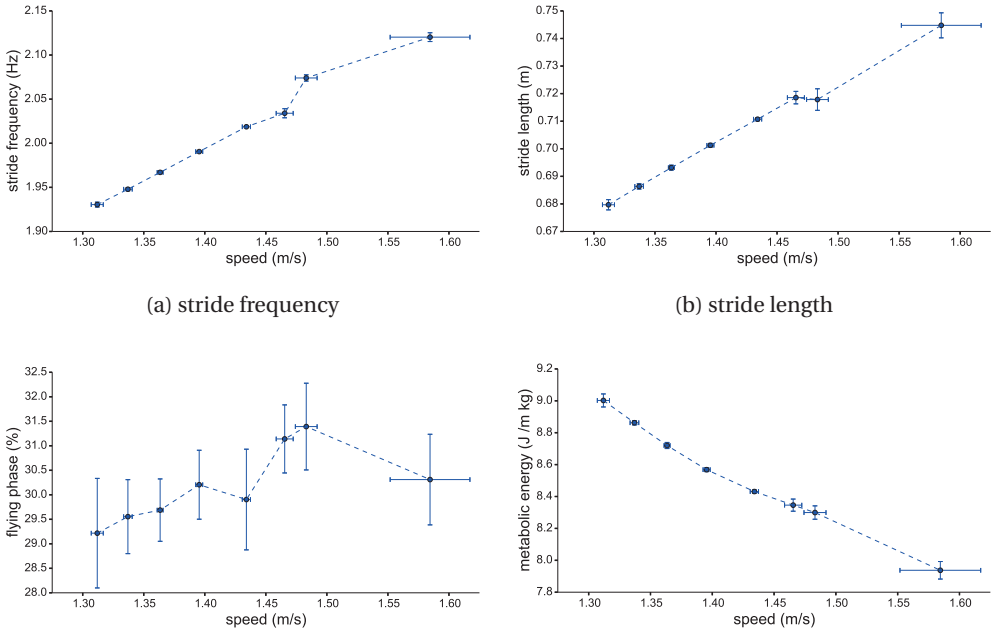
Regarding the flying phase ratio, a slight increase is observed for faster speed references (except for $v_{ref} = 1.6\, m/s$). This speed evolution might suggest that the optimizer favored larger feet propulsion for faster speeds, therefore resulting in longer flying phases. However, this flying phase ratio appears to remain relatively close to its average value of 30 %, so that this trend is not significant.

Finally, the gait-cycle metabolic energy cost (normalized to the traveled distance) decreases as the speed increases. This is mainly due to the larger distances traveled in a given amount of time for faster speeds.

7.6 Conclusion

In this contribution, we developed a neuromuscular controller mixing both reflexes and CPG signals to control the speed of a running biped. The modulation of eight key control parameters as linear or quadratic functions of the target speed resulted in the modulation of the biped forward speed. The selection of these key parameters was done by studying the evolution of these parameters with the target speed.

In steady-state, speeds could be continuously commanded in a range from 1.3 to $1.6\, m/s$. During short periods, the speed could be increased up to $1.7\, m/s$. In contrast, the speed adaptation of the reflex-based controller developed in (Song and Geyer, 2015b) could reach speeds ranging from 2.4 to $4\, m/s$, on an adult model. Once scaled to the size of COMAN, this



(a) stride frequency

(b) stride length

(c) flying phase ratio

(d) metabolic energy consumption

Figure 7.5: Panel (a) presents the stride (i.e. two steps) frequency, while panel (b) shows the stride length. Panel (c) displays the flying phase ratio. In panel (d), the muscles metabolic energy consumption per unit distance is computed (only for the lower-limb muscles acting in the sagittal plane, the arm muscles energy being negligible in comparison). For each speed reference v_{ref} (from 1.25 m/s to 1.6 m/s, using a discretization of 0.05 m/s), ten simulation runs are performed. For each speed, the means and the standard deviations (from the ten corresponding runs) of the different actual speeds and measures are pictured.

lower speed bound is similar to ours. However, the model of (Song and Geyer, 2015b) could reach faster speeds (once scaled to the biped size). In that contribution, this was done by adapting 64 parameters as linear functions of the target speed.

Therefore, it seems that this increase in the number of parameters being modulated allows to reach larger speed ranges. Our controller could therefore be incremented by adapting a few extra parameters, as function of the forward speed. In particular, (Song and Geyer, 2015b) pointed out that the adaptation of the torso lean angle could play a major role in speed modulation. In our contribution, this parameter appeared to remain quite constant with speed alterations. However, this might be related to our fitness function, aiming at maintaining an upright posture to increase stability, as suggested in (Wang et al., 2012).

Another interesting avenue to explore is to use the CPG to trigger the knee special reflexes. In the present contribution, these knee special reflexes are triggered based on the center of mass (COM) position. This strategy was adapted from (Wang et al., 2012), where no CPG control was implemented. Further studies could be carried out, in order to trigger these special knee

reflexes based on the CPG firing rates, rather than on the COM position.

Finally, the robustness of the controller could be improved, in particular for the fastest speeds. For instance, in Chapter 5, we showed that the use of prosthetic feet during walking could improve the biped robustness on uneven grounds. Therefore, the same controller could be tested while using these prosthetic feet. Another avenue to explore is presented in Chapter 8, where a method is presented to learn muscle stimulations resisting to unpredicted external perturbations. Because this approach also relies on muscular control, a similar strategy could be investigated for running gaits to improve the biped robustness (for instance, based on the COM desired position, see Chapter 8).

8 Bio-inspired balance controller

Publication

The material presented in this chapter is adapted from:

> *Heremans F, Van der Noot N, Ijspeert AJ and Ronsse R (2016) Bio-inspired balance controller for a humanoid robot. In: 2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob), Singapore, 26-29 June 2016, pp. 441-448. DOI: 10.1109/BIOROB.2016.7523667.*

The material presented in this contribution was obtained during the master thesis conducted by Mr. François Heremans (UCL). I directly supervised this work, providing help through guidance, advice and strategy discussions, throughout the whole project. I also provided the basic code, and helped during paper writing for revisions.

Robustness is a critical issue to bring humanoid robots out of the laboratories. As presented in Chapters 4 and 5, there is an intrinsic stability in the neuromuscular controllers to resist to small perturbations.

In this chapter, we are interested in learning strategies to withstand larger perturbations. This is studied during straight postures, in 3D scenarios (i.e. no lateral constraint applied on the waist, unlike previous chapters). In particular, strategies are developed to automatically learn appropriate muscle stimulations, based on the center of mass (COM) deviation.

As discussed at the end of this chapter, the proposed strategy could possibly be adapted to walking or running gaits, in order to increase the biped robustness. However, another application of the proposed methodology is to control the posture of the robot to initiate walking in 3D scenarios, by controlling the COM reference position.

In the previous chapters (except Chapter 7 targeting running gaits), the biped gait was initiated by standing upright, while leaning a bit forward through proper ankle control. The swing of the first leg was then activated from this posture. This strategy does not work for the 3D

scenarios targeted in Chapters 9 and 10. Indeed, the walk initiation developed in these next chapters requires to deport the COM on top on one foot, before initiating the swing of the other leg. This COM initial control is addressed in this chapter.

8.1 Introduction

Humanoid robotics has gained much interest during the last decades. This interest for human-like robots is partly driven by a strong incentive: evolving in environments made for humans is challenging for a robot due to the numerous artifacts that were specifically designed for us (stairs, doors, levers, etc.). The quest for humanoid robots that can move and act in those complex environments is therefore partly motivated by scenarios where these robots could replace humans. This is particularly relevant in hazardous environments such as man-made or natural disaster scenarios.

However, the world outside the lab is an uncontrolled environment for legged humanoid robots, with unexpected ground levels, potential collisions and other external perturbations. Coping with these unexpected perturbations requires a robust balance controller. Despite major advances in the fields of robotics and automation, robots are still far from reaching the superb ability of humans to balance in challenging situations. Taking inspiration from the human neuromechanical apparatus thus represents a promising research avenue to bridge this gap. This idea has already been applied for dynamic walking in simulation. For instance, a lower-limbs musculo-skeletal model comprising virtual muscles was developed in (Geyer and Herr, 2010). Still in simulation, this model was further incremented in (Van der Noot et al., 2015b) by the addition of a bio-inspired oscillatory neural controller modulating the forward speed. Nevertheless, the simulation/reality gap represents a challenge for traditional controllers. Bio-inspired controllers can help in reducing this gap by providing natural compliance to cope with the world non-idealities. For instance, (Van der Noot et al., 2015a) shows an almost straightforward transfer of the controller in simulation to the real humanoid robot.

Alternatively, legged robots also provide a fantastic tool to improve our understanding of the human neuro-musculo-skeletal apparatus. Indeed, robots offer to emulate isolated components of this apparatus and so to test hypotheses about their behavior. Moreover, robots can serve to simulate pathologies and thus investigate their consequences. Finally, bipedal robots offer the opportunity to generate a large amount of data and thus to study the system sensitivity to a large set of neuromechanical parameters.

Regarding postural control, different balancing strategies were identified from human experiments. In the case of small to medium perturbations, humans can maintain balance using body coordination only, so that no stepping is required (Stephens, 2007). However, the neuro-motor pathways leading to such postural responses are not straightforward. Over the past years, simplified models were elaborated to capture this complexity. For instance, (Albus, 1971) introduced the Cerebellum Model for Articulation Control (CMAC), a simplified cerebellum model for robot control intended to provide motor learning capabilities.

In addition, the recent developments of torque controlled robots equipped with compliant actuators, i.e. series elastic actuators, provide new tools to validate the aforementioned neurological models. These new platforms also enhance the use of compliant controllers, allowing robust interactions with the environment. For example, (Li et al., 2012) uses a passivity-based admittance (compliant) controller for stabilization, making the robot compliant to external steady-state perturbations. Similarly, (Hyon et al., 2007) modulates ground applied forces for balancing. Such compliant controllers are well adapted to situations with dynamic interactions such as balancing or physical contacts with humans.

In this contribution, we focus on perturbations that do not require stepping to recover equilibrium. We merge bio-inspiration and classical control techniques to achieve a novel controller. In a nutshell, the contributions of this chapter are: (i) the implementation of an inverse muscular model, (ii) the development of a neural controller using machine learning techniques to produce a regression model of the muscular stimulations and (iii) the development of a training chain based on an impedance controller. To the best of our knowledge, this contribution is the first to report the use of machine learning techniques to drive a 3D musculo-skeletal model of the legs for balance control. The developed method is validated in simulation. This chapter is structured as follows. In Section 8.2, the control framework is outlined. The neuro-musculo-skeletal model and the learning mechanisms are described. Section 8.3 details the simulation protocol used to validate the proposed controller. Section 8.4 describes the simulation results. Finally, the simulation outcomes are discussed in Section 8.5.

8.2 Postural control framework

This section details the framework providing a full-body compensatory motion to counter 3D perturbations. A neuro-musculo-skeletal chain inspired by the human neuromechanical apparatus is introduced to capture the robot controller (top part of Figure 8.1). This bio-inspired chain is composed of two modules. A neural controller implemented by a regression engine (machine learning) is in charge of generating virtual muscular simulations. Using these stimulations, a 3D musculo-skeletal model generates joint torques applied to the lower limbs. An impedance controller and an inverse muscular model (lower part of Figure 8.1) were further implemented in order to generate the reference stimulations being required for the learning process of the neural controller. The impedance controller also drives upper body movements for which no musculo-skeletal model was developed.

8.2.1 Musculo-skeletal model

A virtual musculo-skeletal system was developed to drive the lower-limbs degrees of freedom (Figure 8.2), as an extension of the musculo-skeletal model developed by (Geyer and Herr, 2010) and (Song and Geyer, 2013).

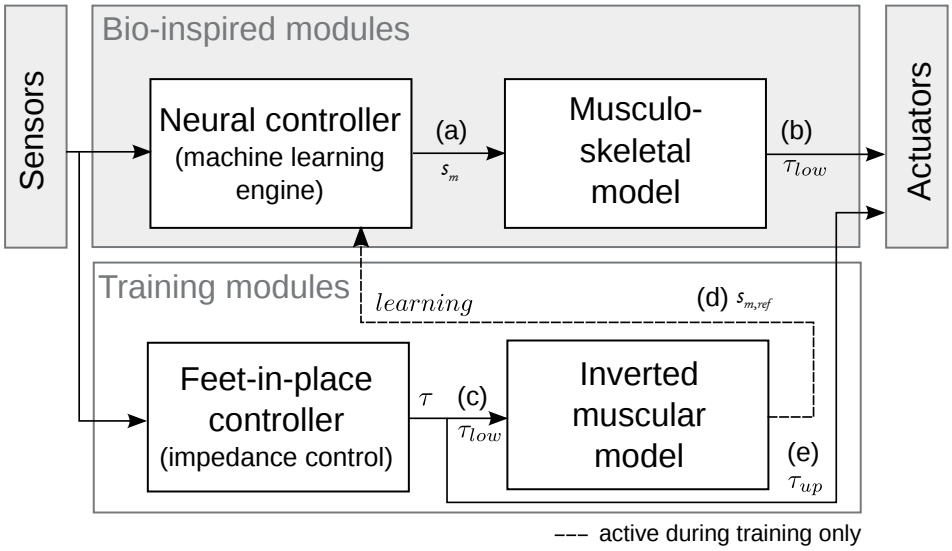


Figure 8.1: Architecture of the main controller. Based on the sensed perturbation, the neural controller generates muscular stimulations (a) to drive a musculo-skeletal model. The resulting joint torques (b) actuate the robot lower limbs. In parallel, an impedance controller computes desired torques (c) being transformed into virtual stimulations (d) by an inverse muscular model. These are used by the neural controller during the learning phase. The impedance controller further computes the upper-limb reaction torques (e), for which no muscular model was developed.

The fundamental building block of the muscular model depicted in Figure 8.2 is the Hill-type muscle model (Figure 8.2(b) and (Hill, 1938)). Each muscle tendon unit (MTU) consists of two main elements: an active contractile one (CE) and a passive series elastic one (SE). On top of that, a parallel-elastic element (PE) and a buffer elasticity element (BE) prevent the muscle from collapsing on itself or overstretching when it leaves its nominal operation range.

The sagittal muscles – soleus (SOL), tibialis anterior (TA), gastrocnemius (GAS), vastus (VAS), hamstring (HAM), gluteus (GLU) and hip flexor (HFL) – were adapted from (Geyer and Herr, 2010) and the hip adduction (HAB)/abduction (HAD) muscles from (Song and Geyer, 2013). The model was further incremented with four extra mono-articular muscles (HER, HIR, EVE, INV) for the hip external/internal rotations and the foot eversion/inversion, respectively. Similarly to (Song and Geyer, 2013), the actuation scheme is simplified by assuming the full decoupling of the sagittal, frontal and transverse planes. This assumption does not capture

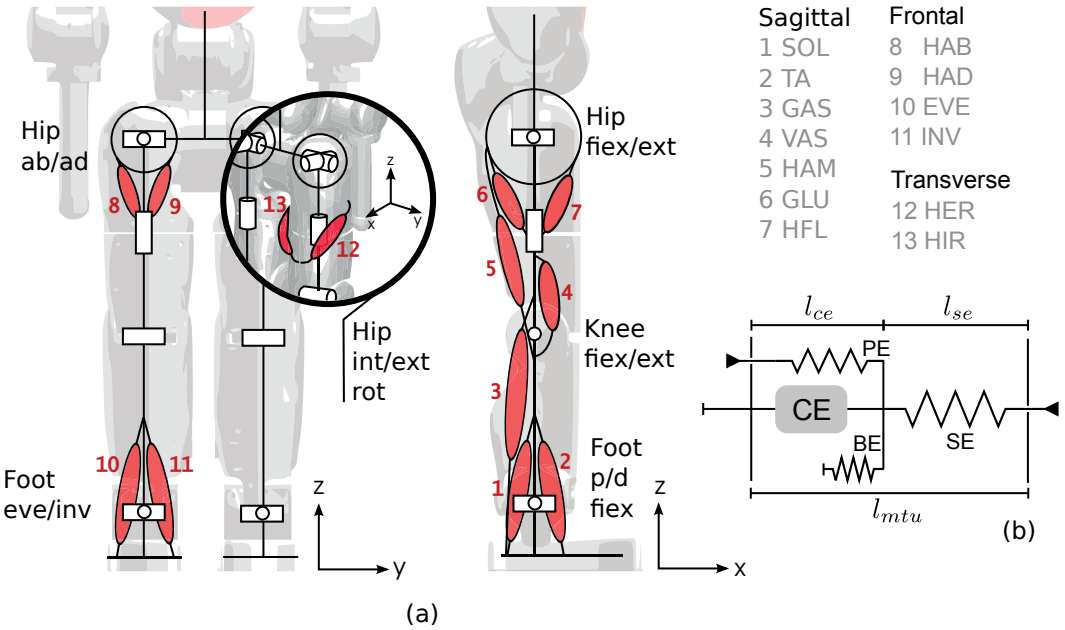


Figure 8.2: (a) 3D lower limbs musculo-skeletal model comprising 13 Hill-type muscles (b) per leg.

the whole complexity of the human motor system but can be relaxed later with no major consequences for our framework. Table 8.1 provides the anatomical parameters of the four extra muscles using the template provided in (Geyer and Herr, 2010) (see (Song and Geyer, 2013) for the other muscles). These muscle parameters were estimated using the lower-limbs model from (Arnold et al., 2010) in OpenSim (Delp et al., 2007). Dynamic scaling was used to tune the muscular parameters to the particular robot being used (Bejan and Marden, 2006; Schepelmann et al., 2012) (see Section 8.3 for the robot description).

Table 8.1: Hip external/internal rotation and foot eversion/inversion muscles parameters

Muscle	HER	HIR	EVE	INV
F_{max} $[N]$	177	283.2	247.8	318.6
v_{max} $[l_{opt}/s]$	18.36	18.36	18.36	18.36
l_{opt} $[cm]$	1.708	3.416	2.135	2.135
l_{slack} $[cm]$	2.135	2.989	10.675	12.81
r_0 $[cm]$	1.708	1.281	1.281	0.854
ϕ_{max} $[deg]$	-	-	-10	5
ϕ_{ref} $[deg]$	10	-20	-5	-10
ρ $[-]$	1	0.7	0.7	0.7

Each virtual muscle is simulated by a direct model that transforms muscle stimulations into muscle forces further transformed into joint torques as a function of their insertion points. At each time step, two operations are repeated: (i) updating each muscle state based on the local skeletal configuration and the muscular stimulations; and (ii) computing the resulting joint

torques based on the muscle forces and skeletal configuration. More precisely, a muscular iteration goes through the following steps (see Figure 8.2(b) and (Geyer and Herr, 2010) for details):

1. Updating the muscle unit length l_{mtu} based on the current joint state.
2. Updating the muscle internal forces: the parallel F_{pe} and buffer elastic F_{be} forces based on the length l_{ce} computed during the previous iteration.
3. Updating the force-length and force-velocity relationships f_l and f_v.
4. Updating the muscle contraction speed v_{ce}.
5. Time integrating v_{ce} to obtain the contractile element length l_{ce}.

For each leg, the transformation of muscle forces into joint torques can be written as follows:

$$\tau = R \cdot F \tag{8.1}$$

with $\tau \in \mathbb{R}^{6\times 1}$ the joint torques (each leg has 6 degrees of freedom), $F \in \mathbb{R}^{13\times 1}$ the muscular forces (each leg has 13 muscles) and $R \in \mathbb{R}^{6\times 13}$ the moment arm matrix depending on model parameters (pennation angles, etc.) and on the current skeletal configuration.

8.2.2 Neural controller

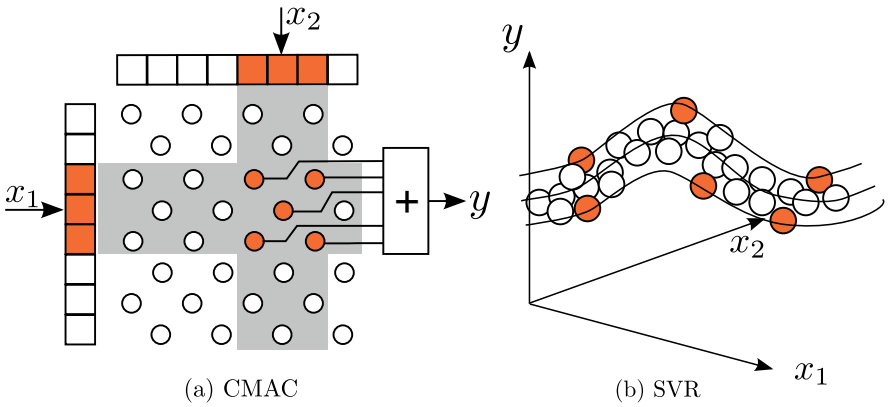
The neural controller is implemented by a regression engine that generates a muscular stimulation S_m (Figure 8.1 (a)) for each individual muscle. Once trained, this engine is expected to produce adequate stimulations as a function of the sensed perturbations. Two algorithms were implemented: CMAC (Cerebellum Model for Articulation Control), a bio-inspired neural network modeling the cerebellum organization (Albus, 1971); and SVR (Support Vector Regression), a modern and powerful regression technique (Smola and Schölkopf, 2004). Both implementations share the same learning objective, i.e. solving the regression problem of finding appropriate muscular stimulations corresponding to a given sensory input (Supervised Learning). Supervised learning is used as a first step to assess the feasibility of muscular control during balancing. Moreover, either CMAC or SVR can only generate a 1D output, so that one learning engine should be connected to each muscle. However, both legs being identical, symmetrical muscles can be actuated by the same engine but with different inputs such that only one module per muscle type is required. In order to comply with the body symmetry, sensory inputs in the frontal and transverse plane must be mirrored, e.g. if one sensory input is y_{COM} (the lateral displacement of the center of mass), the right leg engines receive y_{COM} while the left leg engines receive $-y_{COM}$. The same applies to the other frontal/transverse signals.

Cerebellum model (CMAC)

The cerebellum is known to play a major role in motor adaptation (Smith, 1998). CMAC is a biologically relevant neural network model of the cerebellum providing online learning capabilities. This model captures the reinforcement learning mechanism of the cerebellar Purkinje cells. It builds up a large vector of weights, i.e. a lookup table (LU) with high plasticity. During prediction, depending on the current inputs, some weights are appropriately recruited and summed to produce the output (see Figure 8.3(a) for a simplified representation). A thorough description of the gain recruiting and learning mechanisms is available in (Smith, 1998). In brief, when training is enabled, the core neurons weights evolve according to the following rule:

$$w_k \leftarrow w_k + \alpha \frac{\bar{x} - x}{n_a} \quad (k = 1, \ldots, n_a) \tag{8.2}$$

with w_k being one of the gains recruited during the previous prediction, α the learning rate, $\bar{x}$, the reference stimulation, x the predicted stimulation and n_a the number of association units (AU), i.e. the number of gains recruited for the prediction.



(a) CMAC (b) SVR

Figure 8.3: (a) CMAC model: each input range is discretized. A specific multidimensional input activates a fixed number of input neurons (the orange cells of x1 and x2 in this figure). Based on the activated input neurons, specific core neurons are recruited. Their respective weights are summed to produce the output y. (b) SVR: some data points are elected to become support vectors.

Support Vector Regression (SVR)

SVR is an off-line learning method. It solves a convex optimization problem so it always converges to the global optimum. The model is generated by selecting appropriate data points as support vectors (see Figure 8.3(b)). When training is enabled, the current sensory inputs

and the associated reference stimulations (from the training modules, see next section) are added to the learning dataset. This extended dataset is then used to recompute the regression model. The ϵ-SVR algorithm provided by LIBSVM (Chang and Lin, 2011) was used in this contribution. ϵ-SVR requires the tuning of four parameters in order to provide an appropriate regression model: p the tolerated precision, C the penalization cost, γ the kernel width and K the kernel type. A grid search cross-validation procedure with 10 folds and a radial basis function (RBF) kernel was carried out to identify the optimal regression parameters γ (kernel width) and p (tube size). Each sensory input data was scaled to $[-1,1]$ prior to optimization.

The regression engines, CMAC and SVR, receive the following sensory inputs: (i) all sagittal muscles and the hip int/ext rotation muscles receive the COM forward position and velocity (x_{COM}, $\dot{x}_{COM}$), the COM lateral displacement (y_{COM}), and the hip/foot pitch positions. Additionally, the VAS muscle receives the knee position. (ii) Frontal muscles receive y_{COM}, $\dot{y}_{COM}$, and the hip/foot roll positions.

8.2.3 Reference stimulations

The previously mentioned machine learning algorithms require reference stimulations in order to infer a regression model. These references are computed using two modules (see lower part of Figure 8.1): (i) an impedance controller generating reference torques and (ii) an inverse muscular model transforming these torques into muscular stimulations. The error between the reference stimulations (from the inversion module) and the predicted ones (from the neural controller) is continuously monitored. In normal operation, i.e. once the learning phase converged, the neural controller generates adequate stimulations for the muscles such that it is able to autonomously control the lower-limbs. However, if the error goes above a given threshold, the inversion module takes over the control while learning is enabled again for the corresponding muscle. The *cognitive control ratio* is defined as the proportion of simulation time for which the system is under control of the impedance controller and the inversion module (i.e. with learning being active). As the regression model is trained, this ratio is expected to decrease, capturing that the neural controller gradually becomes autonomous.

8.2.4 Compliant impedance controller

Coping with perturbations requires highly coordinated body motions. Those motions require accurate torque trajectories for a large set of scenarios, difficult to obtain from human data. Therefore, we implemented the full-body compliant force controller introduced in (Hyon et al., 2007). This controller was selected because it requires neither an inverse kinematic nor an inverse dynamic model. It can handle an arbitrary number of contact points with the environment and only requires the computation of the direct kinematic model of each contact point (Hyon et al., 2007). Stability is maintained using virtual feedback forces applied to the center of mass (COM). The robot modulates contact forces with the environment to achieve these virtual forces. A complete description of the algorithm goes beyond the scope of this

contribution, so that only the main computational steps are outlined below:

1. Computing the forward kinematic model for all contact points with the environment (position and Jacobian with respect to the COM) and COM position/velocity.

2. Computing the user force f_u as a proportional-derivative (PD) feedback acting on the COM position: $f_u = \pm(k_p\,(COM_{ref} - COM_{real}) - k_d\,\dot{COM}_{real})$, where k_p and k_d are two gains, COM_{real} is the COM real position, $\dot{COM}_{real}$ its derivative, and COM_{ref} its reference.

3. Computing the environment applied forces with gravity compensation as $f_p = f_u + Mg$, with M the robot total mass and g the gravitational acceleration.

4. Computing the position of the desired center of pressure (COP) and solving a linear optimization problem distributing the contact forces.

5. Transforming contact forces into joint torques using the Jacobian matrices.

In the present contribution, the reference torques being obtained after this step are fed to an inverse muscular module that generates the corresponding muscular stimulations.

8.2.5 Inverse muscular model

Inverting the direct muscular model outlined in Section 8.2.1 requires two steps: (i) solving the over-actuation problem for each leg, i.e. transforming the three sagittal torque references (hip, knee and ankle pitch) and three non-sagittal torques (hip yaw, hip and ankle roll) into 13 muscles forces; and (ii) transforming these desired muscle forces into the corresponding muscle stimulations.

Because multiple muscles actuate the same joint, the human musculo-skeletal apparatus is redundant. Inverting Equation (8.1) is indeed providing an infinite amount of solutions, because the matrix R has rank 6. Isolating a specific solution thus requires using a particular optimization technique. In addition, the inversion should further obey some constraints. Indeed, a muscle can only pull, i.e. provides positive force, and saturates to a given maximal force. Consequently, each force F_i should be bounded between 0 and F_{max}.

Solving over-actuation

The inversion should be computed in real-time since it is intended to work in a real-time learning framework. A quick and efficient way to solve this problem is to rely on linear programming. Indeed, the problem can be stated using only linear constraints and a linear

objective function, i.e.

$$\begin{aligned} \text{Objective:} \quad & min \sum_{i=1}^{13} |F_i|/F_{max}^i \\ \text{Constraints:} \quad & (i)\ 0 \leq F_i \leq F_i^{max}\ (i = 1...13) \\ & (ii)\ \tau = R \cdot F \end{aligned}$$

Normalizing each force by its maximum force is expected to distribute the forces according to the muscle capacities. The linear programming toolbox from the GLPK library (GNU Linear Programming Kit[1]) was exploited to find the unique optimal solution, i.e. the reference muscle forces.

Finding the muscular stimulations

The next step is, for each muscle, to compute the neural stimulations corresponding to this reference force. The following paragraph details the steps required for this computation. The procedure is closely linked to the direct model equations from (Geyer and Herr, 2010).

From the current skeletal configuration, the first step consists in computing all muscles lengths l_{mtu}. The second step consists (ii) in extracting the series element length based on the desired muscular force. By definition, $F_{se} = F_m$, so that,

$$l_{se} = \epsilon_{ref} l_{slack} \sqrt{F_m/F_{max}} + l_{slack} \tag{8.3}$$

with F_m being the desired muscle force, F_{max} the muscle maximum force, l_{slack} the slack length (the series elastic element length under which the buffer element engages) and ϵ_{ref} the muscle reference strain. Then (iii), as depicted in Figure 8.2, the contractile element length can be extracted using the total muscle length l_{mtu}:

$$l_{ce} = l_{mtu} - l_{se} \tag{8.4}$$

The following step (iv) involves a time differentiation, required to obtain the contractile velocity v_{ce}. Indeed, in the direct model, the contractile length l_{ce} is obtained by integrating the contractile velocity. This differentiation is implemented using a backward finite differences scheme of order three. Then, the internal forces and force/length and force/velocity relationships are computed (v) following the direct model. The last step (vi) requires to invert the force relationship, bounding the result in the simulation interval, which is [0.01;1], the

[1] https://www.gnu.org/software/glpk/glpk.html

lower bound being the basal activity, i.e. the minimum muscular activity. This provides the muscular stimulation S_m that corresponds to the desired force F_m:

$$S_m = \frac{F_{ce}}{F_{max} f_l f_v} \tag{8.5}$$

Last but not least, we define the reconstruction error e_r as the difference between the reference torques generated by the impedance controller and the torques generated by the musculo-skeletal model when stimulated with the output of the inversion module. This reconstruction error is central in evaluating the performances of the inversion module.

8.3 Validation tools & protocols

The proposed algorithms were validated by controlling a simulation model of the 95 cm tall COmpliant HuMANoid platform (COMAN, see Figure 8.4). This robot, developed by the Italian Institute of Technology (IIT), has 23 actuated degrees of freedom (DOFs), most of them being equipped with series elastic actuators (Dallali et al., 2013; Tsagarakis et al., 2009, 2013). Each joint is equipped with position, velocity and torque sensors. The robot also features an inertial measurement unit (IMU) and 6-DOF feet sensors measuring ground reaction forces and torques. Our controller only uses the sensory inputs available on the actual robot. The COMAN was modeled in a simulation environment called Robotran (Robotran, 2009). This simulator provides a direct dynamics engine for rigid multi-body systems. An accurate modeling of the robots series elastic joints was implemented and described in (Dallali et al., 2013). For all the following experiments, the robot joints are torque driven using the lower-level controller available on the real robot. As depicted in Figure 8.1, only the lower limbs are driven by the musculo-skeletal model while the other joints are systematically driven by the impedance controller.

In a first experiment, the muscular inversion module was validated by comparing the reference and reconstructed torques, i.e. the torques produced by the impedance controller (Figure 8.1 (c)) and the torques generated by the musculo-skeletal model being driven by the stimulations from the inversion module (Figure 8.1 (b)). The simulation started with the robot standing straight with both feet aligned as depicted in Figure 8.4, rightmost panel. Three perturbations were applied on the robot body, each lasting 0.2s and separated by 3s from each other (see Figure 8.4 for the forces application points): a 25N force on the robot trunk, horizontally in the sagittal plane and in the forward direction; a 15N force on the robot waist, horizontally in the frontal plane, lateral direction; and a 20N force on the robot wrist, horizontally in the forward direction. The signals were recorded with the impedance controller governing the robot reactions to perturbations (i.e. no learning in this case).

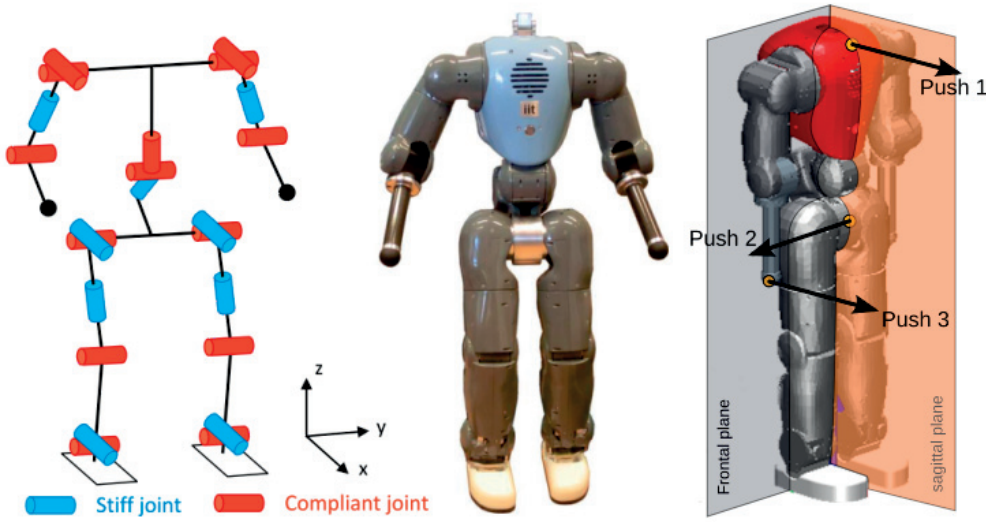


Figure 8.4: COMAN: a 23 DOF compliant humanoid robot: real robot and simulation model (adapted from (Van der Noot et al., 2015a)).

In a second experiment, the learning performance was validated by recording the evolution of the cognitive control ratio (see Section 8.2.3). The threshold error on the predicted stimulations that triggers learning (cognitive control active) was fixed to 0.04. Table 8.2 summarizes the machine learning parameters used for this experiment. The simulation again started with the robot standing straight with both feet aligned as depicted in Figure 8.4. The validation was performed through two different perturbation scenarios.

In the first one, perturbations were restricted to the sagittal plane. The robot underwent a force perturbation on the torso with a random magnitude in the range $[-10,30]\,N$, every $4\,s$ and lasting $0.2\,s$ each. A qualitative validation of the learning progress was also provided for this scenario. The reference and predicted stimulations were recorded at different stages of the learning process for the same $10\,N$ push perturbation on the torso and in the sagittal plane (0, 5 and 50 pushes). As more data becomes available, learning should progress up to making the robot able to predict the stimulations corresponding to the sensory information arising from the perturbed posture.

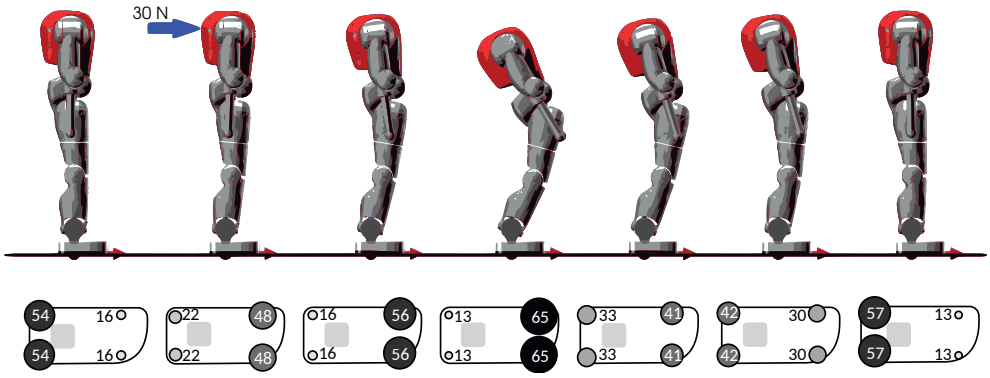
In the second scenario, learning in the other planes (frontal and transverse) was validated. The robot underwent a 3D horizontal force perturbation on the trunk with a random amplitude ("Push 1" application point in Figure 8.4). The sagittal and frontal components are uniformly selected in the ranges $[-5,15]\,N$ and $[-10,10]\,N$ respectively. The duration was fixed to $0.2\,s$ and the perturbation are triggered every $4\,s$. For both scenarios, five runs were performed. Both CMAC and SVR were tested and compared in the first scenario, while the 3D scenario was only tested with SVR due to the relatively poorer performance of CMAC in 2D (see Section 8.4). SVR off-line retraining occurred every $15\,s$ or earlier if more than 5000 stimulation errors exceeding the threshold (across all muscles) have been detected since the last retraining.

Table 8.2: Machine learning parameters

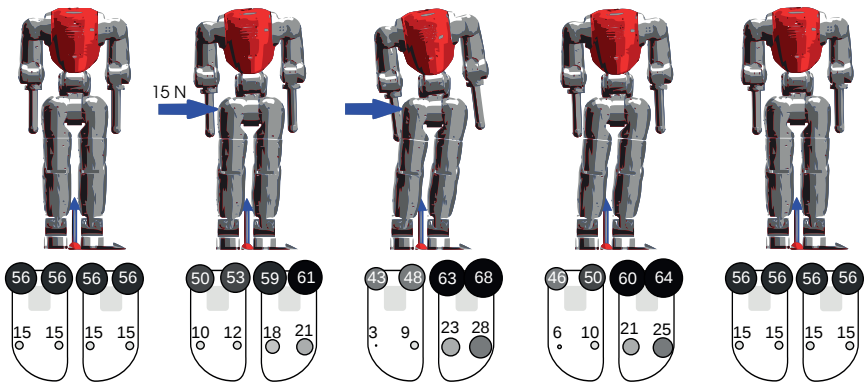
CMAC			SVR			
LU size	Quant.	#AU	C	γ	ϵ	K
400000	300	10	200	1	0.02	RBF

8.4 Results

The quality of the generated reference stimulations (muscular inversion) and the ability of the robot to learn these stimulations were tested in the simulation environment following the previously reported protocol. Figure 8.5 shows the robot recovery motion for a sagittal and a frontal perturbation with the neural controller governing the robot motion. The force under the four virtual foot contact points show how the robot transferred the position of his COP to stabilize the motion of the COM.



(a) horizontal perturbation on the torso in the sagittal plane



(b) horizontal perturbation on the waist in the lateral plane

Figure 8.5: Recovery motion for a horizontal perturbation on the torso (sagittal plane, panel (a)) and a horizontal perturbation on the waist (frontal plane, panel (b)). The lower panels show the evolution of the contact forces (in N) during the recovery motion. In both panels, COM_{ref} is set to the value of the robot COM at homing position.

8.4.1 Inverse muscular reconstruction

Figure 8.6 shows the 13 muscular stimulations for the right leg that were generated for the first experiment (three pushes). A noticeable element is that the gluteus muscle (GLU, see Figure 8.2) was never recruited by the optimization to counter the perturbations. For the same set of perturbations, Figure 8.7 shows the reference and reconstructed torques for the six degrees of freedom from the right leg when using the inverse muscular model. The mean squared error (mse) for each joint is computed.

8.4.2 Cumulative learning

Figure 8.8 gives a qualitative insight into the learning process for the sagittal muscles. As more experience was accumulated, the predicted stimulations (from the SVR regression engine) became more accurate and converged towards the reference stimulations (from the muscular inversion module). At the beginning of the learning phase, the predicted stimulations were systematically zero. At the end of the learning phase, the residual error was systematically below the learning threshold. Figure 8.9 shows the global results of the second experiment, first scenario. In particular, it shows that the control was gradually transferred from the impedance controller to the neural controller as more training data was available. After 160 pushes in the sagittal plane, while SVR managed to fully control the robot, CMAC can only predict accurate stimulations during around 75% of time. In the case of learning with 3D horizontal perturbations (second scenario), SVR also managed to reduce the cognitive control ratio to a low value, as depicted in Figure 8.10.

8.5 Discussion

The previous section provided quantitative results about the ability of the robot to predict muscular stimulations necessary to keep balance. The torques reconstruction of Figure 8.7 shows an excellent performance for computing adequate muscular stimulations. The small reconstruction error has presumably different causes: (i) the limited accuracy of the numerical derivative necessary to compute the muscles contraction velocity, (ii) the basal muscular stimulation that generates parasitic forces (muscular stimulations are bounded between $[0.01, 1]$ with the lower bound being the basal activity) and (iii) the muscular saturation that limits the muscles output force (muscular stimulations saturate to an upper bound). The systematic silence of the gluteus muscle (GLU) can be explained by the fact that the biarticular hamstring muscle (HAM), also acting on the sagittal hip, can generate the required torque alone.

We showed that the neural controller is able to learn the stimulations with high accuracy, leading to an almost exclusive control by the neural controller (i.e. with no muscular model inversion) even for 3D random perturbations. The residual error is probably due to the dimensional reduction of the problem (SVR/CMAC engines only receive a subset of all available

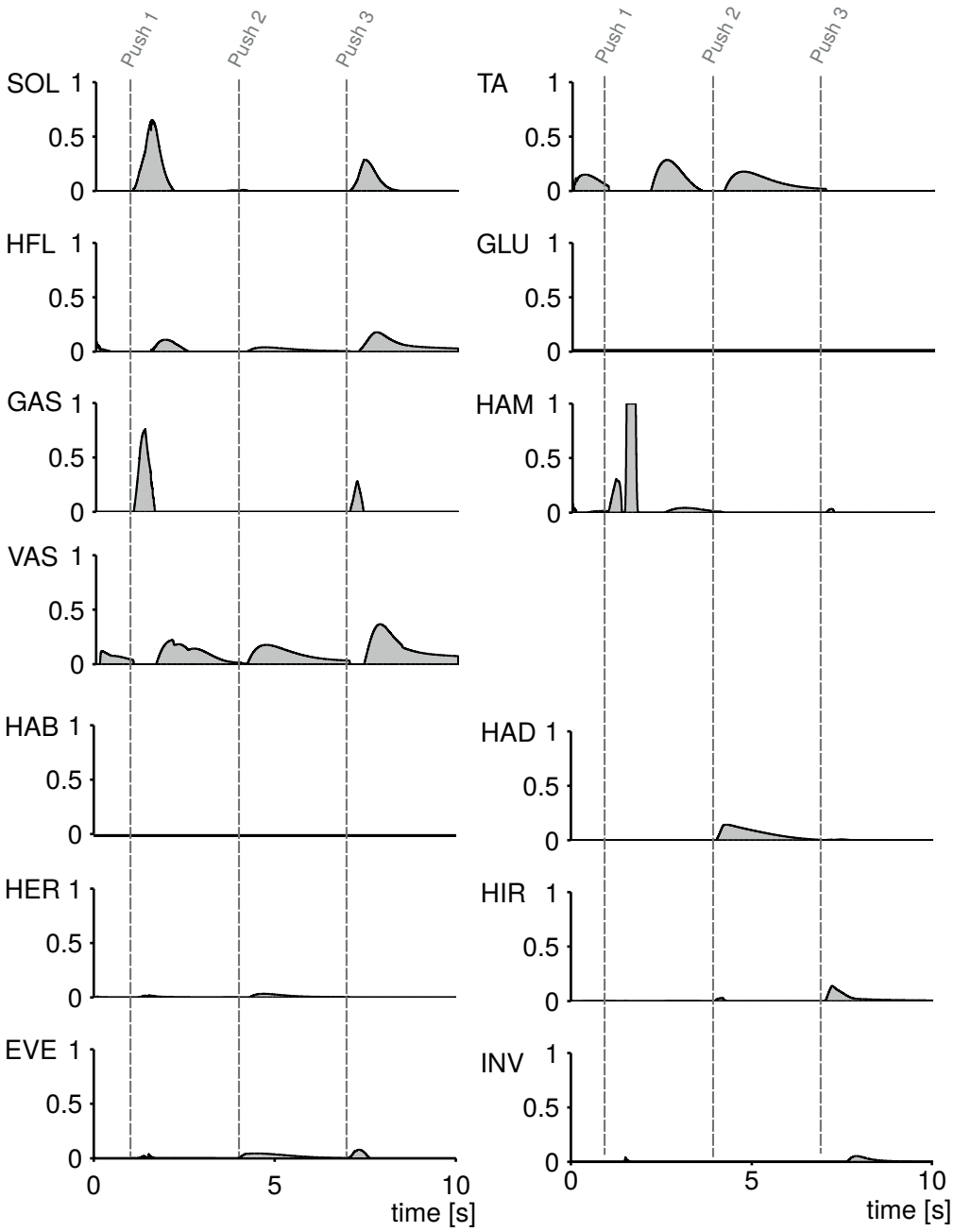


Figure 8.6: Muscular stimulations for all 13 muscles generated by the inverse muscle model when the robot is receiving the benchmark perturbations.

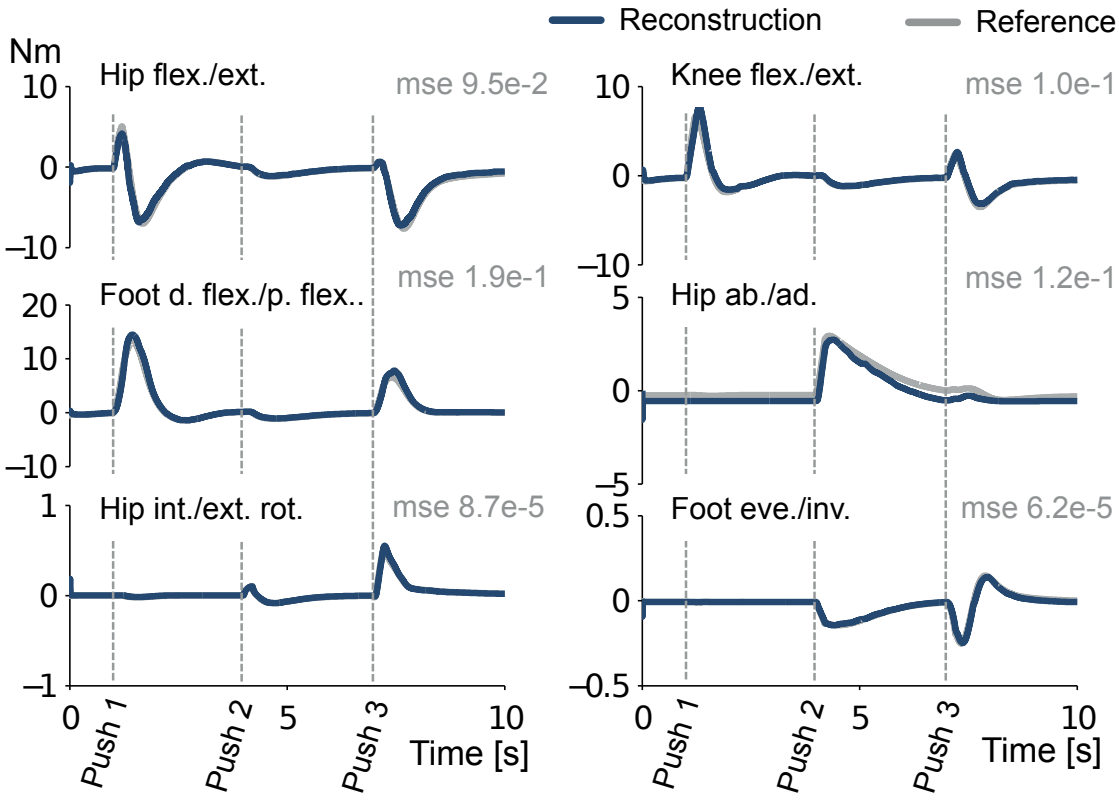


Figure 8.7: The reconstructed torques (dark blue) approximate with high accuracy the reference torques (grey).

sensory inputs to limit the nonlinear growth of the model as a function of the dimension of the input space). With a longer learning period, the robot would presumably further reduce the cognitive control ratio to lower values, the bottleneck being the richness of sensory inputs. We also highlighted the better performance of SVR as compared to CMAC for predicting stimulations when using the given settings. Moreover, the neural prediction is fast compared to the impedance computation (even faster with dedicated hardware) which is important for real-time control. This is highly desirable for future testing on real robots.

8.6 Conclusion and perspectives

The developed neuro-musculo-skeletal model displayed the ability to generate muscular stimulations to counter external perturbations in order to keep balance control of a humanoid robot. Two machine learning techniques, namely CMAC and SVR, were applied for the generation of a regression model capturing the muscular stimulations required for balance. An impedance controller regulating the COM position and a module inverting the Hill-muscle model were also developed to produce reference stimulations required during the learning process. The algorithm was tested in simulation with a torque-controlled child-sized humanoid robot (COMAN). The results suggest that the control can be gradually transferred from

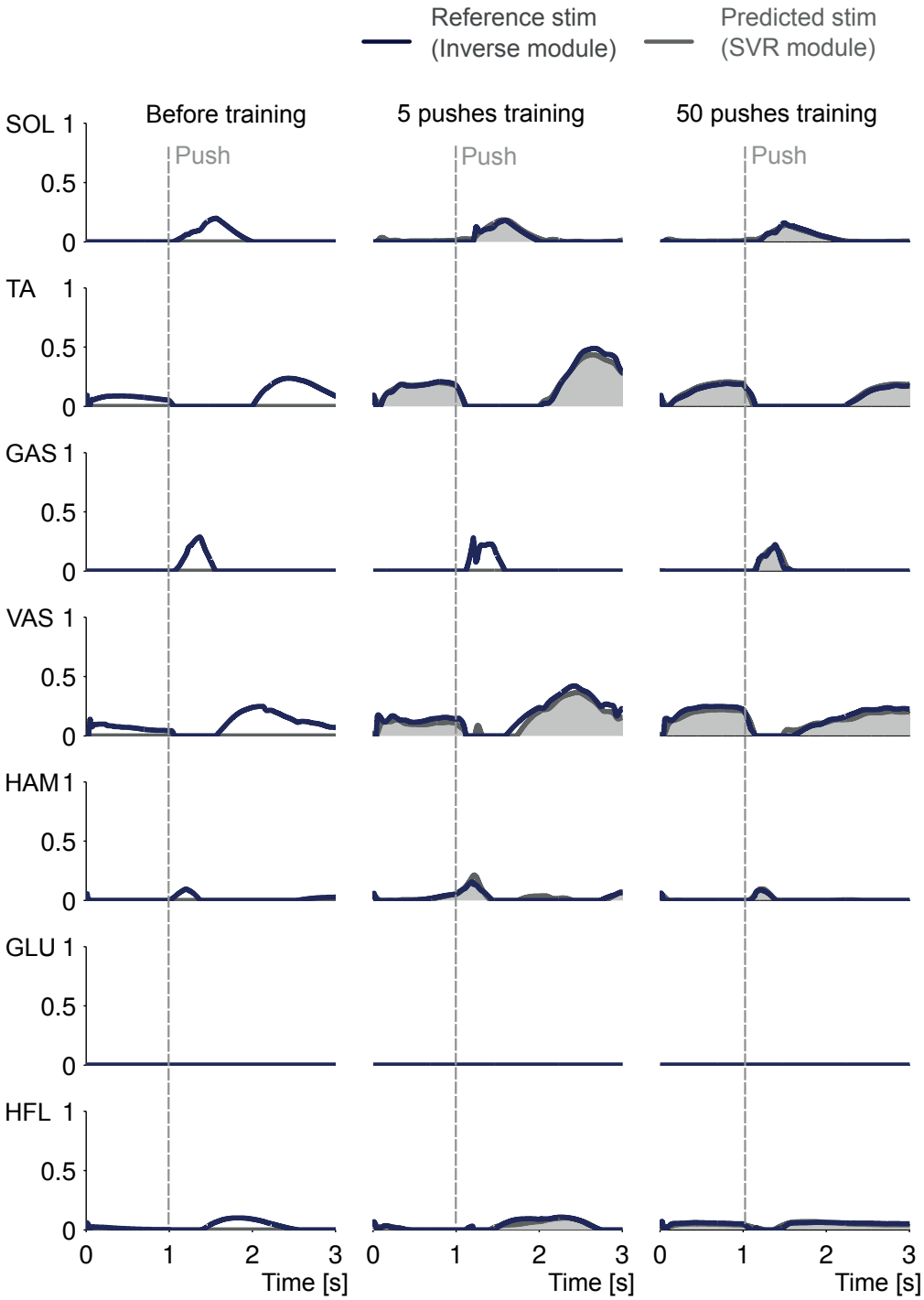


Figure 8.8: Evolution of the predicted muscular stimulations at different learning stages for the same forward push, in the sagittal plane and on the torso (10N). The robot is trained with random magnitude pushes in the sagittal plane.

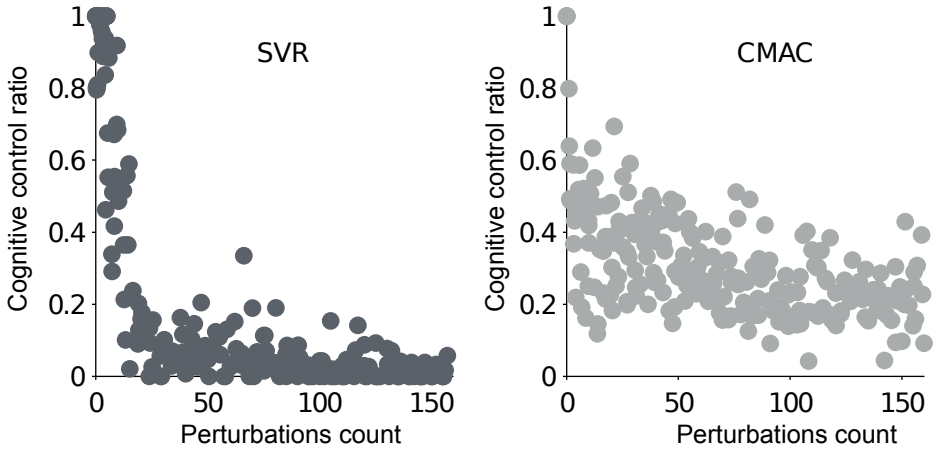


Figure 8.9: The cognitive control ratio decreases as more training data becomes available. The figure overlays five runs while delivering perturbations in the sagittal plane only.

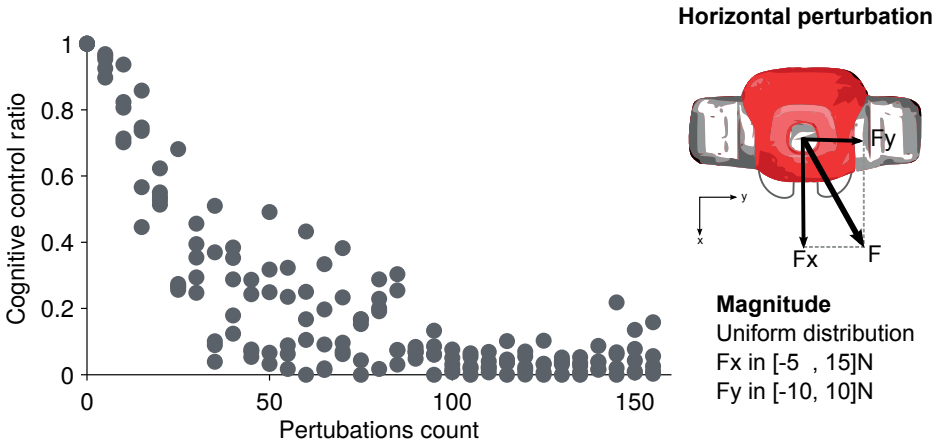


Figure 8.10: Using SVR, the cognitive control ratio decreases as more training data becomes available. The figure overlays five runs while delivering perturbations in the whole transverse plane.

the impedance/inverse model (training modules) to the regression models (neural controller), as learning progresses. This approach was validated by applying random perturbations forces in the sagittal plane and in the whole transverse plane. The robot managed to learn the correct stimulations required to withstand small to medium perturbations, i.e. with no stepping being necessary.

The development of the inverse muscular model is valuable to better understand the actual human behavior when subjected to perturbations. For instance, the stimulations computed by this algorithm can be later compared to EMG signals measured on humans. Or following the same bio-inspired approach, data acquired during human balancing experiments could also be used for learning, taking advantage of biological optimizations.

While muscles display interesting viscoelastic properties that have perturbation filtering capabilities, they also enable the coordination between joints due to the presence of multi-articular muscles with variable stiffness. In the line of this contribution, these properties can be more thoroughly explored to improve the controller robustness. Furthermore, a reinforcement learning technique might outperform the actual performances obtained using supervised learning from the impedance controller. Also, the analysis of the learned stimulations might show some correlations with the sensory inputs, enabling the synthesis of simple neural control rules.

Interestingly, the proposed algorithm manages push recovery by computing virtual muscle stimulations. Other algorithms using stimulations-driven musculo-skeletal models (like the walking controller of (Song and Geyer, 2015a) or the ones developed in this thesis) already produce energy-efficient human-like gaits. Incrementing them with the algorithm developed in this contribution would offer to use the same tools to produce virtual muscle stimulations both for walking and running, as well as for robust postural control. In particular, the algorithm developed in this chapter recruits a PD controller tracking a desired COM position. This module could be adapted to walking or running gaits in case a moving COM reference (i.e. COM_{ref}) was computed. This COM reference could for instance be obtained from a simulation model running in parallel and acting like an internal model.

Finally, the compliant impedance controller and the muscle inversion modules developed in this chapter are both recruited in Chapters 9 and 10, in order to deport the COM position above one foot, before initiating the gait. This is done by linearly adapting COM_{ref} from its initial position (i.e. corresponding to the robot in its homing position) to an optimized target position above one foot.

9 Forward speed modulation during 3D straight walking gaits

Publication

The material presented in this chapter is adapted from the the following submitted paper:

Van der Noot N, Ijspeert AJ and Ronsse R (conditionally accepted) Bio-inspired controller achieving forward speed modulation with a 3D bipedal walker. In: International Journal of Robotics Research (IJRR).

The neuromuscular controller developed in Chapter 6 was driven by the combined action of a central pattern generator (CPG) and reflexes. The resulting gaits could be adapted by modulating a few parameters as functions of the target speed, resulting in step length and frequency alteration. When tested on COMAN, it was possible to continuously modulate the speed from 0.4 to 0.9 m/s, i.e. in a range close to the human one when scaled to the size of the robot. These results were obtained in simulation for 2D gaits. In other words, the waist was constrained to stay in the sagittal plane, so that the walker could not fall laterally.

This chapter extends this contribution to 3D gaits. Therefore, new neural signals are developed to drive the robot non-sagittal degrees of freedom (DOFs), providing lateral stability to the biped. The control of the sagittal DOFs of Chapter 6, while mainly kept intact, is also slightly altered for these 3D scenarios.

Walk initiation is performed by moving the center of mass (COM) above one foot, before initiating the swing phase with the other leg. This COM motion is obtained by computing appropriate stimulations, using the modules developed in Chapter 8.

9.1 Introduction

Mobile robots hold the promise of a better integration of robotics in our everyday life. However, they are usually restricted to environments adapted to their mobility. Humanoid robots offer an interesting perspective in this context, since their body - roughly similar to ours - is

potentially perfectly adapted to our world, designed for humans (Schaal, 2007). Also, they offer the possibility to manipulate tools designed to comply with human dexterity, so that these tools do not need to be adapted for the robot (Fitzpatrick et al., 2016). This is particularly appealing in contexts where the robot is expected either to take over a human laborious duty, or to co-work in synergy with human operators.

Nowadays, these robots skills are still far from reaching the level of the human ones, thus preventing them from being routinely used. This is especially true regarding locomotion. The most popular methods developed to achieve dynamic walking rely on the zero-moment point (ZMP) as an indicator of gait feasibility (Vukobratovic and Borovac, 2004). The ZMP can then be used to generate walking patterns guaranteeing dynamic stability at every moment during the gait. Many locomotion experiments were successfully conducted using this indicator, for example with ASIMO (Chestnutt et al., 2005) or with the HRP-2 platform (Kaneko et al., 2002).

However, there are several shortcomings related to these ZMP-based bipedal controllers, notably energy inefficiency (Dallali, 2011). Furthermore, the generated pattern gaits look quite unnatural (low waist position, permanent knee bending, feet kept parallel to the ground...) and the resulting walking speed is typically much slower than the one achieved by a healthy human displaying the same morphology (Kurazume et al., 2005; Sardain and Bessonnet, 2004b). In particular, ZMP-based controller synthesis usually requires to avoid singular configurations, thus preventing the leg to reach full extension during the stance phase (Kurazume et al., 2005). This has a direct impact on the energetic consumption, since a bended knee requires to maintain a torque balancing the body static and dynamic forces. Some contributions however managed to address this problem (Ogura et al., 2006).

Another concept frequently used to achieve dynamic walking is the inverted pendulum model (IPM). In its most basic version, the IPM models the biped as a single point mass with contact forces acting at the feet level, in order to produce desired motions for the COM. The IPM can then possibly be used to control the ZMP (Faraji et al., 2014a,b). The linear inverted pendulum (LIP) is a special case of the IPM where the point mass is constrained to move in a plane of constant height (Razavi et al., 2017).

The limit cycle walking concept relaxes the need to guarantee the local stability at all times of the gait. It treats the gait as a limit cycle and investigates its global stability (Hobbelen and Wisse, 2007). (Quasi-)passive walkers are successful implementations of this concept (McGeer, 1990; Collins and Ruina, 2005; Hobbelen et al., 2008). Although they display human-like gait patterns and require zero (or little) energetic consumption, they are usually limited to very controlled environments, since they usually lack control variables to modulate the gait or to resist perturbations like obstacles or collisions.

Another avenue to explore the limit cycle walking concept is through the development of so-called bio-inspired walkers. Here, bio-inspiration means that the principles governing the design of the walker's body and/or controller rely on concepts identified in humans. In particular, the seminal paper of (Geyer and Herr, 2010), further extended in (Song and Geyer,

2015a), developed a bipedal model being actuated by a human-like neuromuscular model. Using reflexes to drive these muscles, they could reproduce human-like walking patterns and leg kinematics, and predict muscle activation patterns similar to human walking experiments. In addition, the simulated viscoelastic properties of these virtual muscles provided robustness to external perturbations.

This approach was further extended to provide realistic motions of 3D animated characters (Wang et al., 2012; Geijtenbeek et al., 2013). Interestingly, part of this model was also adapted to control a powered ankle-foot prosthesis (Eilenberg et al., 2010), thus further enhancing the bio-inspired framework. In (Van der Noot et al., 2015a), we brought this controller to a real humanoid robot. When external assistance was provided to the lateral balance, the robot was capable of walking on a treadmill.

However, the reflex rules developed in (Geyer and Herr, 2010) do not feature modulation capabilities, for instance regarding the control of the forward speed. (Song and Geyer, 2012) solved this limitation by optimizing the many parameters of this controller to reach different forward speeds. Large speed variations requested then to run additional optimizations to find new parameter modulations between pre-optimized walking gaits.

An alternative bio-inspired gait modulation strategy requires the addition of a central pattern generator (CPG). CPGs are neural circuits capable of producing rhythmic patterns of neural activity without receiving rhythmic inputs. They feature valuable properties like distributed control, redundancies handling, and locomotion modulation using simple control signals (Ijspeert, 2008).

While locomotor CPGs were identified in many vertebrates, their involvement in human locomotion is still a matter open to discussion (Dimitrijevic et al., 1998). Yet, computational models showed that CPGs could play a major role in human locomotion. For instance, (Taga, 1994) could adapt the locomotion of a bipedal model on uneven terrains, using CPG modulation. (Aoi and Tsuchiya, 2005) could achieve robust walking with a biped robot by recruiting nonlinear oscillators, both in numerical simulations and with a hardware platform. In (Dzeladini et al., 2014), a CPG was added to the controller of (Geyer and Herr, 2010), in order to act as a feedback predictor and, then, to modulate the forward speed. This provided an interesting implementation of Kuo's framework for combining feedback (i.e. reflexes) and feed-forward (i.e. CPG) pathways in the control of a periodic task (Kuo, 2002). In (Paul et al., 2005), a neuromuscular model used a CPG as central element to investigate the effects of a spinal cord injury on locomotor abilities. Importantly, modeling efforts investigating the potential role of CPG in human locomotion ubiquitously display their complex intertwining with feedback mechanisms (Rossignol et al., 2006).

In the present contribution, we embrace the idea of combining a CPG and reflexes in a neuromuscular torque-based controller for bipedal locomotion. More precisely, we design a controller capable of generating robust and human-like locomotion gaits on a 3D bipedal walker. In particular, forward speed modulation is achieved through the adaptation of some

high level parameters, i.e. mainly the CPG inputs. Preliminary results of this controller (i.e. limited to the 2D sagittal plane) were already published in (Van der Noot et al., 2015b).

This chapter is divided as follows. In Section 9.2, the walking controller is extensively detailed. Then, Section 9.3 presents both the simulation environment and the robotic platform that was used for embodying our controller, namely COMAN, a 95 cm tall humanoid robot. The controller is further extended in Section 9.4, in order to achieve forward speed modulation. The resulting gait features are analyzed in Section 9.5, while Section 9.6 evaluates the robustness of the controller when walking blindly in perturbed environments. Finally, Section 9.7 concludes the chapter.

9.2 Controller design and architecture

Our controller is expected to provide torque references for all the joints of a bipedal walker. These torque references are computed from a bio-inspired approach: they derive from forces being produced by virtual muscles. These muscles are in turn "activated" by receiving appropriate stimulations. The coordination of these stimulations is governed by a CPG central unit. Importantly, this contribution reports the successive increments performed while designing this CPG network, in order to generate the stimulation patterns governing different walking features. Combining these stimulations with virtual reflexes, robust and efficient gaits can be obtained after an optimization of the many parameters controlling both the reflexes and the CPG. The different modules developed in this controller, together with the biped embodiment, are summarized in Figure 9.1.

9.2.1 Neuromuscular model

The investigated joints configuration is provided in Figure 9.2. This configuration fits the one of the COMAN robot (Tsagarakis et al., 2013), which served as embodiment for our experiments (see Section 9.3.1). This joint configuration is quite ubiquitous in humanoid robots, so that the proposed controller should be adaptable to many other humanoid robots.

To drive these joints, the robot recruits (virtual) muscles. This approach is directly inspired by the paper of (Geyer and Herr, 2010) and is outlined below. Different muscle groups are identified in each body part, and correspond to muscles of the actual human leg anatomy: 27 different types of muscle groups are recruited to actuate the 23 joints of the biped, as reported in Figure 9.2.

More precisely, each muscle group is computed as a set of equations, called the Hill muscle model (Hill, 1938) and pictured in Figure 9.2e. Each muscle tendon unit (MTU) consists of two main elements: a contractile one (CE) and a series elastic one (SE). Two additional passive elements further engage when the muscle state is outside its normal operation range: the parallel elastic one (PE) and the buffer elasticity one (BE). The length l_{mtu} of each MTU is

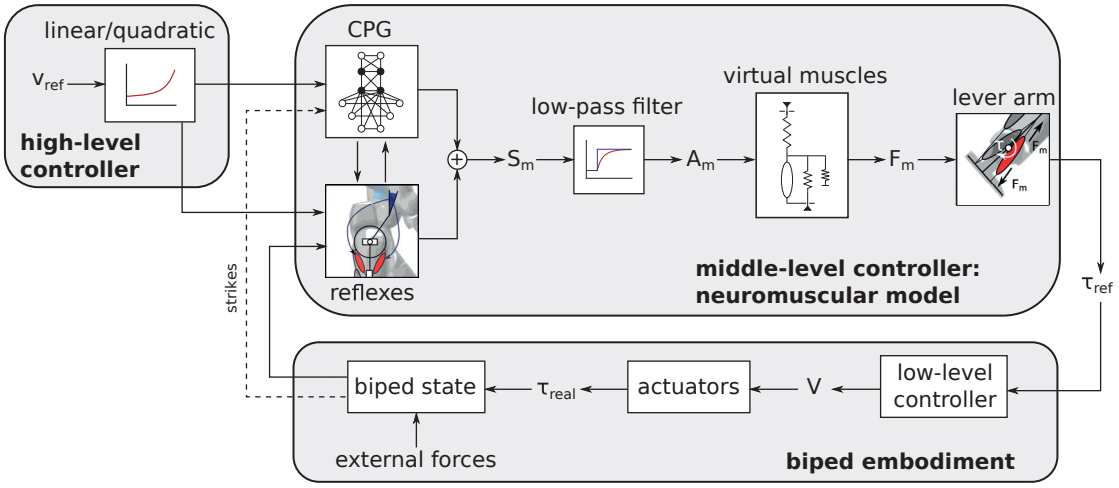
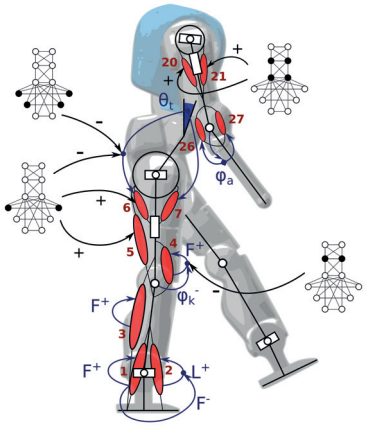


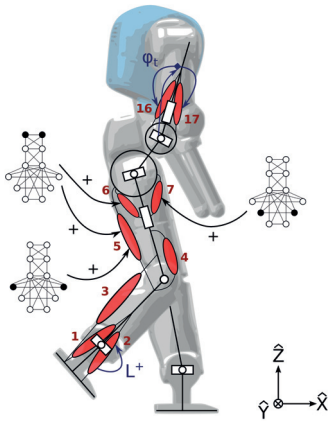
Figure 9.1: The neuromuscular controller purpose is to provide torque references τ_{ref} to the biped joints when receiving sensory information from the biped state. On top of that, high-level commands are provided by the user as linear or quadratic functions of a scalar input: the speed reference v_{ref}. Then, the interplay between the CPG and the reflexes provides stimulation signals S_m. They are later converted into activations A_m controlling the virtual Hill-type muscles. These muscles finally produce forces F_m, converted to the joint torques via lever arms. The biped embodiment used in this contribution tracks the desired torques τ_{ref} by feeding the actuators with appropriate voltages V. The actual torques τ_{real}, combined with the external forces, drive the time evolution of the biped state, eventually resulting in locomotion. This figure is a specific case of the more general scheme of Figure 2.3.

computed by geometrical relationships involving the joint angles and the MTU attachment points. The length of CE l_{ce} is integrated based on l_{mtu} and on the muscle activation A_m, which is detailed later. Then, the deformation of SE (i.e. the length l_{se}, computed as $l_{se} = l_{mtu} - l_{ce}$), provides a direct computation of the force F_m generated by the muscle. Finally, this force F_m is multiplied by the muscle lever arm r_m to generate a torque contribution to the corresponding joint. For bi-articular muscles (i.e. GAS and HAM in Figure 9.2a and 9.2b), a single muscle provides two torque contributions with two different lever arms. The full implementation of these equations can be found in Appendix G.1.

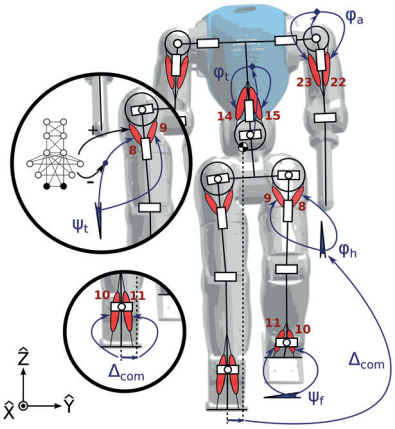
In sum, this musculo-skeletal model provides joint torques through virtual muscle forces and attachment points. So, instead of directly controlling the torques, we rather control each MTU through input signals called muscles activations A_m. They are related to neural inputs S_m called stimulations, using a first-order low-pass filter capturing the excitation-contraction dynamics (see Figure 9.1 and Appendix G.1.2). The following sections detail how the stimulations S_m of each muscle are computed.



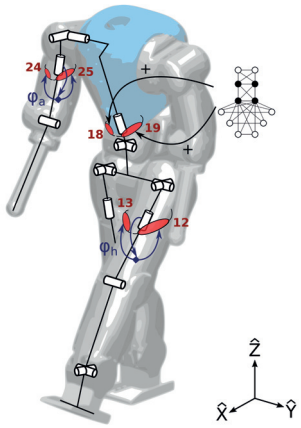
(a) Sagittal muscles (arm and stance leg)



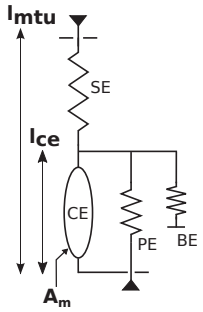
(b) Sagittal muscles (torso and swing leg)



(c) Lateral muscles (arm, torso and leg)



(d) Transverse muscles (arm, torso and leg)



(e) Hill muscle model

Sagittal leg		Lateral leg		Torso		Arms	
1	SOL	8	HAB	14	BTR	20	SET
2	TA	9	HAD	15	BTL	21	SFL
3	GAS	10	EVE	16	BET	22	SAB
4	VAS	11	INV	17	BFL	23	SAD
5	HAM			18	BRR	24	SER
6	GLU	**Transverse leg**		19	BRL	25	SIR
7	HFL	12	HER			26	EET
		13	HIR			27	EFL

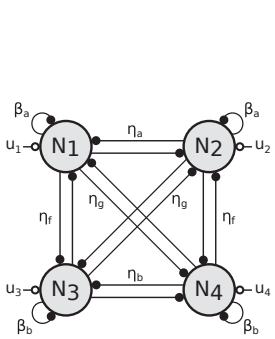
Figure 9.2: To actuate the biped's 23 joints, the controller recruits 27 different Hill muscle models (panel (e)) acting in different planes. These muscles are commanded by a combination of reflex signals and the CPG central unit. Muscles acting in the sagittal plane are displayed in panels (a) and (b), the ones affecting the lateral plane are displayed in panel (c), and finally, the ones acting in the transverse plane are depicted in panel (d).

9.2.2 Frequency and phasing signal construction

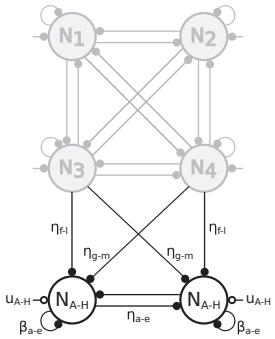
Our controller uses both CPG signals and reflexes to drive the muscles. The combination between these two types of signals mainly follows a proximo-distal gradient. In other words, muscles close to the hips are mainly controlled by CPG signals (feed-forward), while the ones close to the feet are mainly driven by reflexes (feedback) (Dzeladini et al., 2014). This builds upon the rationale that distal muscles are more impacted by external perturbations like ground interactions (Daley et al., 2007).

Our CPG is designed as a twelve-neurons network of Matsuoka oscillators (Matsuoka, 1985, 1987). These are bio-inspired artificial oscillators, capturing the mutual inhibition between half-centers located in the spinal cord. They also have interesting properties. Indeed, they feature stable limit cycles, have a low computational cost and are easy to integrate with sensory feedback signals.

In this contribution, the CPG network is divided into two main parts (see Figure 9.3). The first one is in charge of providing the main frequency and phasing of the gait cycle. Its neurons are denoted with a number (from 1 to 4) and are called "rhythm generator" neurons (RG). The second layer relies on the RG neurons to generate signals shaping the patterns of muscle stimulations. The corresponding neurons are denoted with a letter (from A to H) and are called "pattern formations" neurons (PF). This two-layered division is inspired by the two-level CPG biological structure proposed by (McCrea and Rybak, 2008). In that contribution, the authors report several experiments of fictive locomotion in the decerebrated cat that can be reproduced with this particular CPG architecture.



(a) Rhythm generator (RG)



(b) Pattern formation (PF)

Figure 9.3: The CPG network is built by assembling two types of components: (a) the rhythm generator (RG) part (four fully connected Matsuoka neurons) and (b) a pair of pattern formation neurons (PF) driven by the RG neurons. The vertical symmetry corresponds to the left/right legs symmetry.

During the gait cycle, the strike impact is a crucial moment where the load is quickly transferred from one leg to the other. Simultaneously, a large effort is requested from the new stance leg to prevent the torso from collapsing forward, as a result of this large impact. Therefore, it is

critical for the CPG network phase to be synchronized with the foot strike, so that it provides large stimulations right after impact. During the following loading response, the leg leaving the stance phase must also provide significant efforts, in order to propel the body and prepare the swing phase through proper hip flexion and foot push-off. Next, before the following strike, hip moments are less significant in both legs. Indeed, the stance leg already absorbed the main shock and only needs to maintain the torso orientation, while the swing leg mainly relies on ballistic motion. As a consequence, it is convenient to divide the gait cycle into four stages. Two stages are triggered by foot strikes from both legs, while the two others approximately start during mid-stance. This decomposition is similar to the high-level control states presented in (Yin et al., 2007) or in (Wang et al., 2012).

The CPG RG part is thus constructed with four neurons, one for each stage. More precisely, we use four fully connected Matsuoka neurons (Matsuoka, 1985, 1987). This structure is displayed in Figure 9.3a.

The Matsuoka equations governing this CPG are detailed below. Each neuron N_i main state is captured by its so-called firing rate x_i. Its evolution with time is governed by Equation (9.1), where τ is the time constant for the rate of discharge, v_i is the self-inhibition modulated by an adaptation constant β_j and u_i is the external input:

$$\dot{x}_i = \frac{1}{\tau}(-x_i - \beta_j v_i - \sum_{1}^{3} \eta_k [x_l]^+ + u_i) \tag{9.1}$$

Finally, the connexion strengths η_k govern mutual inhibition, i.e. the fact that the activation of a given neuron decreases when another is active. It is captured by the function $[\bullet]^+ = max(0, \bullet)$, so that only positive firing rates are considered for inter-neurons inhibition. The self-inhibition state variable is governed by Equation (9.2), whose time constant is related to the one of Equation (9.1) through the adimensional parameter γ_j:

$$\dot{v}_i = \frac{1}{\gamma_j \tau}(-v_i + [x_i]^+) \tag{9.2}$$

In Equations (9.1) and (9.2), the index i corresponds to the neuron index, while the gains β_j, η_k, and the neurons x_l are specified in Figure 9.4a. Finally, γ_j takes the same index as β_j. These equations are fully developed in Appendix G.2.

Interestingly, the time constant τ is inversely proportional to the CPG frequency. This provides a useful access for modulating the gait frequency. Regarding phase locking, different models exploited the capacity of CPGs to achieve entrainment, i.e. to synchronize their firing pattern with stimulations generated by the actuated body and/or its environment. In particular,

(Aoi et al., 2010) developed a locomotor CPG model to achieve bipedal locomotion, also by recruiting a two-level CPG biological structure (i.e. combining RG and PF networks). In this model, phase resetting was applied to the RG layer, based on foot-contact information. CPG entrainment was also achieved using Matsuoka oscillators. For instance, in (de Rugy and Sternad, 2003) and (Ronsse et al., 2009), this mechanism was investigated for uni- and bi-manual upper-limb movements, while (Paul et al., 2005) and (Taga, 1994) investigated locomotion. Here, a similar mechanism generating short excitations modulations at foot strike is used. Basically, all the excitations u_i consist in a tonic excitation of $u = 1$. Then, if a neuron N_i is too slow (i.e. not firing while the corresponding phase already started) or too fast, its excitation u_i is shortly modulated as reported in Appendix G.3. Combining it with the time constant τ modulation, this guarantees that the CPG and the walker display the same frequency, while staying in phase with feet strikes.

The four RG neurons N_1, N_2, N_3 and N_4 are the central elements of the whole CPG network in Figure 9.4a. Their typical firing rates temporal evolutions are pictured in Figure 9.4b. In the next sections, this network is incremented with the PF neurons.

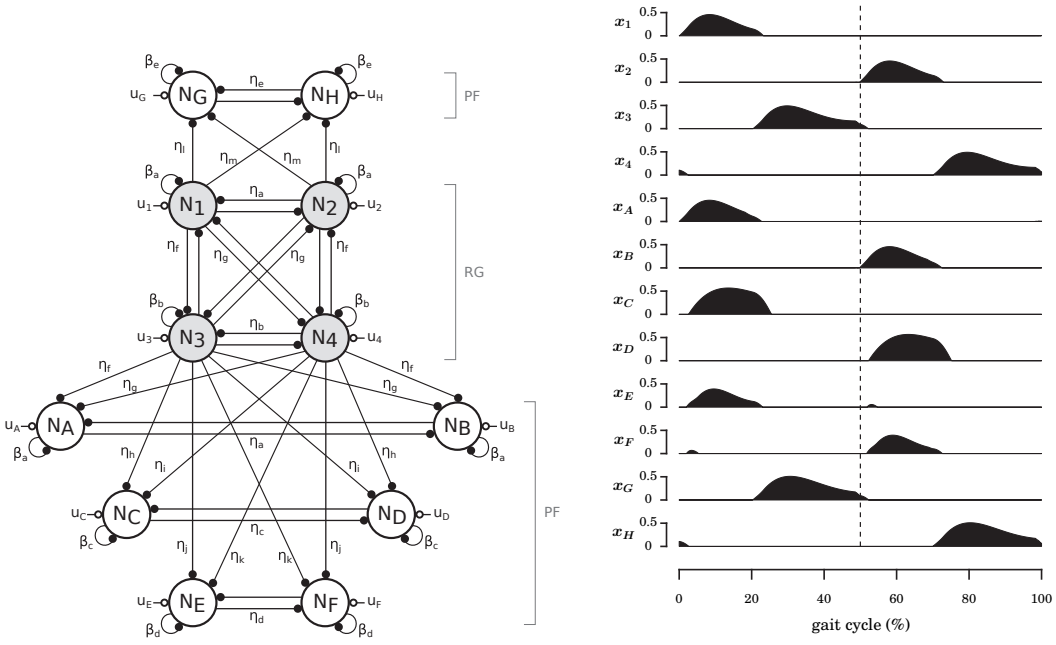
9.2.3 Leg sagittal stance control

The four RG neurons network determines the CPG frequency and phase synchronization. In order to send appropriate stimulations to the muscles, this network is further incremented with pairs of pattern formation neurons (PF). These receive inputs from the RG neurons but not the other way around. This is achieved with the unidirectional structure displayed in Figure 9.3b.

To generate the CPG contribution to a particular muscle stimulation S_m, the different CPG outputs y_i are computed as detailed in Appendix G.4.1. They mainly consist in extracting the positive firing rate of a PF neuron x_j (i.e. $y_i = [x_j]^+$). Then, the CPG contribution to a particular stimulation is computed as $S_m = \sum k_i\, y_i$, where k_i is a gain.

As mentioned earlier, fast hip muscle reactions are required after strike impact to prevent the torso from collapsing forward. This is provided by the gluteus (GLU) and hamstring (HAM) muscle groups. Therefore, neurons being aligned (i.e. firing at the same time) with the N_1 and N_2 neurons of the RG structure are requested, so that they can quickly fire right after strike. This is the purpose of the two neurons N_A and N_B (see Figure 9.4a). They are in charge of providing the requested stimulation patterns. In order to keep them aligned with N_1 and N_2, similar weights are used for the self and mutual inhibitions, as well as for the time constant gains. As can be seen in Figure 9.4b, their firing signals (x_A and x_B) are indeed well aligned with x_1 and x_2, as expected.

After the strike impact absorption, reflexes are activated at the hip level to maintain the torso sagittal lean angle θ_t close to a reference θ_{ref}. The requested muscles are the hip flexor (HFL) and GLU muscles. As proposed in (Geyer and Herr, 2010), this is performed by



(a) full CPG network (b) neurons firing rates

Figure 9.4: Panel (a): Full CPG network: inter-neuron excitations are indicated with an empty circle, while plain circles represent inhibitions. The "rhythm generator" neurons (RG, shaded) affect the "pattern formation" ones (PF), but not vice versa. The network vertical symmetry produces motor commands for both body sides (legs and arms). The neurons' main contributions are the following. $N_{\{1-4\}}$: rhythm generator and upper-body control; $N_{\{A,B\}}$: knee bending and torso sagittal stabilization; $N_{\{C,D\}}$: hip flexion; $N_{\{E,F\}}$: torso lateral stabilization; $N_{\{G,H\}}$: late swing leg retraction. The corresponding muscular activations are highlighted with plain circles in Figure 9.2. The full CPG equations are provided in Appendix G.2. The corresponding CPG equations are provided in Appendix G.2. Panel (b): Time-evolution of the twelve neurons firing rates of Figure 9.4a over one gait cycle (0% and 100% correspond to right foot strikes, the dashed line corresponds to left foot strike). These signals are obtained during one typical gait cycle of the locomotion resulting from the controller used in most of the results of this contribution (called *reference controller*), with a speed reference of 0.65 m/s.

a proportional-derivative (PD) control of the lean angle error, $\Delta_{PD} = \theta_{ref} - \theta_t$. This signal generates a stimulation to one of the two antagonist hip muscles, i.e. one muscle receives a stimulation proportional to $[\Delta_{PD}]^+$, the other to $[\Delta_{PD}]^-$ (with $[\bullet]^- = -min(\bullet, 0)$). This reflex can however send contradictory signals to the ones generated by the CPG. To avoid this, an inhibition mechanism ruled by the CPG was implemented (see Appendix G.4.1).

The remaining leg sagittal muscles are the distal ones, namely soleus (SOL), tibialis anterior (TA), gastrocnemius (GAS) and vasti (VAS) muscle groups. They are mainly controlled by similar reflexes as those reported in (Geyer and Herr, 2010). Most of them either combine a positive constant prestimulation (S_0) with positive/negative force feedbacks ($F^{+/-}$), or a local positive length feedback (L^+). On top of that, the VAS reflex is inhibited when the knee exceeds

a given threshold to prevent over-extension; or during the double support phase of the leg entering in swing phase, in order to allow knee flexion.

All the reflexes mentioned in this section are only activated during the stance phase, i.e. when the ground reaction force vertical component under one foot is larger than an arbitrary threshold (here, 20 N). The full sagittal stance control is presented in Figure 9.2a. Further details about its implementation can be found in Appendix G.4.

9.2.4 Leg sagittal swing control

Because swing leg motion is less affected by external perturbations, its control mainly relies on feed-forward stimulations provided by the CPG. First, hip flexion is achieved by sending appropriate stimulations to the HFL muscle. This activation already starts in late stance, usually a bit after the contralateral foot strike, and spans during early swing. Therefore, the CPG network is augmented with a new pair of PF neurons: N_C and N_D. As expected, their corresponding firing rates x_C and x_D fire slightly after the contralateral leg strike (see Figure 9.4b).

Approximatively at the same time, knee bending is achieved through proper HAM muscle activation. Preliminary results showed that it was actually not necessary to add a new pair of PF neurons to control it. Indeed, the corresponding stimulations usually require to be aligned with the existing neurons N_A and N_B. Consequently, we decided to directly shape the corresponding stimulations based on the x_A and x_B neurons firing rates.

After this initial high activity, swing mainly relies on the leg ballistic motion. Therefore, most muscles only receive the basic tonic stimulation. Regarding reflexes, only TA still receives a similar local positive length feedback (L^+) as the one introduced by (Geyer and Herr, 2010), in order to increase foot clearance with the ground.

In late swing phase, the swing leg motion is reduced by the combined action of HAM and GLU, participating into leg retraction. This is achieved with a new pair of PF neurons: N_G and N_H. In contrast to other PF neurons, this new pair is connected to RG neurons N_1 and N_2, so that they are mainly aligned with N_3 and N_4.

The sagittal swing control described in this section is summarized in Figure 9.2b. Its full implementation is described in Appendix G.4.

9.2.5 Leg non-sagittal control

Regarding the leg control in the lateral plane, the gait cycle is only divided into two phases: the supporting and non supporting ones. A leg supporting phase starts with the leg own strike and finishes with the contralateral leg strike. In other words, it corresponds to its stance phase shortened by the terminal double support phase.

During the supporting phase, the hip abductors (HAB) and adductors (HAD) muscles are mainly in charge of controlling the torso lateral lean angle Ψ_t. Similarly to the stance hip control in the sagittal plane, a pair of PF neurons is required to provide a first excitation to the new supporting leg, and prevent the torso from collapsing sideways. This is achieved by the neurons pair N_E and N_F, acting on the HAB muscles.

After the leg first impact, a closed-loop (i.e. reflex-based) PD controller is in charge of maintaining the torso lean angle Ψ_t close to a reference Ψ_{ref}. Similarly to what was done in the sagittal plane, the CPG can inhibit the PD control contribution on the HAB muscle. This inhibition is triggered according to the CPG phase, to prevent contradictory signals between the CPG and this balance controller. In (Song and Geyer, 2013), a similar PD controller is proposed for the whole stance (i.e. no CPG signal is used). The introduction of the CPG first burst allows tuning of the PD control parameters governing the balance dynamics only after shock absorbance. Indeed, closed-loop angle control appears not appropriate during the double support phase, when the weight is transferred from one leg to the other.

Lateral hip control during the non supporting phase is inspired from the approach described in (Yin et al., 2007) and used in (Song and Geyer, 2013). Basically, an active swing foot placement is implemented based on Δ_{com}, the lateral position of the center of mass (COM), relatively to the supporting foot. First, a hip lateral reference position $\varphi_{h,l,ref}$ is computed as the output of a PD controller on Δ_{com}. Then, a second PD controller tracks this reference position with the hip lateral position $\varphi_{h,l}$, by sending appropriate stimulations to the HAB and HAD muscles.

Regarding lateral foot control during the supporting phase, the eversion (EVE) and inversion (INV) muscle groups are in charge of maintaining the body upright by bringing the lateral COM close to a reference position. Again, a simple PD feedback control is applied on Δ_{com}, i.e. on the same input as the one used to compute the hip lateral reference $\varphi_{h,l,ref}$ of the contralateral leg. During the non-supporting phase, EVE and INV control the foot lateral orientation to keep it aligned with the horizontal, in order to prepare proper foot landing. The full leg lateral control is presented in Figure 9.2c.

Finally, the hip transverse joint is controlled by the hip external (HER) and internal (HIR) rotator muscle groups. The generation of straight motion simply requires to maintain this joint in its homing position. Our control is illustrated in Figure 9.2d. All the non-sagittal control rules are fully detailed in Appendix G.4.

9.2.6 Upper-body control

Upper-body control is less critical during walking. In fact, preliminary experiments revealed that freezing the upper body joints would not prevent from achieving stable walking. However, this resulted in slower gaits, with higher energetic consumption in the lower limbs.

The rationales governing upper body motion in unconstrained human walking is still not clear either. For instance, (Collins et al., 2009) explored whether the extra cost required to swing

the arms could lead to potential benefits in the lower limbs. These experiments showed that voluntarily holding the arms required 12 % more metabolic energy.

Consequently, our controller also implements arm swing motion in the sagittal plane. More precisely, the shoulder flexion (SFL) and extension (SET) muscles are stimulated by appropriate CPG neurons, in order to be in phase with the gait cycle. For the sake of simplicity, the RG neurons were directly used to drive the corresponding muscles. Note however that extra PF neurons might further be added for the upper-body, in a similar way as for the lower-body. Here, SFL and SET stimulations are designed to be in phase with the contralateral leg motion.

The other arm muscles are the elbow extension (EET) and flexion (EFL) muscle groups, the shoulder abduction (SAB) and adduction (SAD) ones and the shoulder internal (SIR) and external (SER) rotation ones. They are all controlled with a simple feedback controller to maintain a constant position.

Similarly to the arms swinging motion, the four RG neurons are used to control the torso transverse joints with the back rotation right (BRR) and left (BRL) muscle groups. The remaining torso muscle groups, i.e. back tilt right (BTR) and left (BTL), back flexion (BFL) and extension (BET), use again PD control on their respective joints to stabilize the homing position. All these rules are summarized in Figure 9.2 and fully described in Appendix G.4.3.

9.2.7 Walk initialization

Walk initiation requires the walker to move its COM on top of one of its feet. This is achieved with the muscle control scheme proposed in (Heremans et al., 2016). Basically, a full-body compliant force controller uses virtual feedback forces applied to the COM to generate appropriate torques at the joint level (Hyon et al., 2007). Then, the muscle model presented in Appendix G.1 is inverted to get the corresponding muscle stimulations. This controller only requires the horizontal coordinates $(X_{init}; Y_{init})$ of the target COM position. These coordinates are optimized as presented in Table G.2 (see Appendix G.4).

Once the COM is above the desired foot, this COM controller is deactivated and replaced by the main controller described in this contribution. However, to guarantee that the CPG quickly converges towards its correct state, special excitations are applied during the first 0.2 s of the gait (see Appendix G.3). Similarly, special stimulations are sent to the HAB and HAD muscles to help initial lateral hip control (see Appendix G.4.1).

9.2.8 Optimization

In the controller development, we introduced many parameters requiring proper tuning. They are all listed in Table G.2 with their respective bounds. In this contribution, this tuning was performed through an optimization phase relying on a particle swarm optimization (PSO) algorithm (Kennedy and Eberhart, 1995).

More precisely, each set of optimized parameters was tested with a biped walking during a maximal time of 60 s. After this duration (or earlier if the walker fell), a staged fitness function was computed. This means that different objectives are sorted by order of relevance, such that the next objective is taken into account only when the previous one nearly reaches maximum score. Each fitness stage is limited between 0 and 100. They are described below.

The first stage requests the biped to walk a minimal distance of 15 m, providing a reward proportional to the traveled distance before falling. The main purpose of this stage is to prevent the walker from staying in its initial upright position. After completion of this objective, a second stage requires the biped to walk without falling during the 60 s simulation time, the fitness being proportional to the walked time.

Once this objective is reached, the speed is later optimized to match a reference. The corresponding objective function is given in Equation (9.3), where f is the stage objective function, x the parameter to be constrained (here, the speed), x^* is the reference and α is a weight (set to 100 for this speed stage). This function output is thus bounded between 0 and 100 and presents a bell-shaped profile around the reference x^*.

$$f = 100\, e^{-\alpha (x-x^*)^2} \tag{9.3}$$

When the biped speed is in a range of 0.05 m/s around the target speed, the last three stages are activated in parallel. The first minimizes the equivalent metabolic energy consumption in virtual muscle contraction per unit distance walked. This energy is computed as detailed in Appendix G.1.3. The fitness stage is computed again with Equation (9.3) where α is set to 10^{-3}, x^* to 0 and x is the metabolic energy consumption of both legs per unit distance walked and normalized by the walker mass. The purpose of this stage is not to minimize the actual electrical energy consumption of the robot, but rather to emulate energy saving mechanisms that are likely taking place in real human walking. Indeed, the minimum metabolic energy per unit distance traveled is considered as a valid measure of walking performance, in order to reproduce the salient features of normal gaits (Anderson and Pandy, 2001).

The RG neurons in the CPG network offer to predict when the next strike will happen (i.e. when x_1 or x_2 will start firing). To encourage the emergence of solutions minimizing this prediction error, the mean error between the CPG predicted strike times and the actual ones is computed. The second parallel optimization stage uses Equation (9.3) again, with α set to 250, x^* set to 0 and x set to the mean of this prediction error.

Finally, to avoid lateral leg inter-penetration, the lateral distance between foot strikes of both legs is also optimized. More precisely, the shorter distance between a strike foot position of one leg and the line passing through the last two strike positions of the other leg is computed. The third parallel fitness stage is computed proportionally to the average of this distance,

saturating the fitness to 0 for 9 cm and to 100 for 14 cm. Importantly, some of the numerical parameters presented here depend on the walker embodiment, in this case the COMAN robot presented in Section 9.3.1.

To promote the emergence of solutions with good foot clearance with respect to the ground, obstacles were placed below the swing foot during the optimization. More precisely, these obstacles were trapezoidal shapes located next to the stance foot. Their height linearly increased with the simulation time from 0 cm to 4 cm. Consequently, foot clearance progressively improved when walking a longer distance.

Finally, some noise was added to the muscle stimulations during optimization. More precisely, the noise potential amplitude was set to 5% of the stimulation instantaneous amplitude, similarly to the signal-dependent noise observed in real human signals (Faisal et al., 2008). This noise was combined to the one applied to the motors (see Section 9.3.2). To cope with this uncertainty, each set of parameters was evaluated three times in a row for each optimization. The average fitness value was used, so that more robust controllers were obtained.

9.3 Embodiment and simulation environment

To test the controller presented in Section 9.2, the COmpliant huMANoid (COMAN) robotic platform was used as embodiment. This robot and its controller were developed in a simulation environment reproducing the articulated body dynamics, the ground external forces, as well as the robot motor dynamic equations.

9.3.1 COMAN platform

The COMAN platform is a 23 degrees of freedom (DOFs) full-body humanoid robot. This 95 cm tall robot, weighting 31 kg, was developed at the Italian Institute of Technology (IIT) (Dallali et al., 2013; Tsagarakis et al., 2013). COMAN is pictured in Figure 9.2, along with the reference frames used in the rest of this contribution to describe its kinematics and dynamics. All sagittal joints, as well as the transverse torso and the lateral shoulder joints, feature series elastic actuators (SEA) (Tsagarakis et al., 2009). The other joints are actuated using traditional, stiff actuators.

Regarding the robot sensors, each joint features position encoders, along with custom-made torque sensors. The torque tracking is then mainly achieved with a PI controller, as presented in (Mosadeghzad et al., 2012). On top of that, an inertial measurement unit (IMU) is attached to the robot waist. Finally, custom-made 6 axis force/torque sensors are placed below the ankle joint to measure the ground interaction forces and torques.

9.3.2 Simulation environment

The simulation suite we used to model COMAN is called Robotran (Samin and Fisette, 2003; Docquier et al., 2013). It is a symbolic environment for multi-body systems developed within the Université catholique de Louvain (UCL). Its direct dynamics module was used to generate the symbolic equations of the robot dynamics. To further minimize the gap between simulation and reality, a particular attention was paid to the actuator dynamics, the signals noise and the environment external forces, in particular the ground contact model (GCM). Moreover, we only used sensory signals available on the real robot (see Section 9.3.1).

The actuators model was implemented as reported in (Dallali et al., 2013) and in (Zobova et al., 2017). To control them in simulation, we implemented a low-level controller similar to the one outlined in (Mosadeghzad et al., 2012). To comply with a realistic noisy environment, a uniform noise with a maximal amplitude of $0.4\,Nm$ was added to the actual torque measured in the simulation environment (see also (Van der Noot et al., 2015a)). This corresponds to the noise level obtained from measurements with the real platform. Consequently, the torque references computed by the controller developed in Section 9.2 were not directly applied to the multi-body system joints (see Figure 9.1). Indeed, they were affected by the motor dynamic equations and their sensory noise, as would happen on a real robotic platform.

Regarding external forces, we used two types of custom-made models: (i) a mesh-based model when computing the GCM between the feet and the ground and (ii) a volume penetration model for all other possible contacts (mainly between the biped body and flying projectiles, see Experiment 6). They are both described in Appendix G.5.

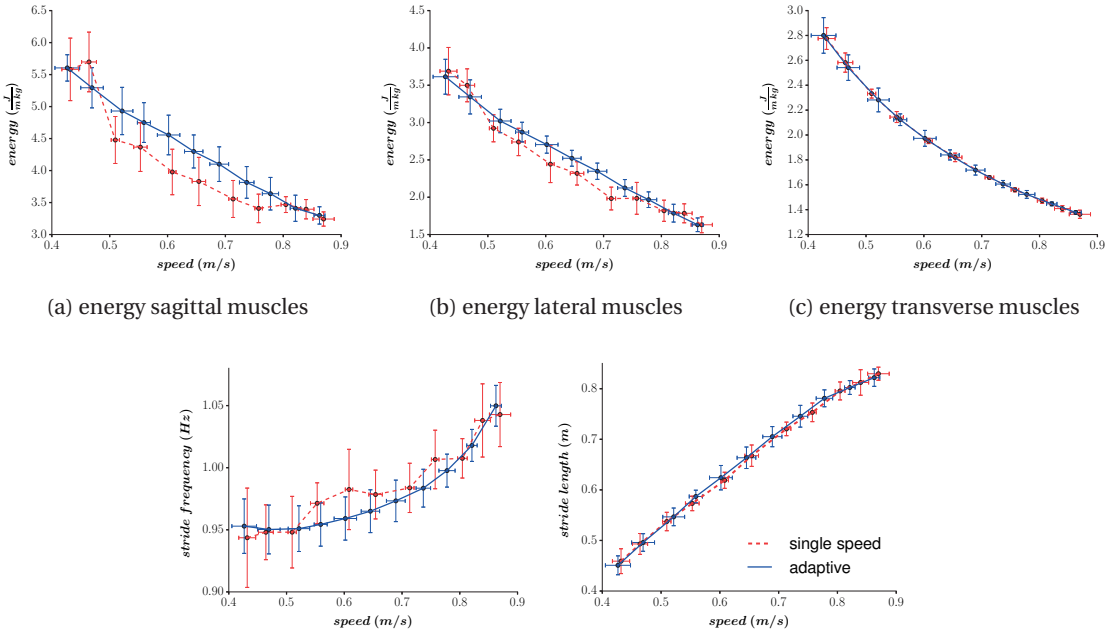
Our simulation environment used a fourth order Runge-Kutta integration scheme with a $250\,\mu s$ time step (i.e. 16 evaluations for $1\,ms$) to compute the dynamics model of the robot, actuators, GCM, etc. The controller sampling frequency was equal to $1\,ms$. When tested on a quad-core Intel(R) Core(TM) i7-4790 CPU, 3.6 GHz and 16 Go RAM (using a single core), an average time of $307\,ms$ was required to simulate $1\,s$.

9.4 Towards a single controller for a large range of forward speeds

The controller developed so far is capable of walking straight in a 3D simulation environment. In this section, this controller is incremented to achieve forward speed modulation, through the development of four experiments. First, the gait main features are analyzed for a set of walkers optimized for a single speed. Then, the key parameters governing gait adaptation are studied. The controller is later incremented to generate speed adaptations and to investigate the resulting gait features. Finally, forward speed modulations are actually reported.

9.4.1 Experiment 1: gait features changing as a function of the speed

The evolution of the following gait features is analyzed, based on the forward speed: (i) the metabolic energy consumption, (ii) the stride frequency, and (iii) the stride length. To do so, eleven speed references are investigated, corresponding to the range $[0.4; 0.9]$ m/s with a step of $0.05\,m/s$. Ten optimizations are performed for each investigated speed, resulting in ten different sets of optimized parameters, due to the heuristic of the PSO algorithm (Kennedy and Eberhart, 1995). The resulting controllers are labeled as the *single speed* controllers. The mean and standard deviations of their metrics are displayed in Figure 9.5.



(a) energy sagittal muscles (b) energy lateral muscles (c) energy transverse muscles

(d) stride frequency (e) stride length

Figure 9.5: In panels (a), (b), (c), the metabolic energy consumption per unit distance for the right leg is computed, for the muscles acting in the sagittal (panel (a)), the lateral (panel (b)) and the transverse (panel (c)) planes, respectively. Panel (d) shows the stride (i.e. two steps) frequency. Panel (e) shows the stride length. Ten controllers are optimized with no speed adaptation (labeled *single speed*) for each speed reference (Experiment 1). Ten other controllers are optimized (labeled *adaptive*), with the ability to adapt their speed on the whole speed range (Experiment 3). For each speed, the mean and the standard deviations (each time from the ten corresponding controllers) of the different measures are pictured.

The virtual metabolic energy consumption (Figures 9.5a, 9.5b and 9.5c) is computed for the right leg muscles, as detailed in Appendix G.1.3. Similar values are obtained for the left leg. As stated in Section 9.2.6, upper-body control is not the main focus of this contribution and barely contributes to the resulting gait. Therefore, its energy consumption is not studied.

The reported energy is actually normalized to the traveled distance. Interestingly, its value decreases with the robot forward speed. Sagittal muscles have the highest metabolic consumption, since they are the main muscles used to propel the body forward. However, the lateral muscle consumption is of the same order, due to the important efforts requested at the hip level during the stance phase. Surprisingly, transverse muscles energy consumption is also of the same order, while their only purpose was to keep the leg straight. The reason is that important gains are used for the corresponding PD controller, generating high co-contraction. A possible improvement would be to optimize these gain parameters.

Regarding the stride analysis (Figures 9.5d and 9.5e), an increase in the forward speed results both in an increase of stride frequency and length. This is coherent with human analysis: faster walking speeds usually correspond to faster walking frequencies and longer step lengths (Murray et al., 1966). For slow speeds, the evolution of the stride frequency is less significant than the one of the stride length. This indicates that the optimizer favors stride length modulation over frequency modulation for slow speeds.

9.4.2 Experiment 2: speed key parameters

Following the proximo-distal hypothesis (Daley et al., 2007), speed modulation is mainly performed by the leg proximal muscles, i.e. the ones close to the hip. In particular, the introduction of a CPG is useful for this purpose, since it modulates the locomotion by simple control signals (Ijspeert, 2008). This section investigates which control parameters could play a significant role in forward speed modulation.

Step frequency is directly related to the CPG frequency, which can be modulated using the time constant τ. Indeed, this value is proportional to the Matsuoka oscillators period (Taga et al., 1991). As reported in Section 9.2.8, the CPG frequency is optimized to match the gait resulting closed-loop frequency. Other potential parameters for speed modulation include the CPG amplitude output signals. They are controlled by the gains k_{HFL}, $k_{GLU,1}$, $k_{GLU,2}$, $k_{HAM,1}$, $k_{HAM,2}$, $k_{HAM,3}$ and k_{HAB} multiplying the CPG outputs (see Appendix G.4.1).

Moreover, faster speeds usually involve larger torso tilt, as reported in (Song and Geyer, 2012). Therefore, the target torso angles θ_{ref} (sagittal plane) and Ψ_{ref} (lateral plane) are also good candidates for modulating the forward speed. Finally, the lateral swing foot placement (being controlled by the parameter $\Lambda_{ref,h}$) might also be dependent on the speed. Therefore, all these parameters are studied for speed modulation.

The influence of these eleven key parameters on the walking speed was analyzed as follows. An optimization was performed for a single speed of 0.65 m/s, i.e. in the middle of the target speed range of Figure 9.5 ($[0.4;0.9]\,m/s$). Then, all the optimized parameters were frozen, except the eleven key parameters mentioned above. The speed range was discretized with a step of 0.05 m/s. For each target speed (including 0.65 m/s again), ten optimizations were performed, initiating the gait with the eleven key parameters corresponding to the initial speed

(0.65 m/s), before switching to new ones after four steps. The evolution of these optimized parameters is reported in Figure 9.6 (except for the target speed of 0.4 m/s, which did not produce suitable gaits in this experiment).

Intuitively, the evolution of most of these key parameters with forward speed can be approximated with polynomial functions, whose orders have to be properly selected to capture the curve without over-fitting. To do so, a model goodness-of-fit analysis using the sum of squared values of the prediction errors (Smith and Rose, 1995) was performed, as detailed in Appendix G.6.

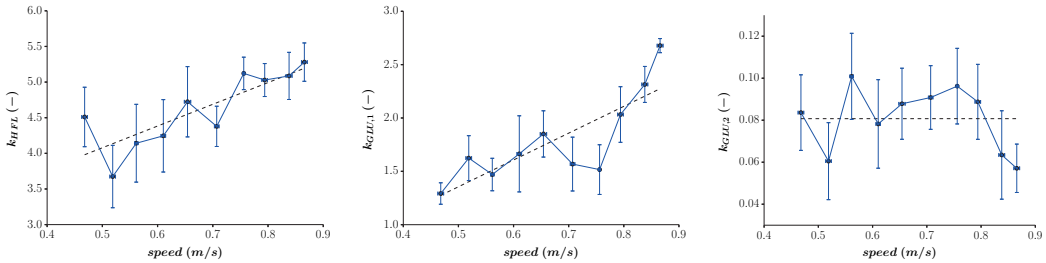
Resulting p-values are presented in Table 9.1. The corresponding null hypothesis is that the model fits the data. Its rejection (i.e. too small p-value) indicates an overall lack of fit regarding the order selected for regression. Fixing and arbitrary threshold to 0.1, the lowest order with a p-value exceeding this threshold was selected as being appropriate for the fit. This is a less strong analysis than rejecting the opposite null hypothesis, but is considered to be sufficient to design the control rules.

Table 9.1: This table reports the polynomial approximations of orders 0, 1 and 2 of the data provided in Figure 9.6, based on the least square errors. Each p-value is then computed as detailed in Appendix G.6. The first order with a p-value larger than 0.1 is then selected (grey cells).

	order 0	order 1	order 2	selected
τ	0	0.002	**0.968**	2
k_{HFL}	0	**0.211**	0.218	1
$k_{GLU,1}$	0	0.002	0.015	∅
$k_{GLU,2}$	**0.115**	0.099	0.293	0
θ_{ref}	0	**0.463**	0.649	1
$k_{HAM,1}$	0	**0.32**	0.517	1
$k_{HAM,2}$	0	0.022	**0.169**	2
$k_{HAM,3}$	0	**0.146**	0.528	1
Ψ_{ref}	**0.2**	0.159	0.727	0
k_{HAB}	0	0.028	**0.162**	2
$\Lambda_{ref,h}$	0	0.063	**0.958**	2

Interestingly, these results are close to the ones reported in (Van der Noot et al., 2015b). In this earlier contribution, similar graphs were obtained when restricting the walker to stay in the 2D sagittal plane, while exploring the evolution of a subset of six of the key parameters.

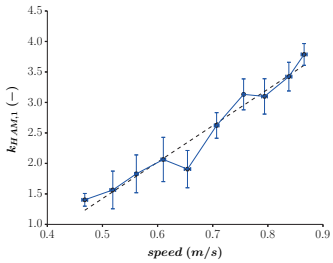
As expected, the time constant τ decreases (and so the frequency increases) with higher speeds. This correlation obeys a parabolic trend, while we reported a linear one in 2D (Van der Noot et al., 2015b). On top of that, the corresponding frequencies are larger in 3D than in 2D, for the same speed references. This is due to the lateral balance, which is easier to maintain with shorter step durations. Also, variations of τ are larger for higher speeds. This indicates that the optimizer favored step length modulation for slow speeds and step frequency modulation for higher speeds. This is coherent with the observations made in Experiment 1.



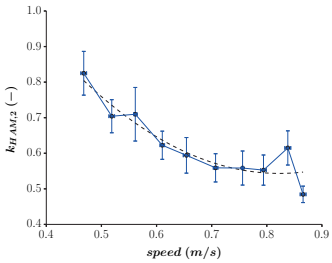
(a) Stimulation gain k_{HFL}

(b) Stimulation gain $k_{GLU,1}$

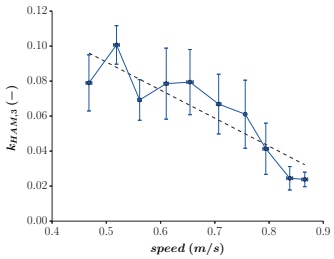
(c) Stimulation gain $k_{GLU,2}$



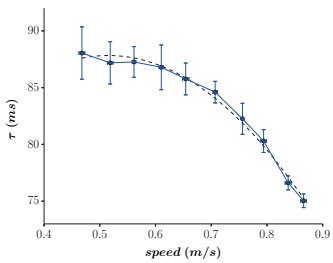
(d) Stimulation gain $k_{HAM,1}$



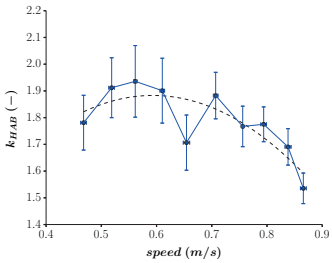
(e) Stimulation gain $k_{HAM,2}$



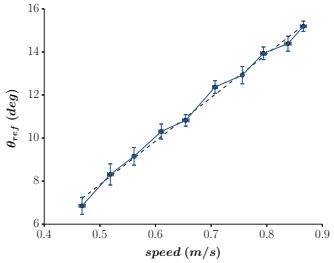
(f) Stimulation gain $k_{HAM,3}$



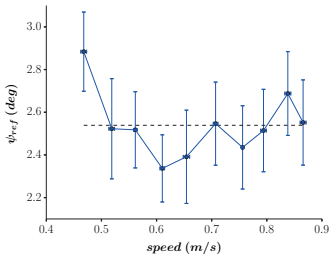
(g) CPG time constant τ



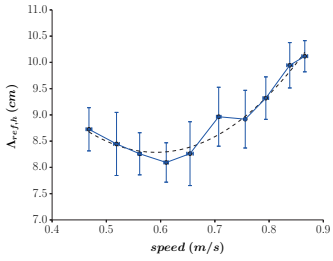
(h) Stimulation gain k_{HAB}



(i) Sagittal torso reference θ_{ref}



(j) Lateral torso reference Ψ_{ref}



(k) COM reference $\Lambda_{ref,h}$

Figure 9.6: Results of Experiment 2: ten optimizations are performed for each target speed (from 0.45 m/s to 0.9 m/s with an interval of 0.05 m/s).The actual speed of each solution is measured, along with the optimized value of the eleven open key parameters. For each target speed, we gather the ten optimization final results, reporting their mean and standard deviations. For graph legibility, the error bars represent half of the standard deviations. Dashed lines correspond to the polynomial approximations whose order is computed in Table 9.1, using the minimum mean square error method.

During the stance phase, $k_{GLU,1}$ and $k_{HAM,1}$ were both recruited to bring the torso back to its reference inclination after foot strike. This requires higher stimulations at higher speeds, due to larger inertia effects and strike impacts. This explains why these gains increase with higher speeds. Note that Table 9.1 reports that the polynomial fits did not reach significance for $k_{GLU,1}$. Since this parameter is redundant with $k_{HAM,1}$, this was considered to be not critical. For this parameter, we arbitrary chose a polynomial approximation of order 1. In the lateral plane, k_{HAB} is sightly larger in the middle of the speed range, indicating a stronger torso lateral stabilization for the corresponding speeds.

During the early swing phase, hip flexion increases for higher speeds. Consequently, the HFL muscles receive higher stimulations (with k_{HFL} increasing) while their antagonist muscles HAM get lower stimulations (with $k_{HAM,2}$ decreasing). In late swing, $k_{GLU,2}$ and $k_{HAM,3}$ are used to favor leg retraction, which reduces the walking speed. This explains why $k_{HAM,3}$ decreases. However, no significant modulation is observed for $k_{GLU,2}$, probably due to its redundancy with $k_{HAM,3}$. Globally, the CPG output modulation conveys similar conclusions as the ones we drew in the 2D case (Van der Noot et al., 2015b).

Regarding reflexes, the torso sagittal lean angle reference θ_{ref} increases linearly with speed, as in (Van der Noot et al., 2015b). Its lateral reference Ψ_{ref} however does not display a significant modulation, due to its high variance. Finally, the COM reference $\Lambda_{ref,h}$ driving the lateral swing hip is minimal in the middle of the speed range. This is coherent with the k_{HAB} evolution. Indeed, a higher k_{HAB} generates a higher momentum, accelerating the COM towards the swing leg (Patla et al., 1999). To counter it, the swing foot must be placed further away, inducing a smaller $\Lambda_{ref,h}$.

9.4.3 Experiment 3: a single controller for the whole speed range

The controller design can now be further extended to generate any forward speed in the $[0.4;0.9]\,m/s$ range. The eleven key parameters studied in Experiment 2 are replaced by polynomial functions whose order is chosen according to Figure 9.6 and Table 9.1 (except for $k_{GLU,1}$). Because the modulation of $k_{GLU,2}$ and Ψ_{ref} are actually of order 0, the corresponding parameters are constants. The speed modulation is then fully achieved with nine parameters: seven CPG parameters and two reflex parameters, as a function of the target speed (see Table G.2 in Appendix G.4). The four initial steps are performed with a speed reference v_{ref} set to $0.65\,m/s$, in order to achieve walk initialization. Then, v_{ref} can be changed to any value in the speed range, at any moment in the gait. This high-level control is depicted in Figure 9.1.

New optimizations were thus performed with the whole range of forward speed being embraced within a single trial. More specifically, eleven target speeds were selected (from 0.4 to $0.9\,m/s$ with a step of $0.05\,m/s$). Then, the same optimization process as the one described in Section 9.2.8 was performed to find the whole parameters set (including the coefficients capturing the modulation of the nine parameters changing as polynomial functions of the forward speed). More precisely, each optimization received this whole parameter set and a

range of target speeds v_{ref} to test (see Table G.2). The resulting fitness value was computed as the average of all the fitness functions of each tested target speed. This co-optimizes all the parameters within a single optimization, leading to more efficient gaits and larger speed ranges than those presented in Figure 9.6.

Ten heuristic optimizations were performed using this approach. They resulted in ten different sets of optimized parameters. The ten corresponding optimized controllers (called *adaptive controllers* and capable of reaching any forward speed in the $[0.4; 0.9]\ m/s$ range) were evaluated similarly to the so-called *single speed controllers* (i.e. controllers optimized for a single speed) from Experiment 1 (see Figure 9.5).

Since no parameter was optimized in the transverse plane, the corresponding energetic consumption was similar for the single speed controllers and the adaptive ones. In the other planes, the single speed controllers turned out to be more efficient than the adaptive ones. However, given that the adaptive controllers were optimized for a large range of speeds in a single shot and not tuned for a precise gait, this small pay-off regarding energetic cost seems a reasonable price to pay. Regarding step size analysis, the single speed controllers favor higher frequencies and shorter steps than the adaptive ones. However, these differences are rather small.

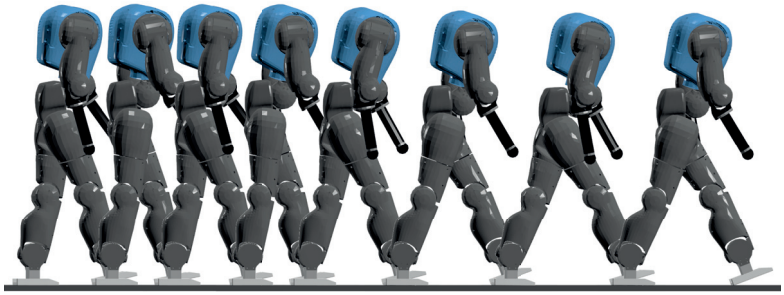
The standard deviations in Figure 9.5 are usually larger for the single speed controllers than for the adaptive ones. This indicates that the gaits (and underlying parameter sets) resulting from different optimizations are more similar when optimizing the whole range of forward speeds in a single trial. Globally, the sagittal energetic consumption and the step frequency display the highest deviations (relative to their respective ranges) between different optimizations. However, the global evolution of all these features with the speed remains close between the different optimization runs. So, while in principle there could have been multiple local minima in the search space, the optimizations tended to converge to similar optimal parameter sets and resulting gaits.

9.4.4 Experiment 4: forward speed modulation

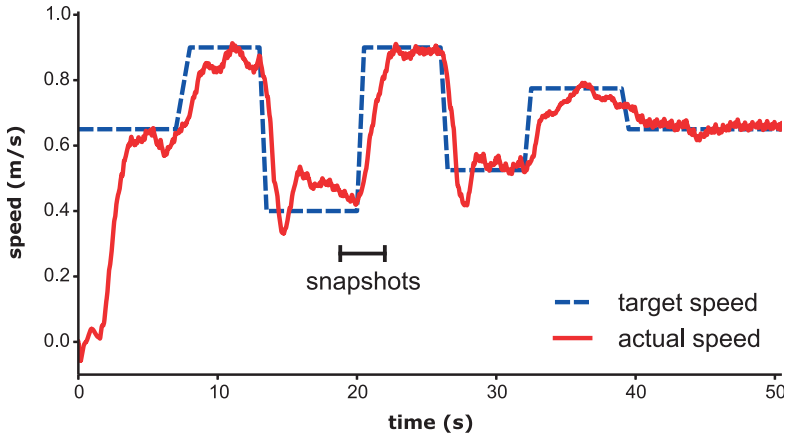
Among the adaptive controllers of Experiment 3, we select one of them and refer to it as the *reference controller*. In the rest of this contribution, we only report results that were obtained with this controller (i.e. corresponding to the same set of optimized parameters in the whole contribution).

The forward speed of the robot can be controlled on-line by adapting the speed reference v_{ref}. The speed modulation achieved with the reference controller on COMAN is visible in Figure 9.7.

In this experiment, the target speed is modulated in the full range, i.e. from 0.4 m/s to 0.9 m/s. The resulting speed (post-processed with a running average of 1 s) can follow this reference with accelerations up to $\pm 0.25\, m/s^2$. This represents less than two strides to go from one speed extremum to the other.



(a) Forward speed modulation



(b) Target speed tracking

Figure 9.7: Panel (a) pictures snapshots of an experiment where the robot forward speed is modulated. Panel (b) displays the tracking of the target speed v_{ref} (dashed line), where the robot actual forward speed (solid line) is post-processed with a running average of 1 s. The time interval during which the snapshots of panel (a) are taken is also displayed.

9.5 Comparisons to an inverted pendulum controller and to human data

The gait obtained from this neuromuscular controller can be compared to both human data and to gaits resulting from more traditional controllers, typically using inverse kinematics or dynamics transformations to compute position or torque references at the joint level (Fitzpatrick et al., 2016). Therefore, the gait of our reference controller is compared to the one resulting from a more traditional linear inverted pendulum (LIP) controller and to human data. These comparisons are performed on kinematics and dynamics data in steady state. Correlations between our muscles activations and surface electromyography signals (EMG)

extracted from human data are also reported. Finally, comparisons to the LIP-based controller are further extended by analyzing the energetic consumption.

9.5.1 Experiment 5: steady state gaits comparisons

Among the controllers relying on inverse modeling, we selected the one reported in (Faraji et al., 2014b). In that paper, a LIP-based torque controller could achieve gait modulation on the simulated COMAN. Using the same embodiment as ours offers to make direct comparisons with our own results (labeled *neuromuscular*). Importantly, this LIP-based controller generates slower gaits than the ones obtained with our neuromuscular model. Therefore, these comparisons are not ideal but remain valuable to provide a benchmark comparing our controller to more traditional approaches.

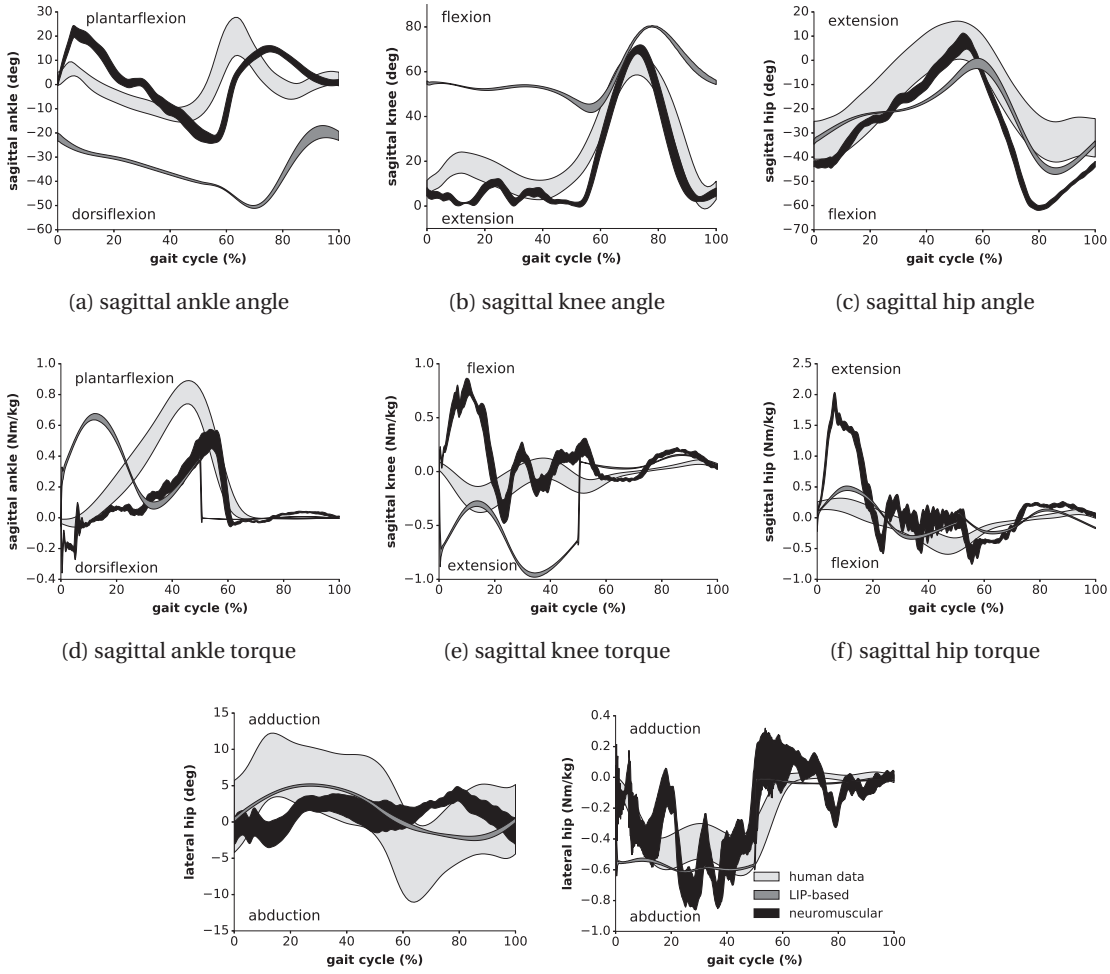
To compare these results with human measurements, we use the data from (Bovi et al., 2011). In that contribution, measures were performed on twenty adult subjects. This includes the temporal evolution of joint positions, torques, ground contact forces and EMG signals. We selected the data set with subjects walking at their natural (i.e. unconstrained) speed.

The average speed of the twenty adult subjects in (Bovi et al., 2011) was equal to $71.36\,\%BH/s$, where BH stands for body height. Considering that COMAN height would be close to $1.06\,m$ if it had a head, this corresponds to a speed of $0.75\,m/s$. Therefore, data for the neuromuscular controller was extracted from our reference controller walking with this reference speed. The LIP-based controller of (Faraji et al., 2014b) is not capable of reaching such a high speed. Consequently, the data presented from its resulting gait were obtained when walking close to its maximal speed, i.e. $0.31\,m/s$. It should also be noted that the LIP-based controller does not include a model of the electrical actuators, therefore bypassing the noise component introduced in Section 9.3.2. The following sections report different measurements performed on this experiment.

9.5.2 Kinematics and dynamics

The position and torque profiles extracted from Experiment 5 are displayed in Figure 9.8, where the data obtained with COMAN (i.e. the LIP and neuromuscular controllers) were averaged over twenty consecutive gait cycles (right leg). We computed the cross-correlation coefficient between each controller gait and the human data shifted in time. More precisely, we tested 100 time shifts equally spaced between 0 % and 100 % of the gait cycle. Here, we report the maximum of these cross-correlation coefficients, namely R and the corresponding time shifts Δ in percent of stride (Wren et al., 2006).

The sagittal joint kinematics globally shows good matching for the neuromuscular model (ankle: $R = 0.8, \Delta = -9\%$; knee: $R = 0.95, \Delta = 0\%$; sagittal hip: $R = 0.97, \Delta = 0\%$), although this is lower for the ankle than the hip and knee. This might be due to the rigid foot used on our model, different from the human one. Indeed, in (Colasanto et al., 2015), replacing the robot



(a) sagittal ankle angle (b) sagittal knee angle (c) sagittal hip angle

(d) sagittal ankle torque (e) sagittal knee torque (f) sagittal hip torque

(g) lateral hip angle (h) lateral hip torque

Figure 9.8: Kinematic and dynamic profiles of Experiment 5: the human data from (Bovi et al., 2011) (natural speed) is compared to our neuromuscular controller (0.75 m/s) and to the LIP-based controller (0.31 m/s) from (Faraji et al., 2014b). The averages of the different measures are displayed over one gait cycle (starting at right foot strike), augmented by their standard deviations (shaded areas).

rigid foot by a model of a human prosthesis led to more robust gaits. This is a possible future improvement for our experiments. The lateral hip kinematics corresponds to a low correlation ($R = 0.57, \Delta = -35\%$). However, the corresponding human motion is rather small and displays a large variance. Therefore, this low correlation is more difficult to interpret.

The correlations obtained with the LIP-based controller are systematically lower than with the neuromuscular controller, in the sagittal plane (ankle: $R = 0.32, \Delta = -40\%$; knee: $R = 0.87, \Delta = -6\%$; hip: $R = 0.93, \Delta = 0\%$) and better in the lateral plane (hip: $R = 0.93, \Delta = -5\%$). In particular, there is a large offset in the sagittal ankle and knee angles. This behavior (bended knee walking)

is typical of most humanoid gaits. The main reason is usually related to the deterioration of their controllers in configurations involving a singularity (Kurazume et al., 2005).

Interesting observations can also be reported from the torque cross-correlations. For the neuromuscular controller, the matching is good for the sagittal ankle and lateral hip joints, but not for the two other joints (ankle: $R = 0.92, \Delta = -5\%$; knee: $R = 0.24, \Delta = 82\%$; sagittal hip: $R = 0.53, \Delta = -3\%$; lateral hip: $R = 0.89, \Delta = 5\%$). The ankle plantarflexion is also of smaller magnitude. As previously mentioned, this might also be due to the lack of compliance in the foot being simulated.

The lower correlations for sagittal knee and hip are also observed in (Geyer and Herr, 2010). Human knee torque mainly oscillates around the zero axis during the stance phase. This is also the case in our model, although this oscillation is like in anti-phase. In Figure 9.8b, a small knee flexion is observed after strike, only for human data. To prevent from collapsing, humans thus apply an initial extension torque. In our model, the opposite happens: heel strike is followed by a slight knee over-extension, counteracted by a flexion torque. Regarding the sagittal hip, the main difference is the larger extension torque after strike, to prevent the torso from collapsing. However, it should be noted that other contributions reported human data displaying a similar large extension torque (Zelik and Kuo, 2010; Riener et al., 2002). This is likely highly dependent on the location of the hip center of rotation, which might also explain our own results. Yet, these bumps usually do not exceed $0.8\,Nm/kg$, indicating that our first hip reaction is above any human data.

The LIP-based controller torques shows similar correlations with human data, except for the sagittal ankle (ankle: $R = 0.89, \Delta = 29\%$; knee: $R = 0.71, \Delta = -19\%$; sagittal hip: $R = 0.66, \Delta = -1\%$; lateral hip: $R = 0.98, \Delta = 5\%$). The ankle torque in the sagittal plane shows a large phase shift regarding the peak in the stance phase. This is due to the lacking heel-toe motion and toe push-off. The lower variances can be explained by the lack of modeling of the motor dynamics and simulation noise.

Figure 9.9 shows the vertical ground reaction forces (GRF) measured during the same experiments. In particular, human data displays an M-shaped pattern, i.e. a well-known feature of human walking gaits. In contrast, the LIP-based controller exhibits a nearly flat profile during its stance phase, and initiates its swing phase earlier. On the contrary, the neuromuscular controller stance phase is better aligned with human data and displays oscillations in the GRF amplitude. However, the corresponding pattern differs from the human one. This discrepancy is probably due to the use of rigid feet in our experiment (and so to the lack of damping at strike impact), in contrast to human feet. Other possible reasons include the lack of toes, the foot length being shorter than the human one, and the knee over-extension issue previously mentioned.

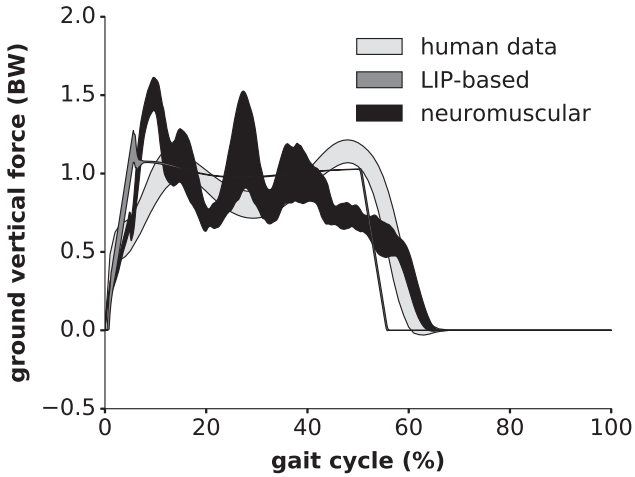
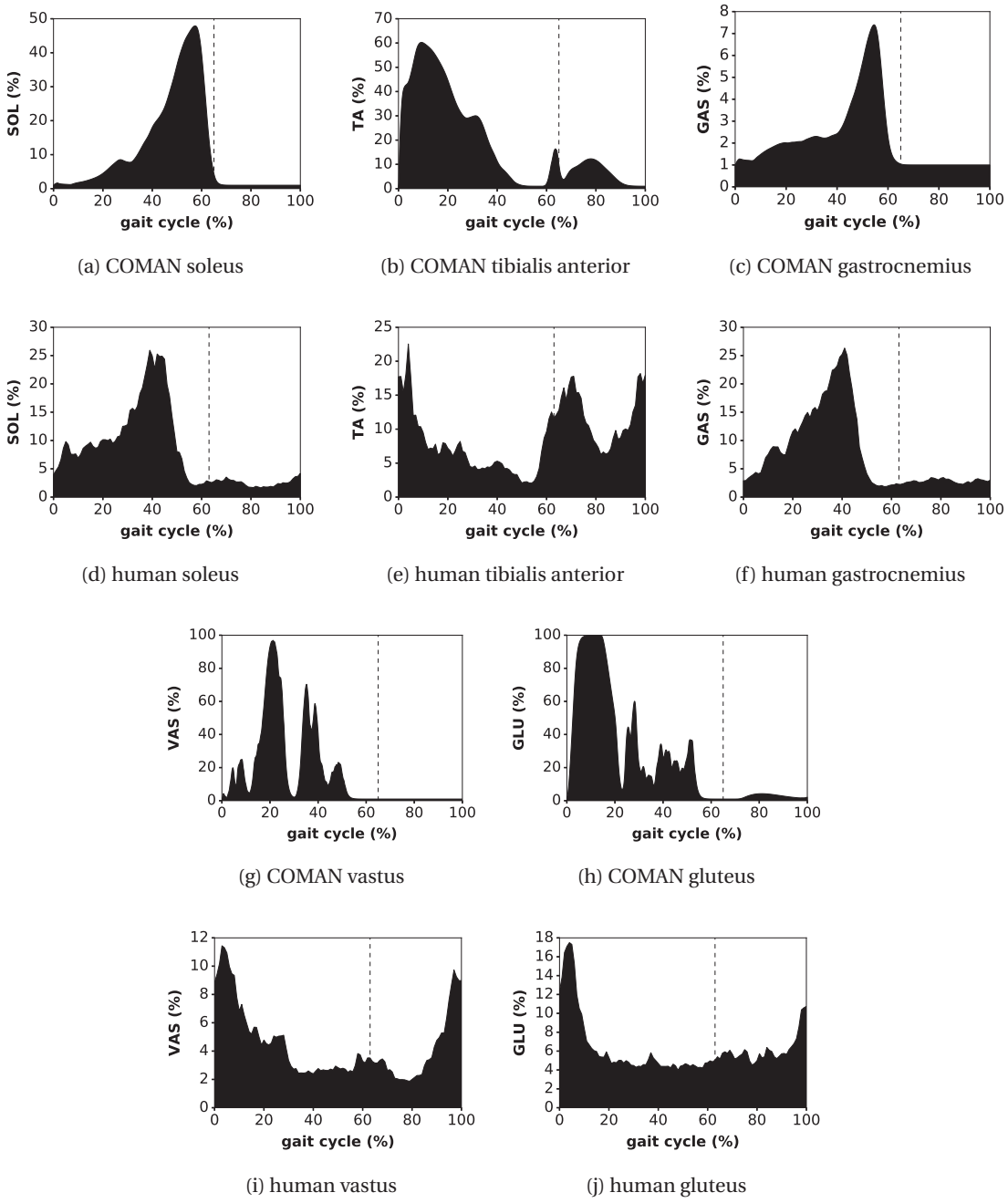


Figure 9.9: Vertical GRF profiles of Experiment 5 (right leg), normalized to the body weight (BW). Both the LIP-based and neuromuscular data are post-processed with a running average of 50 ms.

9.5.3 Muscle activations

Similarly to (Geyer and Herr, 2010), activations controlling the virtual muscles (neuromuscular controller) can be compared to real human EMG signals. Figure 9.10 reports this comparison for the following muscles: (a-d) soleus, (b-e) tibialis anterior, (c-f) gastrocnemius medialis, (g-i) vastus medialis, and (h-j) gluteus maximus.

The SOL and GAS muscle groups feature high cross-correlations coefficients, although with a significant phase shift ($SOL: R = 0.96, \Delta = -14\%$; $GAS: R = 0.96, \Delta = -13\%$). This shift corresponds to the one of Figure 9.8 for the joint being controlled by these two muscles, namely the sagittal ankle. Once again, this might be related to the lack of compliance in the foot, affecting the push-off phase. Correlations for the other muscles are typically lower ($TA: R = 0.69, \Delta = 58\%$; $VAS: R = 0.7, \Delta = -15\%$; $GLU: R = 0.76, \Delta = -3\%$). The stance activations are usually displaying a reasonable matching. During swing however, our virtual muscles are nearly silent because the legs rely on ballistic motion. This is not the case in the reported human measurements.



(a) COMAN soleus (b) COMAN tibialis anterior (c) COMAN gastrocnemius

(d) human soleus (e) human tibialis anterior (f) human gastrocnemius

(g) COMAN vastus (h) COMAN gluteus

(i) human vastus (j) human gluteus

Figure 9.10: Muscle activation profiles of Experiment 5: the activations obtained with COMAN (neuromuscular controller) are compared to EMGs measured on walking humans (Bovi et al., 2011). Due to the high variances of these signals, only their average is reported. The dashed line reports the transition from stance to swing.

9.5.4 Energetic consumption

In order to compare the energetic consumption of the neuromuscular controller to the LIP-based one, the square of the joint torques are integrated over one gait cycle. Figure 9.11a reports the different joint contributions for the right leg (the left leg results are identical). As indicated in Section 9.2.6, the upper-body motion barely contributes to the gait and is therefore not included in this analysis. In contrast to the previous analyses, the measurements were performed with the neuromuscular controller over its whole range of forward speeds. The gaits resulting from the neuromuscular controller are compared to the highest speed ($0.31\,m/s$) obtained with the LIP-based controller (i.e. same gait as in Figure 9.8).

Globally, the neuromuscular controller displays lower torque profiles than the LIP-based one, when walking slower than $0.64\,m/s$. As expected, the LIP-based controller recruits large torques at the knee level, due to the fact that this joint stays bended during the whole stance phase. The neuromuscular model, however, recruits smaller knee torques, but requires much higher torques at the sagittal hip joint (increasing with speed). This is coherent with the observations reported in Figure 9.8.

Torques produced by the ankle in the sagittal plane are also far less important with the neuromuscular controller, especially at slow speeds. The hip torque in the lateral plane are larger with the LIP-based model. Finally, the remaining joints torques are negligible. In particular, the high virtual metabolic energy consumption of the transverse hip (see Figure 9.5c) does not translate in higher torques.

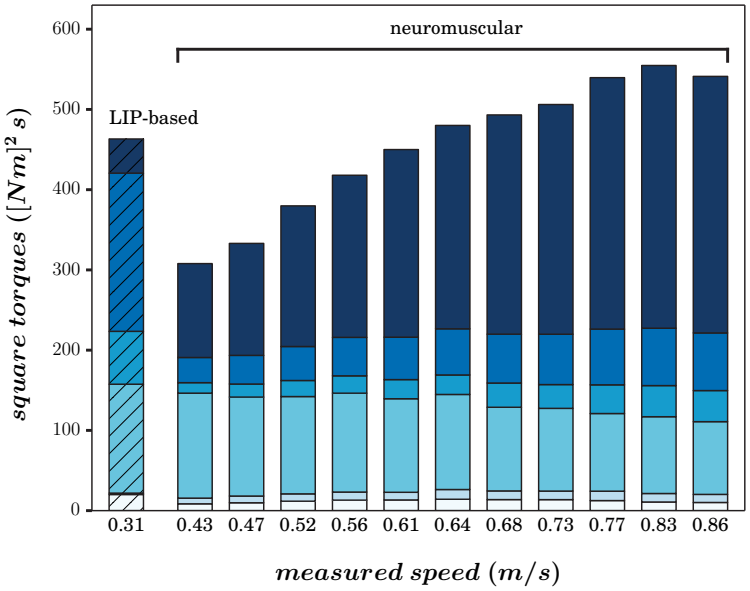
However, this analysis did not take the traveled distance into account. In Figure 9.11b, the same results are displayed, with a normalization by the stride length. Interestingly, the total square torque for the neuromuscular model is quite constant as a function of the forward speed. In particular, the increase in the sagittal hip torque is compensated by the extra traveled distance. This analysis strongly penalized the LIP-based controller since its normalized sum of square torques is about more than twice larger than the one of the neuromuscular controller.

9.6 Gait robustness

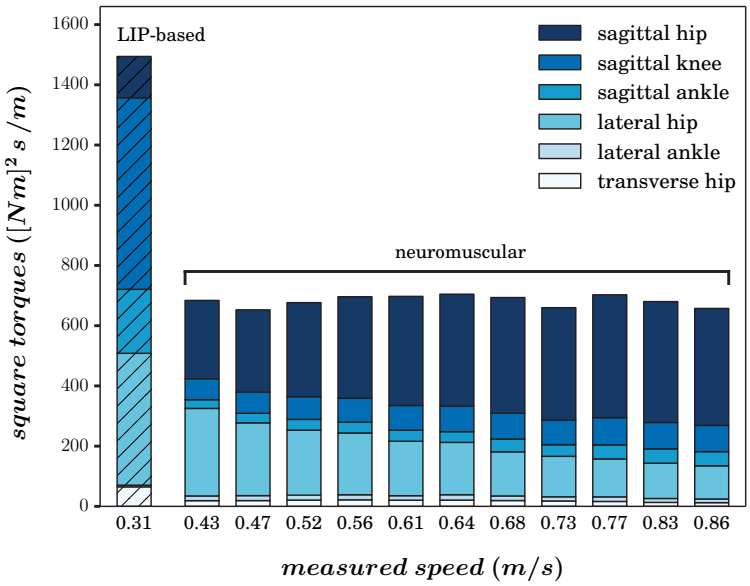
The following section reports experiments with the robot walking blindly (i.e. with no perception of its environment), using the reference controller. Its robustness was tested against external pushes, stairs, slopes and irregular grounds (on top of the simulator noise). During all these experiments, no parameters modulation was applied to the controller.

9.6.1 Experiment 6: resisting to pushes

First, the following experiment was performed. COMAN received random pushes on the torso when walking at different speeds. These pushes were applied with a magnitude between $0\,N$ and $30\,N$ during $0.2\,s$ in the transverse plane. Ten pushes were applied with a time interval



(a) square torques per gait cycle

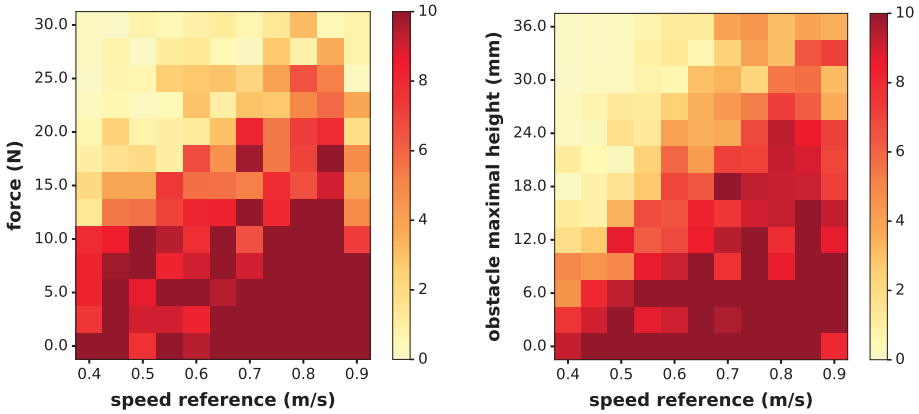


(b) square torques per gait cycle, divided by traveled distance

Figure 9.11: Estimate of the energetic consumption of both controllers tested in Experiment 5. Panel (a) reports the sum of square of the joint torques for the LIP-based controller (hatched) and the neuromuscular one (non-hatched), both integrated over one gait cycle, i.e. one stride. The measures were performed on the right leg at different speeds, and averaged over 20 gait cycles. The contributions of each joint correspond to different colors (see legend). Panel (b) displays the same result, normalized by the distance traveled during one gait cycle.

randomly selected between 5 and 6 s. Each push orientation in the transverse plane was randomly selected in the $]-\pi;\pi]$ interval (i.e. all possible directions selected with an equal probability). Robustness was quantified by counting the number of pushes the robot could sustain without falling.

This result is reported in Figure 9.12a, for the $[0.4;0.9]$ m/s speed reference range (with a discretization of 0.05 m/s). Globally, higher speeds can resist higher pushes. The only exception is the maximal speed (i.e. reference of 0.9 m/s), which was less stable. Indeed, less stable gaits were usually obtained for the extrema of the tested speed range.

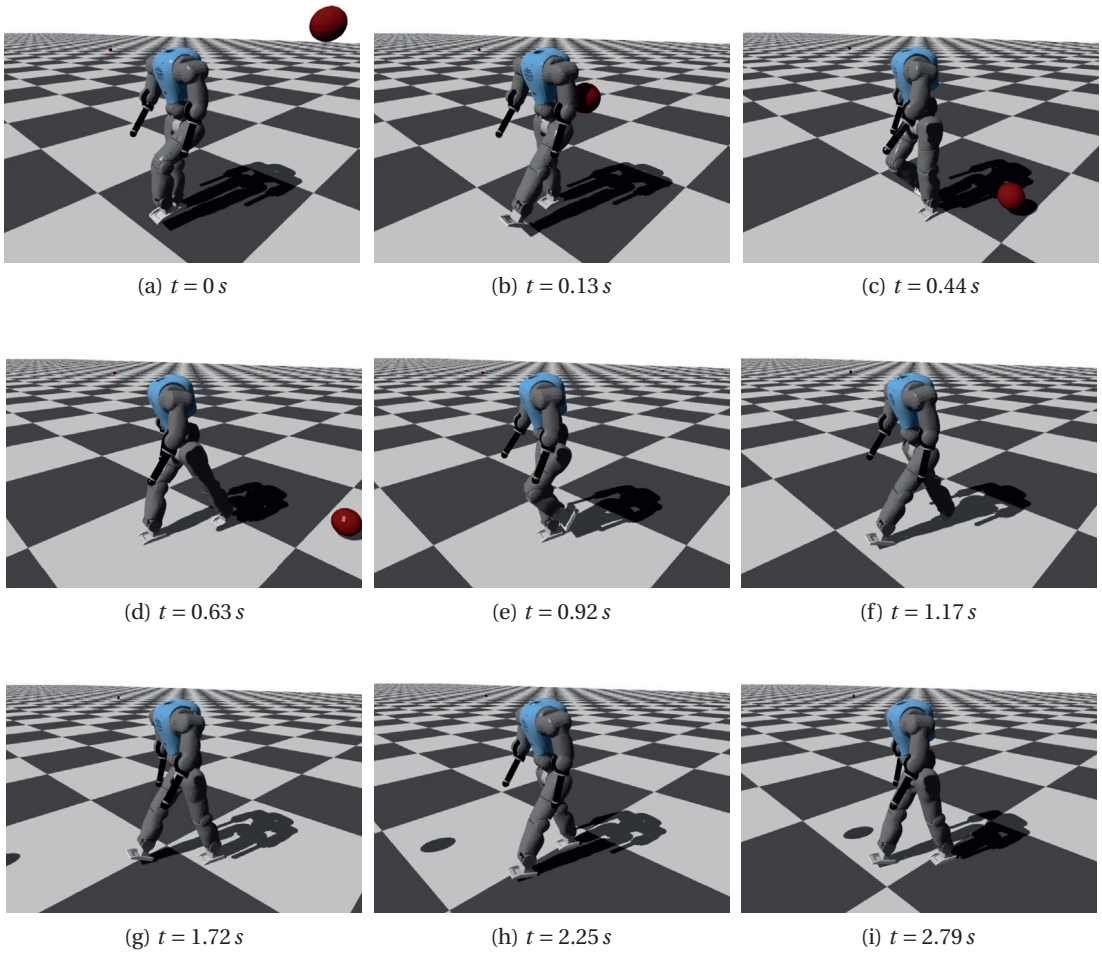


(a) pushes on the torso

(b) uneven ground

Figure 9.12: For the whole spectrum of speed references, COMAN faced two kinds of external disturbances. In panel (a), pushes were applied on its torso (Experiment 6). The color map represents the number of pushes the robot resisted (averaged over five runs) before falling, as a function of the pushes amplitude. In panel (b), COMAN was walking on the irregular ground displayed in Figure 9.17 (Experiment 9). The values of the corresponding H_i heights were randomly selected in the range whose maximum value is reported on the vertical axis. The color map represents the forward distance (i.e. along the x axis $[m]$) COMAN walked before falling (limited to 10 m and averaged over five runs).

Another illustration of the robot resistance was performed using flying balls, when COMAN was walking with the reference controller at a speed of 0.65 m/s. During this experiment, ten balls with a density of 750 kg/m^3 were thrown to it, while COMAN had to resist and continue walking. Some snapshots of this experiment are visible in Figure 9.13. In particular, after absorbing a ball push, the walker could recover its previous gait, thanks to the CPG entrainment (Ijspeert, 2008).



(a) $t = 0\,s$ (b) $t = 0.13\,s$ (c) $t = 0.44\,s$

(d) $t = 0.63\,s$ (e) $t = 0.92\,s$ (f) $t = 1.17\,s$

(g) $t = 1.72\,s$ (h) $t = 2.25\,s$ (i) $t = 2.79\,s$

Figure 9.13: COMAN receives an impact from a flying ball on the upper body. Despite the shock, the robot can recover its balance thanks to the neuromuscular controller. The different panels depict the action and provide the corresponding timing information. In particular, the impact happens in panel (b), while panels (h) and (i) show that the robot can recover its normal gait after shock absorption, thanks to the CPG entrainment. Indeed, after taking large steps due to the impact (see panels (d) and (f)), COMAN recovers its normal (i.e. shorter) steps.

9.6.2 Experiments 7 and 8: natural adaptation to stairs and slopes

Experiment 7 established the capacity of the robot to adapt to ascending and descending (small) stairs. This is presented in Figure 9.14, with the reference controller walking with a speed reference of $0.85\,m/s$. The corresponding stair is made of five ascending and five descending steps, each with a width of $50\,cm$ and a height of $2\,cm$. This performance is similar to the one of (Geyer and Herr, 2010), pending a scaling to our robot size. Interestingly, our controller can even adapt when its foot lands between two consecutive stair steps, as can be seen in Figure 9.14.

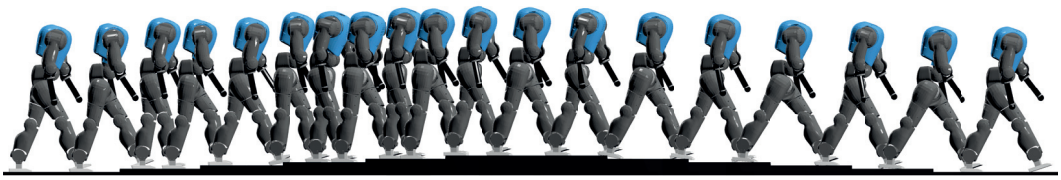


Figure 9.14: Snapshots from Experiment 7: COMAN walked blindly on an ascending and descending stair. Step length was automatically adapted to the environment, without changing the controller. At the end of the stair, COMAN retrieved its initial gait, thanks to the CPG.

Similarly, Experiment 8 tested the robot ability to adapt to ascending and descending slopes. This is presented in Figure 9.15, where COMAN walks blindly with a speed reference of 0.85 m/s on a flat ground before facing a rising slope of 2.58°.

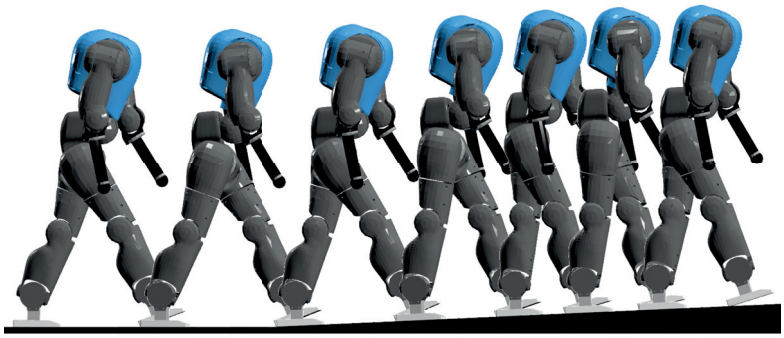
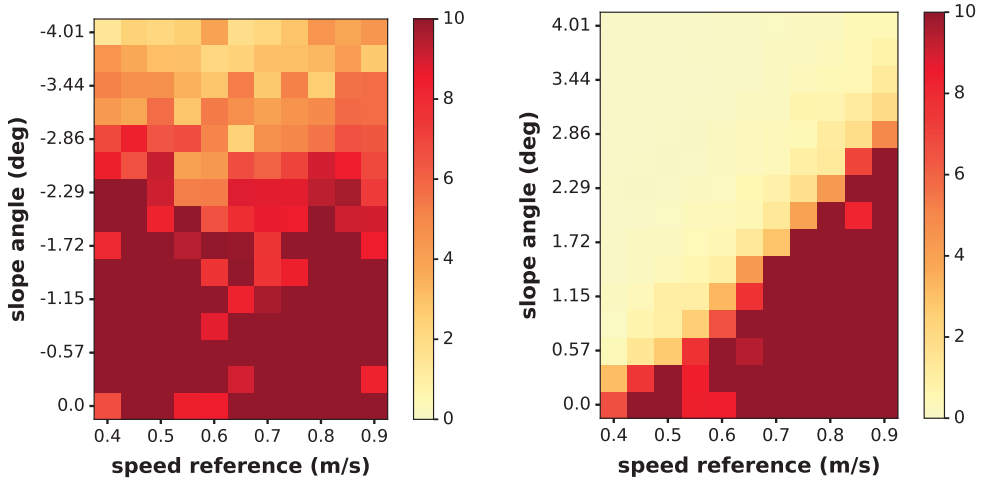


Figure 9.15: The figure displays snapshots of Experiment 8, where the robot faced a slope (here, 2.58°). It automatically adapted its step length, with no change in the controller.

Similar results were obtained on the whole speed range, as reported in Figure 9.16. There is no global trend for descending slopes. Generally, COMAN can walk on negative slopes with an angle smaller than −2.29° (−4%). For rising slopes, a clear correlation appears with the forward speed. As can be seen in Figure 9.15, the walker naturally decreases its step length (and so its speed) when climbing a positive slope. Therefore, a higher initial speed can withstand larger slopes. With its maximal speed reference, COMAN can climb slopes up to an angle of 2.58° (4.5%). This is similar to the results reported in (Geyer and Herr, 2010).

9.6.3 Experiment 9: natural adaptation to irregular grounds

In Experiments 7-8, the walker robustness was tested when facing uneven grounds with regular patterns (i.e. stairs and slopes). This experiment quantifies its robustness to irregular grounds. The description of the corresponding ground is presented in Figure 9.17. Different grounds can then be tested with randomly selected heights H_i.



(a) negative slopes

(b) positive slopes

Figure 9.16: Results from Experiment 8: for the whole spectrum of speed references, COMAN faced grounds with slopes of different angles (from 0° to 4° with a discretization of 0.29°, for positive and negative angles). The color map represents the distance traveled on the slope (in $[m]$) before a possible fall (limited to 10 m and averaged over five runs).

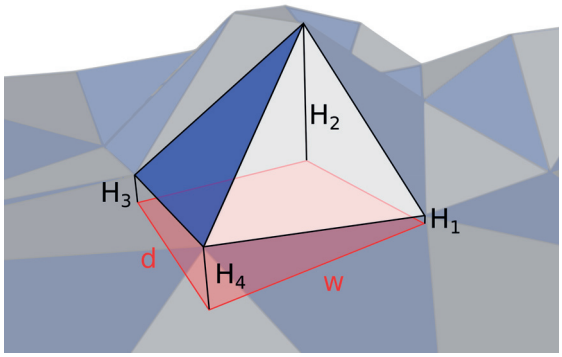
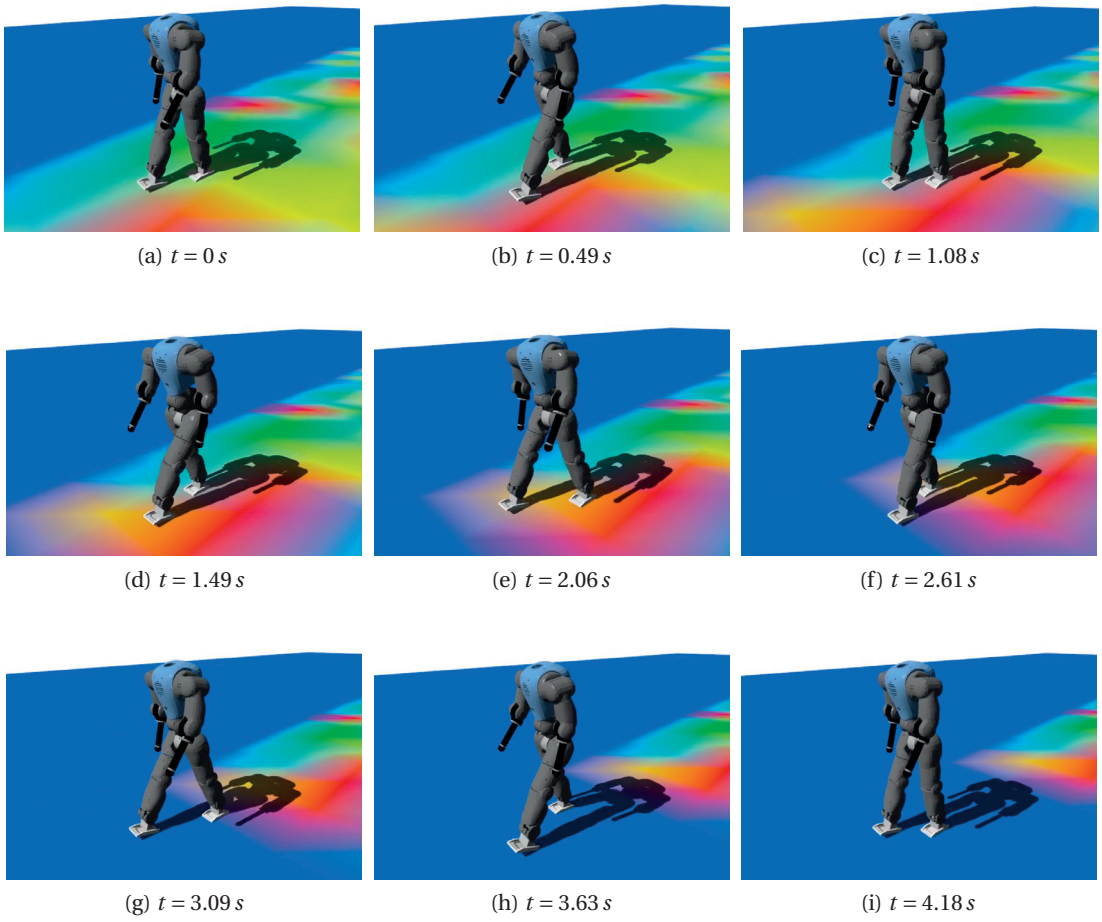


Figure 9.17: Description of the irregular uneven ground generated for Experiment 9. Each triangle composing the ground mesh is based on a rectangle of size $d \times w$ with four randomly selected heights H_i at its corners ($d = w = 50\,cm$).

In Figure 9.18, COMAN walks on this ground (with a speed reference of 0.65 m/s), where the H_i heights were randomly selected in a range of $[0;25]\,mm$. Figure 9.12b reports the result of this experiment over the whole speed reference range and for different maximum obstacle heights. Similarly to the results of Experiment 6, higher speeds produced more robust gaits, except for the maximum speed (0.9 m/s), intrinsically less stable.



(a) $t = 0\,s$ (b) $t = 0.49\,s$ (c) $t = 1.08\,s$

(d) $t = 1.49\,s$ (e) $t = 2.06\,s$ (f) $t = 2.61\,s$

(g) $t = 3.09\,s$ (h) $t = 3.63\,s$ (i) $t = 4.18\,s$

Figure 9.18: COMAN walks on the irregular ground described in Figure 9.17, with the H_i heights being randomly selected in a range of $[0;25]\,mm$. Higher ground heights are depicted with warmer colors. In particular, the step length is adapted as the result of the ground irregularities. See for instance the shorter step of panel (c), due to the increase of the ground height. Moreover, it can be seen that the foot landing in panel (e) is deported on the right of its normal trajectory, due to the higher ground on the left of the robot.

9.7 Discussion

The work presented in this contribution offers an alternative locomotion controller for humanoid robots. The controller can generate gaits across a range of speeds close to the normal human walking one, by recruiting virtual muscles controlled by CPG and reflex signals. By embracing the concept of limit cycle walking, it relaxes constraints inherent to more traditional locomotion controllers. In particular, singularity configurations like stretched legs can be reached, generating faster and more energetically efficient gaits.

9.7.1 Interest of the bio-inspired approach

While using (virtual) muscles might seem natural when working on real human models or on animation characters, it is less obvious for humanoid robots equipped with electrical actuators. This paper showed that using muscles as an intermediate layer offers several interesting properties: (i) the virtual muscles generate continuous torques, being smooth to track for the low-level torque controller; (ii) human-like gaits can be obtained by minimizing the metabolic consumption of these virtual muscles (see Section 9.2.8), in a way likely similar to what humans do; (iii) this configuration - being similar to the one of a human - provides the ideal framework for comparing our model to human data, including the level of muscle activations; and (iv) the walker benefits from the viscoelastic muscle properties, i.e. human-like joint impedance. Regarding this last point, the exact effects of the muscular viscoelastic properties still need to be quantified, which is a potential topic for follow-up work. Finally, note that minimizing the metabolic consumption of virtual muscles (point (ii)) is not a priori equivalent to minimizing the robot's electrical energy consumption. However, the same optimization tool could be used to minimize this electrical energy consumption (i.e. maximizing the actuators efficiency) by replacing the metabolic energy measure by the electrical consumption of the motors. Future work will explore the influence of this regarding the gait kinematics and robustness.

Experiment 5 further showed that it was possible to drastically reduce the joint torque contributions with the proposed method, in comparison to more traditional controllers. This could potentially lead to important energetic cost reductions during locomotion. However, this was tested on two very different speed ranges. More specifically, the highest speed of the LIP-based controller of (Faraji et al., 2014b) was close to the lowest one of our neuromuscular controller. Therefore, an alternative approach would be to use both of these controllers on the same robotic platform, pending the implementation of a transition mechanism as a function of the forward speed. In particular, the neuromuscular controller is likely more appropriate to quickly and efficiently reach a desired spot. A controller recruiting foot step planning would in contrast be more appropriate when accurate positioning is requested. Alternatively, the proposed neuromuscular controller could also be extended to generate slower walking speeds.

Last but not least, this approach is also advantageous regarding computational cost. A single iteration of our neuromuscular controller (i.e. CPG + reflexes + virtual muscles) requested an average time of 61 μs to be computed (on the same computer as the one reported in Section 9.3.2). This is more than 16 times faster than the COMAN controller sampling rate, namely 1 ms. In contrast, many existing locomotion controllers use demanding computations of inverse kinematics and/or dynamics. This often leads to critical issues to fulfill the real-time constraints.

9.7.2 Robustness to unperceived environments

Gait robustness is one of the major issues preventing robots from being used in unknown environments. In particular, many biped locomotion controllers require an accurate dynamic model of the robot, resulting in poor robustness when there are errors in this model. Other approaches, like the virtual model control proposed in (Pratt et al., 2001) require however no dynamic model of the robot to achieve robust gaits during blind walking.

Here, the blind walking experiments performed on the COMAN platform demonstrated impressive robustness when walking in perturbed environments. In particular, the viscoelastic muscle properties commanded by the combined action of the CPG and the reflexes could automatically adapt the gait to various perturbed environments. Importantly, this was achieved without changing a single parameter of the controller. A perfect knowledge of the environment was therefore not requested, which is a key advantage in order to bring humanoid robots in our natural day-to-day life. Using the CPG as a central element, the robot could return to its normal gait after perturbation. This was particularly outlined in Experiments 6-9.

The controller could be further extended to detect possible falls and trigger additional reaction primitives. In (Li et al., 2015), an energy-based fall prediction method is presented for this purpose. Similar strategies could likely allow the walker to withstand higher perturbations than the ones performed in the blind walking experiments.

9.7.3 Gait modulation

Motion diversity control (e.g. deliberate obstacle avoidance) might be easier to achieve with more traditional methods relying on inverse kinematics or inverse dynamics. However, similar motion diversity can also be found when using neuromuscular models. For instance, (Desai and Geyer, 2013) revisited the model of (Geyer and Herr, 2010) in order to control the swing leg placement. This model was further extended in (Song and Geyer, 2015a) to avoid obstacles by increasing the foot ground clearance or the step size. Similar performances can also be obtained with CPG modulations, as we reported in (Van der Noot et al., 2015b), with the objective to step over a hole.

In this contribution, we showed that the inclusion of a CPG could modulate the forward speed by adapting nine key control parameters as linear or quadratic functions of the target speed. This resulted in high speed variations, over a range close to the normal human one, when scaled to the robot size. Because both the step frequency and length are adapted, it provides full control of the foot step placement, in order to avoid small obstacles. However, a high-level controller (see Figure 9.1) modulating the CPG inputs to generate desired gait alterations was not explored and is a potential avenue for future developments.

9.7.4 Parallels with human locomotion

Experiment 5 showed that this controller could also be used to investigate models of human locomotion. This was examined through comparisons with human kinematics and dynamics measurements, as well as EMG signals. Our controller recruited Hill-type muscle models commanded by reflexes, and Matsuoka oscillators, which are components developed on a solid biological background. Our CPG network was divided into two parts: the "rhythm generator" neurons and the "pattern formations" ones. Using a similar two-level CPG biological architecture, (McCrea and Rybak, 2008) reproduced results observed in experiments of fictive locomotion with decerebrated cats. Our approach also followed the proximo-distal hypothesis which was verified by (Daley et al., 2007) on avian bipeds. In other words, muscles close to the hip mainly received feed-forward signals (i.e. from the CPG) while the distal muscles (being highly load-sensitive) received feedback activations (i.e. reflexes).

Using this structure, the modulations of the CPG frequency and amplitude, together with two reflex parameters, led to large forward speed variations and step modulation, as shown in Experiments 3-4. So, similarly to the work performed by (Taga, 1994), (Paul et al., 2005) or (Rossignol et al., 2006), this contribution also supports the assumption that CPGs could play a major role in human locomotion, at least for gait modulation.

Importantly, the recruitment of CPGs to control the walking of most vertebrates is widely accepted, but the neural circuitry generating human locomotion is still not entirely unveiled (Dzeladini et al., 2014). The work of (Geyer and Herr, 2010), further extended in (Song and Geyer, 2015a), obtained similar results as ours, although they implemented only reflex pathways (i.e. without CPG). Therefore, the recruitment of CPG networks during human locomotion remains a matter open to debate.

While many studies use a deductive approach to understand human locomotion (Lacquaniti et al., 2012), this contribution offers a synthesis approach to test hypotheses on human walking. In particular, this is potentially valuable to provide insights about neural and orthopedic disabilities, by understanding their effects on walking, and thus possibly contributing to develop new treatments. Yet, it is important to note that the musculo-skeletal model developed here is a high-level approximation of control principles found in human motor control, not an accurate computational neuroscience model.

Divergence with real human data could possibly lead to model refinements, with the purpose to better explain human locomotion mechanisms. For instance, the large torque peak experienced by the sagittal hip after foot strike could be reduced by the introduction of a stance preparation phase. Indeed, this lack of preparation resulted in an insufficiently damped impact and thus in a large forward torso tilt, as explained in (Geyer and Herr, 2010).

Non-sagittal leg control could also be improved by taking inspiration from human strategies. For example, humans use the hip internal rotation, even in straight walking. This advances the swing leg and increases the step length (Stokes et al., 1989). A possible improvement

of our controller would be to integrate this mechanism. Also, the hip lateral position could sometimes bring the swing leg too close to the stance one, resulting in possible collisions between the legs. In our experiments, this was sometimes observed at speed extrema and during perturbed walking. A first naive solution would be to increase the weight of the fitness stage favoring large lateral distances between both feet. However, this might reduce the range of achievable speeds. Another solution would be to increment the lateral hip swing control. However, this depends on the walker embodiment being used.

Muscles coordination during human locomotion is a complex task due to the large redundancy in the musculo-skeletal system (Ting et al., 2012). To solve this over-actuation problem, human motion control possibly relies on muscle synergies, i.e. on the covariation of muscle activities. Synergies virtually decrease the number of degrees of freedom (Aoi et al., 2010). In our work, muscle synergies are captured by two factors. First, the number of muscle groups (mainly inspired from (Geyer and Herr, 2010)) is much smaller than the actual number of human muscles. Second, some synergies are generated by our reflexes and CPG signals. For instance, the combined activation of the HAM and GLU muscles in early stance stabilizes the torso. Yet, other synergies could be explored, in particular if more muscles were added to the musculo-skeletal system.

The controller could also be tested on a model closer to the human morphology than COMAN. For instance, the human femoral joint is quite different from the robot hip joints. Similarly, feet closer to the human ones could by used on the robot. In (Colasanto et al., 2015), replacing the rigid feet of COMAN by compliant prostheses led to more robust gaits, when using similar neuromuscular control rules.

Computer graphics animation is another avenue for the development of such models, for example through the generation of motion and torque patterns incorporating biomechanical constraints (Wang et al., 2012). Similar neuromuscular models are not limited to humans but could possibly be extended to many biped creatures, as demonstrated by (Geijtenbeek et al., 2013) on an ostrich model.

9.7.5 Perspectives

As detailed in Section 9.7.4, the controller is valuable to better understand human locomotion and to investigate possible pathologies. However, significant differences with human data were reported. These divergences could be more thoroughly investigated to obtain gaits closer to human ones. This could be done by refining the muscuoloskelettal model and the neural controller rules, but also by using a model (instead of a robot) close to the real human morphology (e.g. with toes and some compliance in the segments).

Interestingly, the bio-inspired approach developed here could also be applied to different body types, and even to extinct species. For instance, computer simulations and biomechanical modeling are considered as some of the most rigorous methods to reverse-engineer the gait

of dinosaurs. By combining solid evidence like the morphology of their limb skeletons with external and muscular forces, it is thus possible to reconstruct physically plausible motions (Hutchinson and Gatesy, 2006). Therefore, neuromuscular controllers could possibly be adapted to theropod (i.e. bipedal dinosaur, like *Tyrannosaurus*) gaits.

All the tests performed in this contribution used a faithful simulation model of the COMAN platform (including its actuator dynamics and noisy torque sensing). Therefore, the controller has the potential to be tested on a real robotic device. Similarly to (Van der Noot et al., 2015a), this transfer would require some care regarding the dynamic non-idealities (e.g. impact, friction and backlash).

There is an increasing interest to bring humanoid robots out of the laboratories, as emphasized during the recent DARPA Robotics Challenge. However, biped locomotion remains an important challenge, as illustrated during the terrain task of this contest. Indeed, during the corresponding trials, only 2 of the 16 teams successfully completed the entire terrain task without requiring an intervention, so that the walking challenge for the finals had to be simplified (Johnson et al., 2016). The present contribution does not target DRC-like tasks, but rather studies the scientific question of exploring the benefits of human-like musculoskeletal systems, together with their control properties. This is scientifically interesting, but also potentially valuable for robotics locomotion since humans are still much better than humanoid robots to tackle complex terrains. Moreover, the leg stretching obtained using our approach would potentially offer to cross larger obstacles and to climb stairs with higher steps, in comparison to walkers displaying continuous knee bending.

While bipedal robots are currently far from the walking capabilities of real humans in terms of robustness and energy-efficiency, this contribution thus shows that neuromuscular controllers hold the potential to make a step towards this achievement. Indeed, the generated gaits are closer to the human ones, and so, more adapted to our surroundings. In the future, robots might be able to adapt to our environment, rather than us having to adapt our environment to the robot limited skills.

The natural extension of the forward speed modulation approach reported in this contribution is addressed in Chapter 10: the control of heading direction. In that next chapter, both the heading direction and the radius of curvature of the biped walker trajectory can be controlled during walking, allowing the biped to reach any point in a 3D environment and to navigate around obstacles.

10 Steering control in a 3D environment

Chapter 6 developed a neuromuscular controller recruiting both a central pattern generator (CPG) and reflexes to modulate the forward speed of COMAN. This was explored in a 2D simulation environment. Chapter 9 extended that contribution to 3D scenarios, with the inclusion of lateral balance control. By modulating a scalar input (i.e. the speed reference), these two chapters achieved speed modulation from 0.4 to 0.9 m/s, i.e. in a range similar to the normal human one once scaled to the size of the robot.

This chapter further extends the 3D results of Chapter 9 with new modules governing right/left steering. More precisely, the heading direction and the radius of curvature are both controlled by a new scalar input: the heading reference. Therefore, by adapting two scalar inputs, the controller developed in this chapter is capable of achieving on-line control of the robot forward speed and heading. These features are valuable to steer the robot in its environment, and thus to avoid obstacles.

10.1 Introduction

Nowadays, there is an increasing interest in bringing mobile robots in our everyday life. However, their mobility, usually different from ours, restricts them to move in dedicated environments. In contrast, humanoid robots are potentially adapted to move in environments designed for humans, since their body is very similar to the human one (Schaal, 2007). Moreover, their morphology offers the possibility to manipulate tools fitting human dexterity. Therefore, these tools do not require to be adapted to the robot needs, favoring potential co-operative work with humans (Fitzpatrick et al., 2016).

However, navigation in the unpredictable human world remains an important issue, as emphasized during the recent DARPA Robotics Challenge (Johnson et al., 2016). In particular, robot's locomotion skills are far from reaching the level of the human ones, therefore usually restricting them to move in controlled environments, such as laboratories. Most popular biped locomotion algorithms recruit the zero-moment point (ZMP) as an indicator of gait feasibility,

in order to guarantee dynamic stability at every moment during locomotion (Vukobratovic and Borovac, 2004). Many successful walking experiments were conducted using this indicator, for instance with ASIMO (Chestnutt et al., 2005) or with the HRP-2 platform (Kaneko et al., 2002).

Interestingly, these methods can offer a mathematical framework for postural control and steering, for example through appropriate footstep planner strategies recruiting inverse kinematics or dynamics (Faraji et al., 2014b). However, common drawbacks associated with most ZMP-based bipedal controllers include energy inefficiency, unnatural gait (low waist position, continuous knee bending, feet kept parallel to the ground, etc.), poor resistance to modeling errors or external perturbations and slow walking speeds (Sardain and Bessonnet, 2004b; Kurazume et al., 2005; Dallali, 2011).

In contrast, the limit cycle concept relaxes constraints inherent to the ZMP criterion, by focusing on time-averaged stability during walking, instead of local stability at every moment of the gait (Hobbelen and Wisse, 2007). Successful implementations of this concept include the (quasi-)passive walkers, resulting in efficient human-like gaits (McGeer, 1990; Collins and Ruina, 2005; Hobbelen et al., 2008). However, gait modulation is usually very limited, due to the lack of control parameters.

Bio-inspired controllers are emerging as a promising way to implement adaptive limit-cycle walking, by designing control strategies based on concepts identified in humans. In particular, the neuromuscular model developed in (Geyer and Herr, 2010) (and further extended in (Song and Geyer, 2015a)) can generate robust human-like walking through the recruitment of muscles controlled by reflexes. Experimental validations of this approach include the control a powered ankle-foot prosthesis (Eilenberg et al., 2010) and the locomotion of a humanoid robot in a 2D scenario, i.e. when assistance was provided to the lateral balance (Van der Noot et al., 2015a).

In contrast to inverse kinematics/dynamics approaches, the gait modulation and steering of these bio-inspired approaches remain challenging, due to the lack of straightforward mathematical framework. In (Desai and Geyer, 2013), the reflex rules of (Geyer and Herr, 2010) were extended to control the swing leg placement. Another avenue to modulate the gait is through the introduction of a central pattern generator (CPG). CPGs are neural circuits capable of producing rhythmic patterns of neural activity without receiving rhythmic inputs. They display valuable features among which the possibility to modulate locomotion with simple low-dimensional control signals (Ijspeert, 2008).

While locomotor CPGs were identified in many vertebrates, their involvement in human locomotion is still a matter open to debate (Minassian et al., 2017). In particular, human-like gaits can be achieved using computational models, both with and without CPG. Successful CPG implementations include the robust bipedal walking experiments of (Aoi and Tsuchiya, 2005) by means of nonlinear oscillators, the adaptation of a biped locomotion on uneven terrains using CPG modulation (Taga, 1994), and the development of a neuromuscular model recruiting a CPG as central element, in order to investigate the effects of a spinal cord injury

on locomotor abilities (Paul et al., 2005).

In (Van der Noot et al., 2015b), we designed a 2D locomotion algorithm, combining a CPG and reflexes in a neuromuscular torque-based controller. This is coherent with Kuo's framework, suggesting to combine feedback (i.e. reflexes) and feed-forward (i.e. CPG) pathways in the control of a periodic task (Kuo, 2002). In Chapter 9, we incremented this latest controller, in order to provide lateral balance, and thus to walk in a 3D environment. After a single off-line optimization process, our controller could generate energy-efficient and human-like straight-walking gaits (both regarding kinematics and dynamics). Using a simulated COMAN platform (a 95 cm tall humanoid robot) as embodiment, the forward speed could be continuously commanded from 0.4 to 0.9 m/s. This range is close to the healthy human one, once scaled to the robot size. This speed modulation was achieved by changing high-level parameters, as linear or quadratic functions of the target speed.

The present contribution builds on top of Chapter 9 by extending the 3D straight-walking gaits to achieve control of the heading direction. More precisely, the CPG and reflex rules developed in Chapter 9 are adapted to control both the forward speed and the path curvature (direction and curvature). This is particularly relevant in tele-operation scenarios where the robot has to move in a cluttered environment. Similarly to the existing speed controller, the turning modulation is achieved by controlling a scalar input (i.e. the heading reference), together with the adaptation of high-level parameters, as linear or quadratic functions of the speed reference.

This chapter is divided as follows. Section 10.2 summarizes the neuromuscular controller developed in Chapter 9, achieving straight walking and forward speed modulation. That section also reports the simulation environment and the COMAN platform embodying our controller. The straight walking controller is later incremented in Section 10.3, in order to achieve the control of the steering direction. This additional steering control is further studied in Section 10.4, with focus on adaptation to different walking speeds. Then, Section 10.5 analyses different features of the resulting steering motion, among which the achievable walking curvature, the biped robustness and some tele-operated scenarios. Finally, Section 10.6 concludes the chapter.

10.2 Straight walking controller

Here, biped locomotion control is achieved using virtual muscles commanded by the combined action of reflexes and a central pattern generator (CPG). Using this approach, 3D straight walking with forward speed modulation can be achieved, as reported in Chapter 9. The main principles governing this approach are outlined in this section.

10.2.1 Neuromuscular model

The walking controller generates torque references at the joint level by recruiting (virtual) muscles. The full muscular configuration is presented in Figure 10.1 for the COMAN robot (Tsagarakis et al., 2013), which served as embodiment in our experiments (see Section 10.2.3). Each muscle is modeled as a set of equations based on a Hill muscle model (Hill, 1938), as depicted in Figure 10.1e.

More precisely, each muscle tendon unit (MTU) consists of two main elements: a contractile element (CE) and a series elastic one (SE), capturing the tendon compliance. Two additional elements only engage when the muscle state is outside its normal range of operation: the parallel elastic (PE) and the buffer elastic (BE) elements. The MTU length l_{mtu} can be retrieved from geometric relationships, while the length of CE (l_{ce}) is obtained as the time-integral of its velocity v_{ce}. In turn, v_{ce} depends on l_{mtu} from force-length and -velocity relationships, and on the muscle activation A_m, detailed later. Then, the length l_{se} of SE (computed as $l_{mtu} - l_{ce}$) provides a direct computation of the muscle force. This force is later multiplied by a lever arm to generate a torque contribution at the joint level. Finally, the different torque contributions are summed and sent to the robot joint torque low-level controller. More information is provided in (Geyer and Herr, 2010) and in Chapter 9.

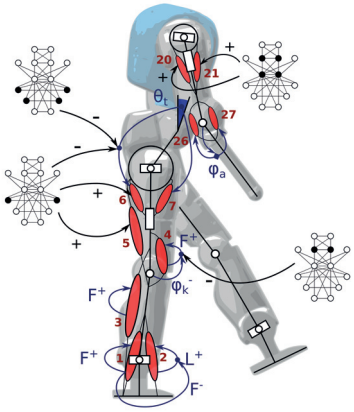
The muscle activations A_m are related to neural inputs S_m called stimulations, through the following excitation-contraction coupling first-order equation: $\tau_m \, dA_m/dt = S_m - A_m$, where τ_m is a time constant of $10\,ms$. Actuating the muscles, and so controlling the whole biped locomotion, thus amounts to compute appropriate stimulations S_m.

10.2.2 Reflexes and central pattern generator

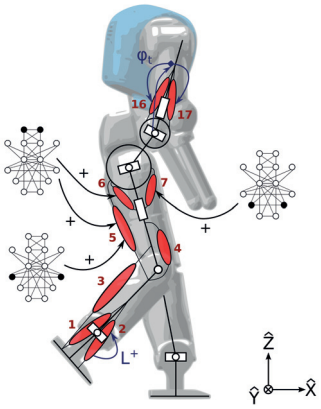
The neural stimulations S_m are computed as a combination of reflex mechanisms (feedback) and signals produced by a central pattern generator (CPG, feed-forward) (Rossignol et al., 2006). The combination of these two types of signals mainly follows a proximo-distal gradient. In other words, muscles close to the hips are mainly controlled by CPG signals, while the ones close to the feet (and so more impacted by external perturbations, like ground contact) are mainly driven by reflexes (Daley et al., 2007; Dzeladini et al., 2014).

All these reflexes and CPG inputs computations are fully detailed in Chapter 9 and Appendix G.4. They are also depicted in Figure 10.1. For instance, in the stance phase, the soleus muscle (SOL) is stimulated with the following positive force feedback reflex: $G_{SOL}\,\tilde{F}_{SOL}$, where G_{SOL} is a fixed parameter and $\tilde{F}_{SOL}$ is the SOL muscle force normalized by its maximal force. Most of the reflexes - similar to the one here as example - are adapted from (Geyer and Herr, 2010).

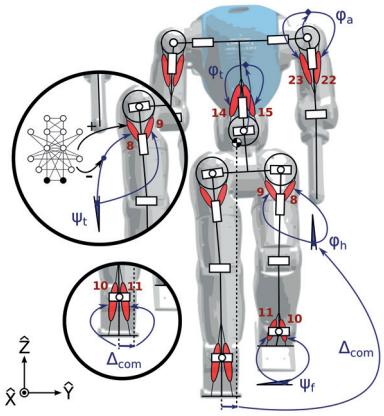
For straight walking, the CPG was designed as a twelve-neurons network of Matsuoka oscillators (Matsuoka, 1985, 1987). These are bio-inspired artificial oscillators, capturing the mutual inhibition between half-centers located in the spinal cord. Each neuron N_i obeys the state



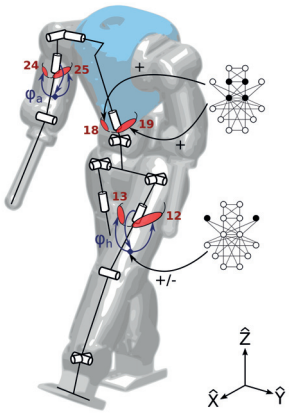
(a) Sagittal muscles (arm and stance leg)



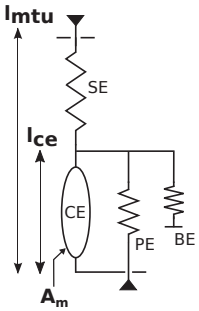
(b) Sagittal muscles (torso and swing leg)



(c) Lateral muscles (arm, torso and leg)



(d) Transverse muscles (arm, torso and leg)



(e) Hill muscle model

Sagittal leg		Lateral leg		Torso		Arms	
1	SOL	8	HAB	14	BTR	20	SET
2	TA	9	HAD	15	BTL	21	SFL
3	GAS	10	EVE	16	BET	22	SAB
4	VAS	11	INV	17	BFL	23	SAD
5	HAM			18	BRR	24	SER
6	GLU	**Transverse leg**		19	BRL	25	SIR
7	HFL	12	HER			26	EET
		13	HIR			27	EFL

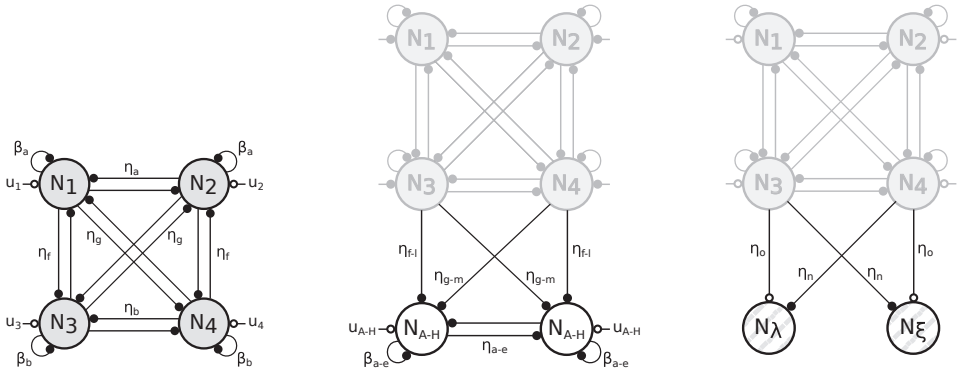
Figure 10.1: To actuate the biped's 23 joints, the controller recruits 27 different Hill muscle models (panel (e)) acting in different planes. These muscles are commanded by a combination of reflex signals and the CPG central unit. Muscles acting in the sagittal plane are displayed in panels (a) and (b), the ones acting in the lateral plane are displayed in panel (c), and finally, the ones acting in the transverse plane are depicted in panel (d). In particular, turning is mainly controlled by the HAB and HAD muscles in the lateral plane (panel (c)) and by the HER and HIR muscles in the transverse plane (panel (d)). The full muscle names are provided in Chapter 9 (see Section 9.2).

equations provided in (10.1), where x_i is the firing rate, v_i is the self-inhibition modulated by an adaptation constant β_j, η_k are the mutual inhibition factors (captured by the function $[\bullet]^+ = max(0,\bullet)$) and u_i is the external input. Finally, τ is the time constant for the rate of discharge of x_i, while the one of the self-inhibition v_i is related to τ through the adimensional parameter γ_j.

$$\begin{aligned} \dot{x}_i &= \frac{1}{\tau}(-x_i - \beta_j\, v_i - \sum \eta_k\, [x_l]^+ + u_i) \\ \dot{v}_i &= \frac{1}{\gamma_j\, \tau}(-v_i + [x_i]^+) \end{aligned} \tag{10.1}$$

In these equations, the index i corresponds to the neuron index, while the gains β_j, η_k, and the neurons x_l are specified in Figure 10.3a. Finally, γ_j takes the same index as β_j.

Furthermore, this CPG network was divided into two main parts. The first one, in charge of providing the main frequency and phasing during the gait, is composed of four neurons, i.e. the "rhythm generator" neurons (RG). They are denoted with a number (from 1 to 4), and depicted in Figure 10.2a. The second layer relies on the RG neurons to generate signals shaping the patterns of muscle stimulations. The corresponding neurons are denoted with a letter (from A to H) and are called "pattern formations" neurons (PF). They are displayed in Figure 10.2b. This two-layer structure is consistent with the two-level CPG biological structure proposed by (McCrea and Rybak, 2008) and validated during fictive locomotion experiments with decerebrated cats.

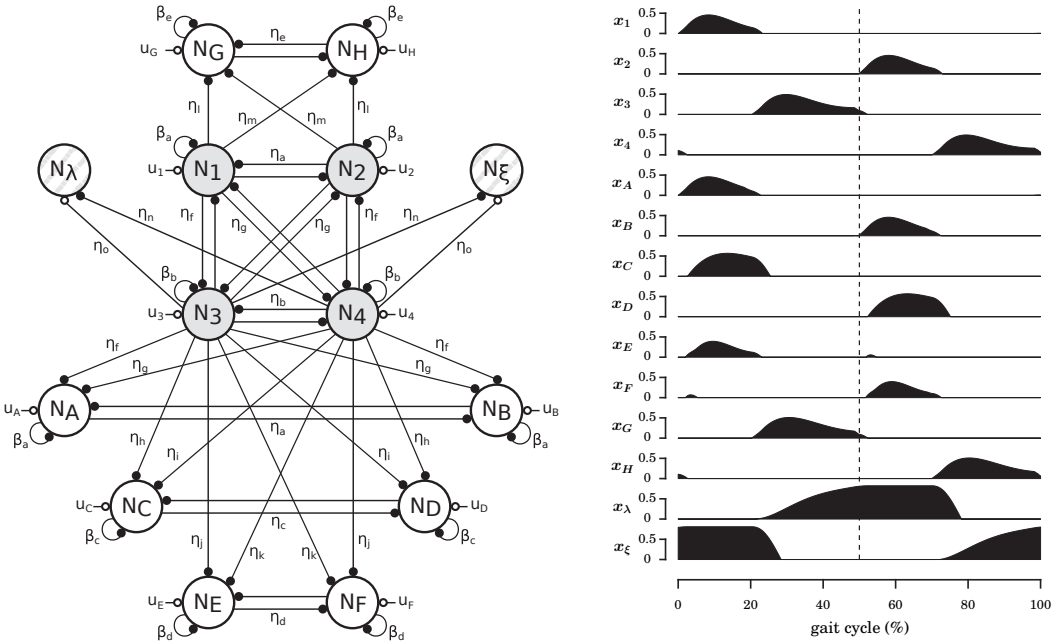


(a) RG neurons (b) PF neurons (c) TR neurons

Figure 10.2: The CPG network is built by assembling three types of components: (a) the rhythm generator (RG) layer (four fully connected Matsuoka neurons), (b) a pair of pattern formation Matsuoka neurons (PF) driven by the RG neurons and (c) a pair of non-Matsuoka neurons controlling turning (TR), also driven by the RG neurons. The vertical symmetry corresponds to the left/right legs symmetry.

The full CPG network, for straight walking, is pictured in Figure 10.3a. The corresponding

time-evolution of the neurons firing rates are displayed in Figure 10.3b. To generate the CPG contribution to a particular muscle stimulation S_m, different CPG outputs y_i were computed. They mainly consisted in extracting the positive firing rate of a PF neuron x_j (i.e. $y_i = [x_j]^+$). Then, the CPG contribution to a particular stimulation was computed as $S_m = \sum k_i\, y_i$, where k_i are gains. All details about these computational steps are provided in Chapter 9 and in Appendix G. In sum, Figure 10.1 displays all CPG contributions to the actuation of the different muscles.



(a) full CPG network

(b) neurons firing rates

Figure 10.3: Panel (a): The full CPG network is composed of the three components detailed in Figure 10.2: (a) "rhythm generator" neurons (RG, shaded, N_{1-4}), (b) "pattern formation" ones (PF, N_{A-H}) and (c) "turning" ones (TR, hatched, $N_{\lambda,\xi}$). TR neurons obey Equation (10.2), while the rest of the network is composed of Matsuoka oscillators, obeying Equation (10.1). Inter-neuron excitations are indicated with an empty circle, while plain circles capture inhibitions. All neurons but N_λ and N_ξ were already used for straight walking (see Chapter 9). Panel (b): Time-evolution of the 14 neurons firing rates of Figure 10.3a over one gait cycle (0% and 100% correspond to consecutive right foot strikes, the dashed line corresponds to the left foot strike in-between). These signals are obtained during one typical gait cycle of the locomotion resulting from the controller used in all the results of this contribution (called *reference controller*), with a speed reference of 0.65 m/s.

As detailed in Chapter 9, nine key control parameters were identified to modulate the biped forward speed. More precisely, these parameters were adapted as linear or quadratic function of a scalar input: the speed reference v_{ref}, providing forward speed modulation in the range of $[0.4; 0.9]$ m/s.

10.2.3 Experimental embodiment

As embodiment for our experiments, we used the COmpliant huMANoid (COMAN) robotic platform, in the Robotran simulation environment. COMAN is a 23 degrees of freedom (DOFs) full-body humanoid robot developed at the Italian Institute of Technology (IIT) (Dallali et al., 2013; Tsagarakis et al., 2013). This 95 cm tall robot, weighting 31 kg, features both series elastic actuators (SEA) (Tsagarakis et al., 2009) (mainly for sagittal joints) and traditional, stiff actuators (for the other joints). Regarding the robot sensors, each joint features position encoders and custom-made torque sensors. On top of that, an inertial measurement unit (IMU) is attached to the robot waist, while custom-made 6 axis force/torque sensors are placed below the ankle joints.

The simulation suite used to model COMAN is called Robotran (Samin and Fisette, 2003; Docquier et al., 2013). It is a symbolic environment for multi-body systems developed within the Université catholique de Louvain. To further minimize the gap between simulation and reality, a particular attention was paid to capture proper actuator dynamics and external forces from the environment, in particular with the ground contact model (GCM). Moreover, only sensory signals available on the real robot were used. On top of that, a uniform noise with a maximal amplitude of $0.4\,Nm$ was added to the torque measurement, to reproduce the one measured with the real platform. The motor equations, coupled to this noise, generate actual joint torques being different from their references, as would happen with the real robot. The integration was performed using a Runge-Kutta integration scheme with a $250\,\mu s$ time step. COMAN, in its simulation environment, is visible in Figure 10.1. More information about the robot and its simulator is provided in (Zobova et al., 2017).

10.3 Extension to curved motion

The controller outlined in Section 10.2 and fully detailed in Chapter 9 achieved straight walking in a 3D environment. In particular, forward speed modulation was achieved by adapting the speed reference v_{ref}, i.e. the main input of this bio-inspired controller. Building on top of this former contribution, this section introduces new features in order to achieve the control of the steering direction (curvature).

10.3.1 Lateral hip control

In (Courtine et al., 2006), ten healthy male adults walked along straight and curved paths while kinematics and electromyographic (EMG) data were recorded. A critical observation was that walking along curved paths did not require a dramatic re-organization of the basic EMG patterns required for walking straight-ahead. Inspired by this observation, the mechanisms governing the stimulation computations for straight walking described in Section 10.2 were mostly kept intact for curved motion. For most of them, changes in steady-state profile would thus be the result of the interplay between reflexes, CPG signals and the environment.

In this contribution, the turning motion is controlled by a scalar reference: the heading reference h_{ref}, bounded in the $[-1;1]$ interval during on-line control (with -1 corresponding to maximal left curvature, 0 to straight motion and 1 to maximal right curvature). This value is used at each strike to compute two turning control signals. At each right strike, the left turning command is updated as $\Gamma_L = [h_{ref}]^-$ (with $[\bullet]^- = -min(\bullet, 0)$). Similarly, the right turning command is updated as $\Gamma_R = [h_{ref}]^+$ at each left strike.

The curved motion mainly emerges from the foot transverse orientation control (detailed in Section 10.3.2). However, the body center of mass (COM) must also be controlled in the lateral plane, as mentioned in (Patla et al., 1999). In that contribution, two mechanisms were identified to laterally move the COM towards the new travel direction (i.e. towards the center of the curve): foot placement and hip strategy.

Foot placement impacts the distance between the COM and the center of pressure (COP), and so alters the COM acceleration magnitude and direction (Winter, 1995). Therefore, when turning indications are provided in advance, lateral COM control during turning is initiated by placing the inner foot (i.e. the foot inner to the curved motion) closer to the outer one (i.e. the other foot), thus accelerating the lateral COM towards the center of curve (Patla et al., 1999).

In Chapter 9, foot placement was mainly controlled during the supporting phase (i.e. stance phase excluding the last double support phase) of the contralateral leg, through the hip abductors (HAB) and adductors (HAD) muscles, in the lateral plane (see Figure 10.1c). More precisely, a hip lateral reference angle $\varphi_{h,l,ref}$ was computed, based on a feedback controller constraining the lateral COM position around its reference $\Lambda^*_{ref,h}$. This is detailed in Appendix H.1.

During straight walking, $\Lambda^*_{ref,h}$ was set to a fixed value $\Lambda_{ref,h}$. Steering motions require to augment this reference as $\Lambda^*_{ref,h,\{R,L\}} = \Lambda_{ref,h,\{R,L\}}\,(1 + \Delta_\Lambda\,\Gamma_{\{R,L\}})$, where R, L stand for right or left leg and Δ_Λ is a scaling parameter. This moves the lateral COM position reference of the inner foot away from the outer foot during curved motion. As a consequence, the inner foot will come closer to the outer foot (see Chapter 9 for more details about lateral control).

The other lateral COM strategy relies on controlling the body pendulum in the stance phase. This is achieved through the so-called hip strategy using muscle actuation at the hip and torso (Horak and Nashner, 1986). In fact, the body is controlled as a double pendulum with the lower limbs and the upper body moving in opposite directions, so that the COM moves towards the center of the curve (Patla et al., 1999).

In Chapter 9, the lateral hip muscles received a first burst provided by neurons N_E (right leg) and N_F (left leg), acting at the beginning of the corresponding leg supporting phase. Then, a feedback controller was activated on the torso lateral lean angle to track a reference Ψ^*_{ref}. This is detailed in Appendix H.1.

The hip strategy is mainly active during the outer leg stance phase, after completion of the foot placement strategy (Patla et al., 1999). In order to anticipate a possible change in steering direction with foot placement before activating the hip strategy, the values of the turning commands $\Gamma_{\{R,L\}}$ computed during the former gait cycle are used. More precisely, Γ_R^{-1} is equal to the value of Γ_R at the penultimate left strike, Γ_L^{-1} is equal to the value of Γ_L at the penultimate right strike.

First, the neuron N_E excitation (i.e. u_E) is augmented by a contribution equal to $\nu_l \Gamma_R^{-1}$, while u_F is augmented by $\nu_l \Gamma_L^{-1}$ (ν_l is a scaling parameter). This increases the CPG burst during curved motion and brings the torso closer to the hip, thus moving the COM towards the center of the curve. In order to achieve a similar effect for the feedback controller on the torso lateral lean angle, its reference is set as $\Psi^*_{ref,\{R,L\}} = \Psi_{ref}\,(1 + \Delta_\Psi \Gamma^{-1}_{\{L,R\}})$, where Δ_Ψ is a scaling parameter.

10.3.2 Transverse hip control

The hip transverse joints control the foot orientation motion, and thus impact the biped change in heading. In (Courtine and Schieppati, 2003), six healthy male adults walked along straight and curved paths. When walking along a curved path, the body turning mainly occurred during the stance phase of the outer foot. It also appeared that the inner foot rotation occurred mostly during the inner limb swing phase.

In Chapter 9, the transverse hip muscles (i.e. hip external (HER) and internal (HIR) rotator muscle groups, see Figure 10.1) were controlled to maintain the corresponding joint in its homing position, as depicted in Figure 10.1d. To do so, these two antagonist muscles received stimulations proportional to the output of a feedback controller on the hip transverse reference angle $\varphi_{h,t,ref}$ (set to zero for straight walking). This is detailed in Appendix H.1.

To generalize this for curved motion, the $\varphi_{h,t,ref}$ reference angle must be adapted to produce leg transverse motion coherent with the observations of (Courtine and Schieppati, 2003). Therefore, the turning motion is expected to start approximately at the swing phase initiation of the inner foot, while the legs realignment (i.e. feet realigned with the waist in the transverse plane) is expected at the beginning of the swing phase of the outer foot. To keep things simple, the two legs transverse rotations are commanded simultaneously (i.e. turning and realignment of both legs happen at the same time).

Phase locking is thus crucial to synchronize legs rotation during the gait cycle. This mechanism is provided by the RG neurons. N_1 and N_2 start firing respectively after the right and the left feet strikes. N_3 and N_4 fire during the rest of the gait, i.e. before the next strike happens (mainly during left swing phase for N_3 and right swing phase for N_4). This behavior can be observed in Figure 10.3b.

To achieve the requested synchronization, two new neurons are introduced: the "turning" neurons (TR), depicted in Figure 10.2c. N_λ is expected to control left turning, while N_ξ is in

charge of right turning. Taking inspiration from the Matsuoka rules detailed in Equation (10.1), the $\eta_k [x_l]^+$ terms are recruited (introducing two gains η_n and η_o) to generate both excitation and inhibition of the TR neurons. N_λ is excited by N_3 to start leg rotation during the left leg swing motion, while N_ξ is excited by N_4, for symmetrical reasons. The legs realignment is achieved with inhibition connections, triggered during the outer leg swing phase. This is done with neuron N_4 for N_λ and N_3 for N_ξ. Their time derivative rules are thus the following:

$$\begin{aligned} \dot{x}_\lambda &= \frac{1}{\tau}(\eta_o [x_3]^+ - \eta_n [x_4]^+ [x_\lambda]^+_{1/0}) \\ \dot{x}_\xi &= \frac{1}{\tau}(\eta_o [x_4]^+ - \eta_n [x_3]^+ [x_\xi]^+_{1/0}) \end{aligned} \tag{10.2}$$

The $[\bullet]^+_{1/0}$ function returns 1 if its argument is positive, 0 otherwise. Its purpose is to prevent the corresponding neuron firing rate to become negative. The CPG network is incremented by these two (non-Matsuoka) TR neurons in Figure 10.3a. The firing rates evolution of N_λ and N_ξ are also displayed in Figure 10.3b.

Using these two new neurons, the hip transverse reference $\varphi_{h,t,ref}$ is computed as follows:

$$\begin{aligned} \varphi_{h,t,ref,R} &= -\Gamma_R^{-1} k_{y,in} [x_\xi]^+ - \Gamma_L^{-1} k_{y,out} [x_\lambda]^+ \\ \varphi_{h,t,ref,L} &= \quad \Gamma_L^{-1} k_{y,in} [x_\lambda]^+ + \Gamma_R^{-1} k_{y,out} [x_\xi]^+ \end{aligned} \tag{10.3}$$

$k_{y,in}$ and $k_{y,out}$ are respectively scaling factors for the inner and outer legs. Similarly to the hip strategy (see Section 10.3.1), $\Gamma^{-1}_{\{R,L\}}$ is used instead of $\Gamma_{\{R,L\}}$, in order to engage heading modulation after the foot placement strategy.

10.3.3 Steering parameters optimization

Seven key turning parameters were introduced in Sections 10.3.1 and 10.3.2, namely $k_{y,in}$, $k_{y,out}$, Δ_Λ, Δ_Ψ, η_n, η_o and v_l. While it is possible to manually tune them to achieve robust curved motion, another solution is to rely on an optimizer to find these parameters, while maximizing a desired fitness function.

In Chapter 9, a set of optimized parameters achieving straight walking (called *reference controller*) was used in order to produce most results. Starting from this reference controller, a particle swarm optimization (PSO) algorithm (Kennedy and Eberhart, 1995) is run on the seven key turning parameters. The purpose is to achieve turning motion with the shortest steering radii (i.e. sharpest turns) without falling.

Therefore, the following scenario is used. The biped must walk during 90 s with a fixed speed reference v_{ref} (see Chapter 9) and a changing heading reference h_{ref}. During the $[10;25]$ s time interval, h_{ref} is set to 0.3, during the $[30;45]$ s time interval to -0.6, during the $[50;65]$ s time interval to 0.9 and during the $[70;85]$ s time interval to -1.2. The rest of the time, h_{ref} is set to zero. Using this, the walker faces increasing heading references, in both directions. We arbitrarily chose to restrict the range of h_{ref} to $[-1;1]$ during on-line control, and to $[-1.2;1.2]$ during optimizations, so that the walker is optimized in tougher conditions, thus increasing its robustness for extreme heading references.

The resulting fitness function to be maximized is the cumulative increments in the heading angle (i.e. absolute walker direction angle measured in the transverse plane) during the time periods when h_{ref} is non zero.

10.4 Steering parameters evolution with speed

Seven key turning parameters were identified in Section 10.3. This section studies how these parameters adapt with forward speed, providing a single controller with optimal turning control for the whole range of speed commands.

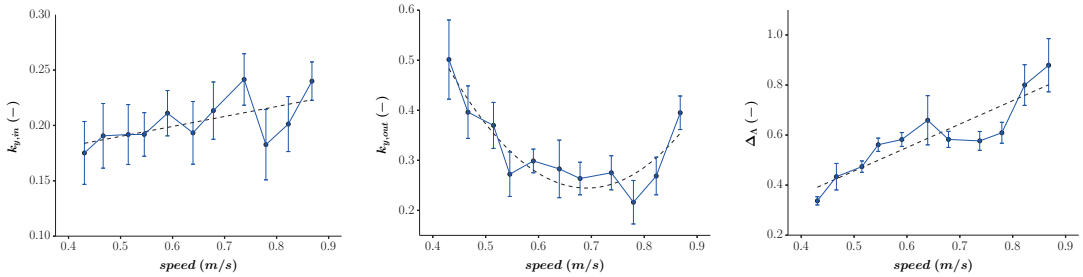
10.4.1 Polynomial approximations

The evolution of the seven key turning parameters is performed as follows. For the whole range of speed references v_{ref} (i.e. from 0.4 m/s to 0.9 m/s, with a discretization of 0.05 m/s), ten optimizations are performed for each target speed, according to the method introduced in Section 10.3.3. The corresponding results are reported in Figure 10.4.

Intuitively, the evolution of these parameters can be approximated with polynomial functions. To select the appropriate orders capturing parameters evolution without over-fitting, a model goodness-of-fit analysis using the sum of squared values of the prediction errors is used (Smith and Rose, 1995). In fact, for each polynomial order, the corresponding p-value is computed to measure the likeliness that the selected order is appropriate to represent the parameter evolution. More information is provided in Appendix G.6.

Table 10.1 reports the p-values corresponding to polynomial approximations of orders 0, 1 and 2 of the data provided in Figure 10.4, based on the least square errors. Similarly to Chapter 9, the first order with a p-value larger than 0.1 is selected (grey cells). This is a less strong analysis than rejecting the opposite null hypothesis, but is considered to be sufficient to design the control rules.

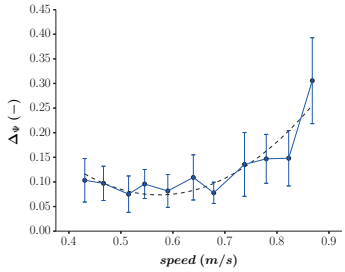
It appears from the results of Table 10.1 that two parameters (Δ_Λ and v_l) did not reach the threshold of 0.1. For these two parameters, the order with the largest p-value was selected, i.e. 1 for Δ_Λ and 2 for v_l. The $k_{y,in}$ p-value for order 0 barely exceeds the critical threshold of 0.1, while order 1 is much larger (i.e. close to 0.35). Therefore, we arbitrarily decided to select



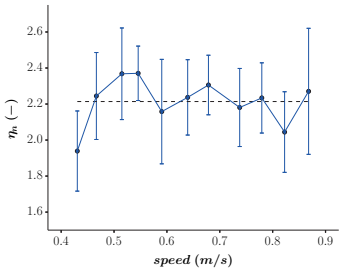
(a) Transverse gain $k_{y,in}$

(b) Transverse gain $k_{y,out}$

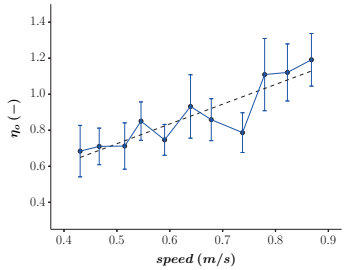
(c) COM modulation Δ_Λ



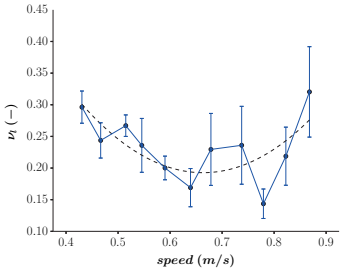
(d) Hip modulation Δ_Ψ



(e) CPG connexion η_n



(f) CPG connexion η_o



(g) Lateral excitation ν_l

Figure 10.4: Ten optimizations are performed for each target speed (from 0.4 m/s to 0.9 m/s with an interval of 0.05 m/s).The actual speed of each solution is measured (during straight walking), along with the optimized value of the seven key turning parameters. For each target speed, we gather the ten optimization final results, reporting their mean and standard deviations. For graph legibility, the error bars represent half of the standard deviations. Dashed lines correspond to the polynomial approximations whose order is reported in Table 10.1, using the minimum mean square error method.

Table 10.1: Polynomial approximations: p-values

	order 0	order 1	order 2	selected
$k_{y,in}$	0.108	**0.342**	0.268	1
$k_{y,out}$	0	0	**0.163**	2
Δ_Λ	0	**0.001**	0.001	1
Δ_Ψ	0	0.028	**0.528**	2
η_n	**0.729**	0.645	0.691	0
η_o	0	**0.487**	0.54	1
ν_l	0.001	0.001	**0.044**	2

order 1 for this parameter. Regarding the η_n parameter, a polynomial approximation of order 0 was selected (i.e. constant value), thus reducing to six the number of key turning parameters evolving with speed. The polynomial approximations using the selected orders are depicted with dashed lines in Figure 10.4.

10.4.2 Parameters analysis

The scaling parameters $k_{y,in}$ and $k_{y,out}$ are directly related to the curvature radius. Indeed, they control the legs transverse rotations, and so the turning. The inner foot parameter ($k_{y,in}$) increases with forward speed, favoring sharp turns for the highest speeds. The range of values obtained for the outer foot parameter ($k_{y,out}$) is larger than the one obtained for the inner foot ($k_{y,in}$). Therefore, $k_{y,out}$ appears to be more speed dependent than $k_{y,in}$. Here, the sharpest turns are obtained at speed extrema. Globally, the highest speed are expected to produce the largest curvatures. Indeed, despite their lower $k_{y,out}$ when compared to the slowest speeds, they also benefit from the largest $k_{y,in}$ parameters. In contrast, speeds in the middle of the range are expected to generate the most gentle turns.

Directly related to these parameters, the CPG excitation and inhibition weights η_n and η_o also affect the legs transverse rotations. Only the excitation parameter (η_0) appears to evolve with speed, producing larger commands for faster speeds. This increases the speed of the hip motion, as well as the plateau reached at the end of this initial motion, thus increasing the heading change. The constant η_n value is much larger than η_o, resulting in a quick realignment of the feet with the waist during the outer leg swing motion.

In the lateral plane, the inner foot initiates turning by coming closer to the outer one. This foot position strategy is amplified with a bigger Δ_Λ. Therefore, the linear increase of this parameter with forward speed indicates that the walker takes advantage of the inertia effects to induce bigger accelerations towards the center of the curve, when walking at high speeds.

Finally, the lateral hip strategy is controlled by the remaining parameters ν_l (CPG excitation) and Δ_Ψ (reflex). Similarly to $k_{y,out}$, the highest values are reached at the boundaries of the speed range (especially at high speeds). Increasing the hip strategy effects appears therefore to be correlated with sharper turns.

10.4.3 Parameters co-optimization

The controller design can now be further extended to co-optimize all key turning parameters in a single optimization, therefore recruiting the optimal turning parameters for the whole speed range, and not for a single speed. The seven key parameters studied in Section 10.4.1 are replaced by polynomial approximations, whose order is selected according to Figure 10.4 and Table 10.1. Because η_n is actually of order 0, the corresponding parameter is left as a constant.

The corresponding rules are reported in Appendix H.2. They involve sixteen parameters to optimize, whose bounds are indicated in Table H.1. A last optimization was then performed with the reference controller from Chapter 9, in order to optimize these turning parameters. The resulting controller, combining forward speed and turning modulation, keeps the name *reference controller*.

10.5 Results

The results presented in this section were obtained using the reference controller. The evolution of the walking curvature with heading reference is first studied before characterizing the gait main features. Finally, some tele-operated steering scenarios are presented.

10.5.1 Curvature radius control

The biped can achieve sharper turns when increasing its heading reference h_{ref}. However, this also depends on its walking speed (and so, on v_{ref}). This effect was first studied with the reference controller receiving three representative speed references: the middle of the speed range (i.e. $v_{ref} = 0.65\,m/s$) and two speeds close to the speed extrema ($0.45\,m/s$ and $0.85\,m/s$). To quantify it, the following experiment was performed. For each of the speed references, the biped received a heading reference ranging between 0 (i.e. straight walking) and 1 (i.e. maximal right steering command). The steady-state behavior thus corresponded to a motion being close to a circle (except for $h_{ref} = 0$): both speed and heading references were stationary and non-zero. The resulting curvature was measured as the inverse of the radius of the circle described by COMAN in steady-state, after a full rotation. Corresponding results are presented in Figure 10.5. Because of the left/right symmetry, left steering is not reported.

As expected, the curvature increased (sharper turns) with increasing h_{ref} commands. Moreover, a linear relationship was observed between the heading reference h_{ref} and the resulting curvature. This was mainly due to the transverse hip joints control, tracking position references proportional to h_{ref}. The curvature also depended on the speed reference v_{ref}. For large heading references, larger curvatures were obtained for faster speeds. However, this trend was not clear at lower speeds. Therefore, the experiment of Figure 10.5 was extended to the whole range of speed references (i.e. from $0.4\,m/s$ to $0.9\,m/s$). Corresponding results are depicted in Figure 10.6.

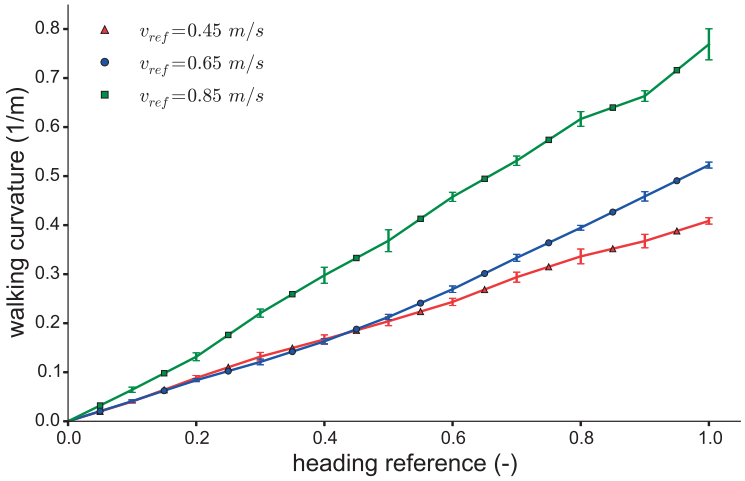


Figure 10.5: For each representative speed references (i.e. v_{ref} set to 0.45 m/s, 0.65 m/s or 0.85 m/s), COMAN received right heading references h_{ref} (from 0 to 1 with a discretization of 0.1). The walking curvature was measured from the circle described by the robot, over ten trials. The corresponding mean and standard deviations are reported. For left heading references (i.e. $h_{ref} < 0$), similar results were obtained (due to the symmetry of the configuration).

It appears that the curvatures ranged between 0 (i.e. straight walking, obtained with $h_{ref} = 0$) and 0.79 for these commands. As mentioned in Section 10.4.2, the sharpest turns were obtained for the fastest speeds (followed by the slowest ones for small h_{ref} commands), while the speeds around 0.55 m/s provided the most gentle curves. This is mainly due to the large high-level parameters $k_{y,in}$ and $k_{y,out}$ which were obtained during the optimization process for the speed extrema (see Section 10.4.2).

Importantly, collisions between legs were frequently observed for heading references h_{ref} larger or equal to 0.5. These collisions could also be sometimes detected for lower h_{ref} values, especially for speed references close to the extrema. In fact, this problem was already observed in Chapter 9 for straight walking at speed extrema. This issue is further discussed in Section 10.6.4.

10.5.2 Gait main features evolution with turning reference

The following gait features were studied: speed, stride length, stride period and swing ratio. Their evolution with h_{ref} (positive for right turns) is reported in Figure 10.7 for the same three representative speed references used in Section 10.5.1 (i.e. 0.45 m/s, 0.65 m/s and 0.85 m/s).

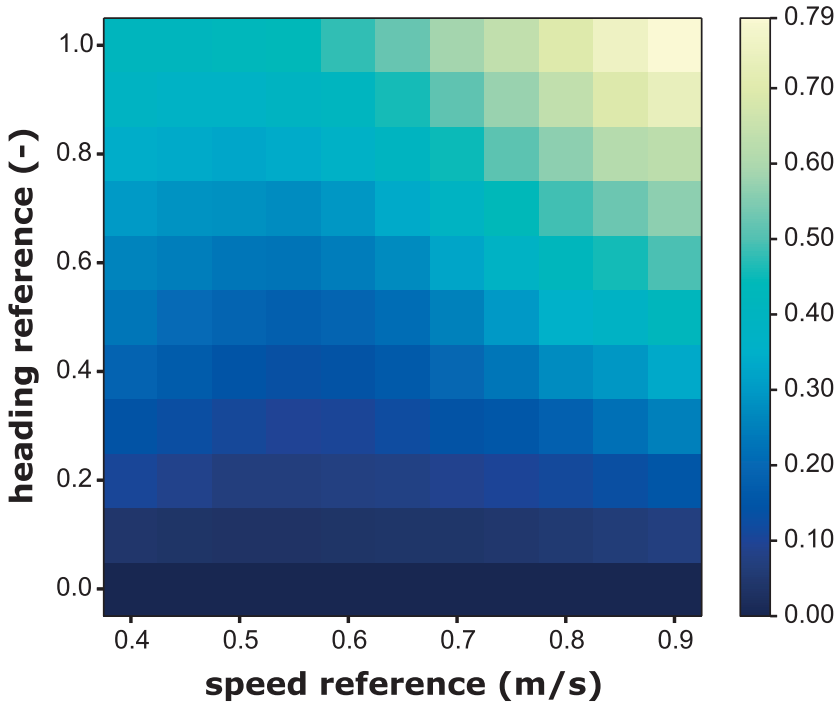
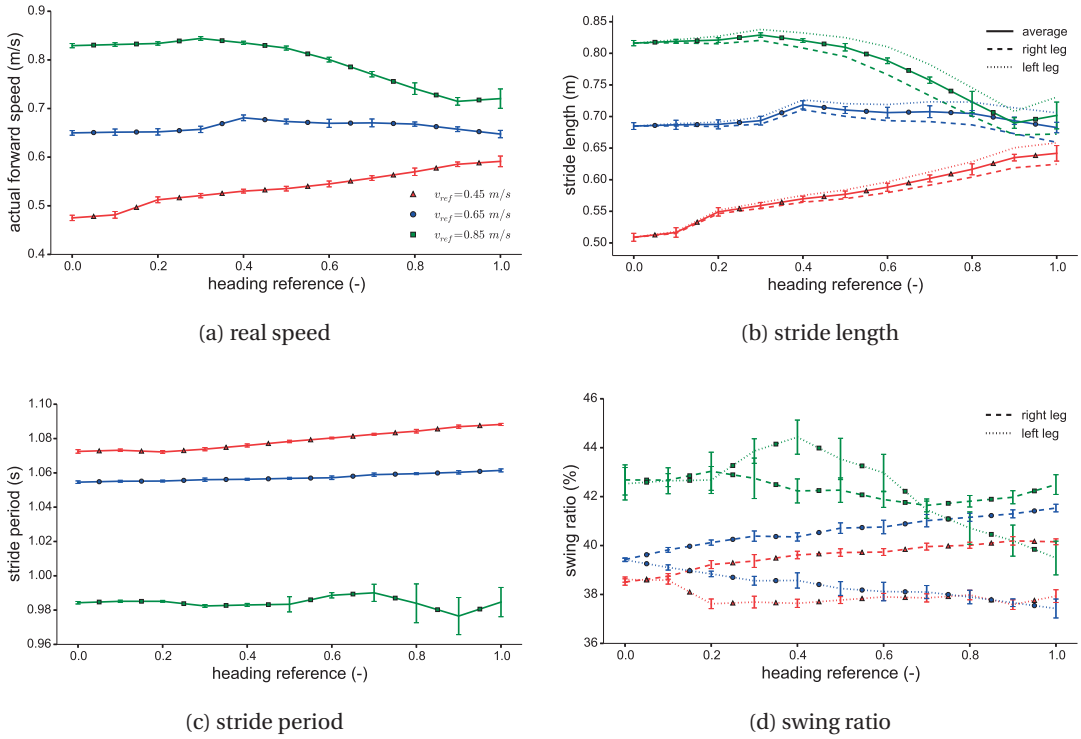


Figure 10.6: Similarly to Figure 10.5, the walking curvature was measured when COMAN received right heading references. This is presented here for the whole spectrum of speed references, i.e. from 0.4 m/s to 0.9 m/s, with a discretization of 0.05 m/s. The color map represents the resulting walking curvature (averaged over ten runs).

Regarding speed evolution, Figure 10.7a shows that the speed remained quite constant and close to its reference when this reference v_{ref} was set to 0.65 m/s. In contrast, the other v_{ref} values resulted in speeds converging towards the middle of the speed range. Both in (Courtine et al., 2006) and (Courtine and Schieppati, 2003), human subjects tended to steadily decrease their speed when facing an increasing curvature. During these two experiments, the mean velocity of the subject was around 68% BH/s, where BH stands for body height. Considering that COMAN height would be close to 1.06 m if it had a head, this corresponds to a speed of 0.72 m/s, so in between 0.65 m/s and 0.85 m/s. Therefore, our results reporting a speed decrease are consistent with human observations.

When comparing the stride length (Figure 10.7b) and the stride period (Figure 10.7c), it clearly appears that the change in speed was mainly correlated with the stride length evolution. During curved trajectories, the stride lengths of the inner and outer limbs were expected to differ, in contrast to straight walking. Indeed, the external leg covered a longer path because its foot moved along a circular trajectory whose radius was larger than the one of the inner foot (Courtine and Schieppati, 2003). This is consistent with the results of Figure 10.7b where the inner (right) stride length became significantly lower than the outer (left) one, as the turning reference increased. In (Courtine et al., 2006), it was observed that the stride length



(a) real speed

(b) stride length

(c) stride period

(d) swing ratio

Figure 10.7: The evolution of some gait features with the heading reference h_{ref} is studied, while the reference controller receives three distinct speed references v_{ref}: 0.45 m/s (red, triangles), 0.65 m/s (blue, circles) and 0.85 m/s (green, squares). Panel (a) presents the real walker speed. In panel (b), the average stride length (solid) is decomposed between right (dashed) and left (dotted) leg contributions. Panel (c) shows the evolution of the stride period. Finally, the ratio of time during which each leg is in swing phase is presented in panel (d). For each measurement, ten simulation runs were performed. Their mean and standard deviations are presented. For graph legibility, panel (b) only depicts the standard deviation of the average stride length (standard deviations of each leg are similar to this average value). These results are presented during right steering experiments, but are similar for left turns, due to the symmetry of the configuration.

modification during turning (compared to straight walking) mainly affected the inner limb. In Figure 10.7b, this trend is not observed: both the inner and outer legs stride lengths were affected. This is potentially related to the transverse hip control rules being recruited (see Section 10.3.2).

The stride period evolution with turning reference in Figure 10.7c is not significant. A small increase was observed for the highest heading references when v_{ref} was set to 0.45 m/s or 0.65 m/s, while the highest speed reference (0.85 m/s) oscillated around its mean value with higher standard deviations. Similarly, (Courtine et al., 2006) observed that human gait cycle duration was hardly affected by the curvature of the path. Only the tighter curves caused a modest increase in cycle duration.

Results of Figure 10.7c could not be decomposed between right and left contributions because the averaged stride period had to be the same for both legs. However, the portion of time when each leg was in swing phase (over the whole gait cycle) was leg-dependent, as presented in Figure 10.7d.

In (Courtine et al., 2006), both legs exhibited similar stance durations during straight-walking but showed opposite modulation of stance duration during curved motion. A similar behavior can be seen in Figure 10.7d, especially, for v_{ref} set to 0.45 m/s or 0.65 m/s. However, (Courtine et al., 2006) also observed that the swing duration of the inner leg significantly decreased with curvature, while changes in stance duration of the outer limb were less pronounced. Only the speed reference of 0.85 m/s (with h_{ref} smaller than 0.7) showed a lower swing ratio for the inner limb, compared to the outer one. The swing ratio evolution in Figure 10.7c being relatively small, this result is less significant, but still indicates some discrepancy compared to human walking.

10.5.3 Robustness when turning

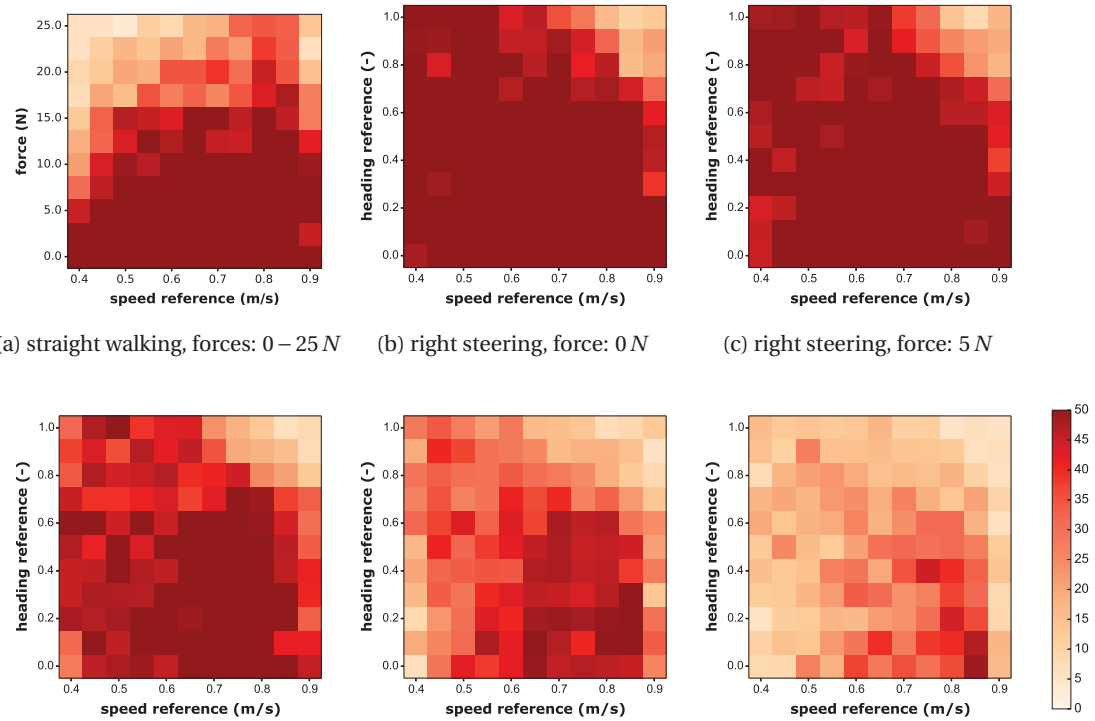
In order to compare the robustness of the controller during straight and curved walking, the following experiment was performed. COMAN received random pushes on the torso when walking at different speeds. First, these pushes were applied during straight walking with a magnitude selected between 0 N and 25 N, during 0.2 s in the transverse plane. These pushes were applied with a time interval randomly selected between 5 and 6 s. Each push orientation in the transverse plane was randomly selected in the $]-\pi;\pi]$ interval (i.e. all possible directions with an equal probability). The torque noise presented in Section 10.2.3 was active. The robot was blind, i.e. it had no other feedback than the reflexes driving its neural controller.

Robustness was quantified by measuring the time the robot could walk without falling, when facing these pushes. The time count started during steady-state walking, when the external pushes started, and was limited to an upper bound of 50 s. The results are reported in Figure 10.8a.

Globally, faster speeds could resist to larger pushes. The only exception was the maximal speed reference ($v_{ref} = 0.9\,m/s$), which was less stable. Indeed, less robust gaits were usually obtained for the extrema of the optimized speed range.

The same experiment was performed during turning motion, when receiving right heading references h_{ref} ranging between 0 and 1, through steps of 0.1. This was tested for the following set of push amplitudes: 0 N (Figure 10.8b), 5 N (Figure 10.8c), 10 N (Figure 10.8d), 15 N (Figure 10.8e) and 20 N (Figure 10.8f).

Globally, the walker robustness was not significantly deteriorated with increasing turning references, with one notable exception: the highest speeds v_{ref} with the highest heading references h_{ref}, usually causing a quick fall of the walker. In particular, this was also observed without external pushes (i.e. only with torque noise reading), as depicted in Figure 10.8b.



(a) straight walking, forces: $0-25\,N$ (b) right steering, force: $0\,N$ (c) right steering, force: $5\,N$

(d) right steering, force: $10\,N$ (e) right steering, force: $15\,N$ (f) right steering, force: $20\,N$

Figure 10.8: For the whole spectrum of speed references, pushes were applied to the torso of COMAN, as described in Section 10.5.3. The color map represents the time the robot could walk before falling (in [s], averaged over ten runs and limited to $50\,s$). In panel (a), different push amplitudes were selected during straight walking. In the other panels, a single push amplitude was selected ($0\,N$ (b), $5\,N$ (c), $10\,N$ (d), $15\,N$ (e) or $20\,N$ (f)) while the biped received increasing right heading references h_{ref}. These graphs are similar for left steering, due to the left/right symmetry.

However, this corresponds to very small steering radii, up to $1.26\,m$ (i.e. corresponding to curvatures up to 0.79, see Figure 10.6). This smallest radius is about twice the size of the robot leg, and thus represents a very sharp turn for COMAN.

Interestingly, for speed references up to $0.65\,m/s$, the walker barely never fell when receiving pushes up to $5\,N$, even for extreme h_{ref} commands. For higher push amplitudes, a slight decrease in robustness was observed with higher h_{ref}. However, this usually corresponds to push amplitudes also causing falls in straight walking gaits (see Figures 10.8e and 10.8f).

10.5.4 Tele-operated steering

Using the reference controller, it is possible for a human tele-operator to achieve on-line control of both speed and steering (direction and curvature). In particular, this can be done by

using a single joystick, e.g. with one axis controlling v_{ref} in the $[0.4; 0.9]\ m/s$ range and the other axis controlling h_{ref} in the $[-1; 1]$ range. This allows an intuitive control and modulation of the robot gait, which can be used to freely navigate in a cluttered environment. Snapshots of such a modulation are visible in Figure 10.9.

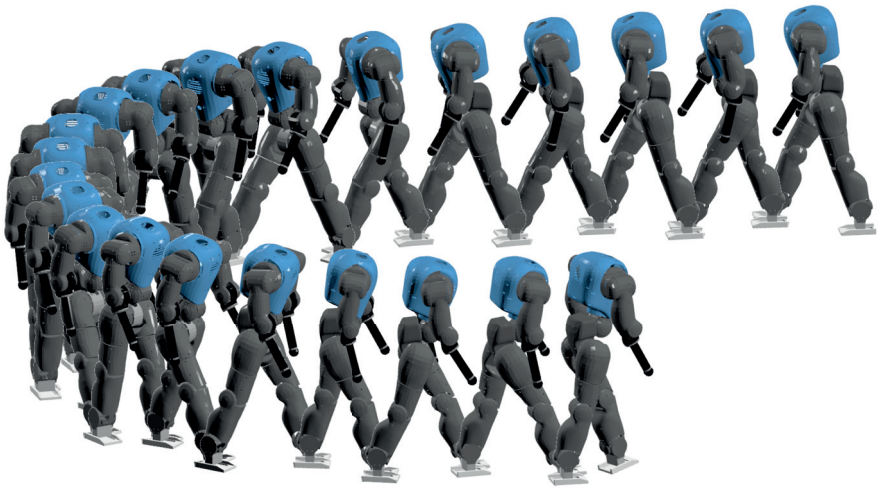


Figure 10.9: Receiving a speed reference v_{ref} of 0.65 m/s, the robot walks with an initial heading reference h_{ref} of 0 (straight walking) before changing this command to -1 (maximum left steering).

A longer walk experiment was performed with COMAN receiving speed and heading references whose evolution with time is depicted in Figure 10.10. The trajectory of COMAN during this last experiment is visible in Figure 10.11, where its footprints are depicted together with the evolution of its COM and center of pressure (COP). Steering was performed in both directions, and with different curvatures.

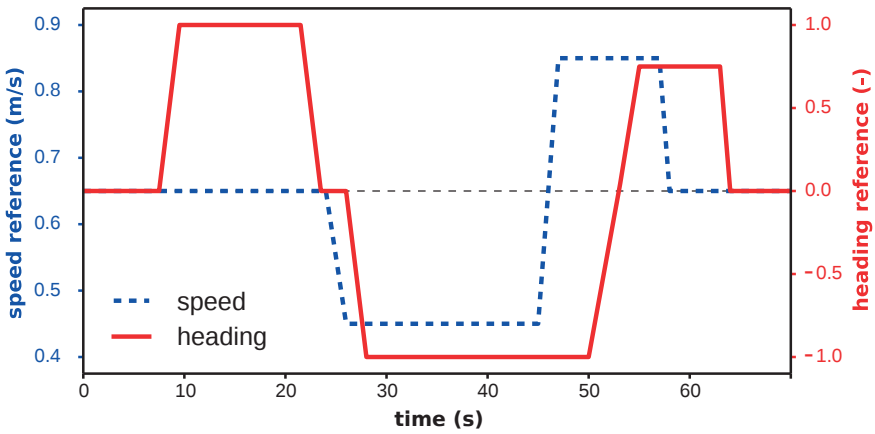


Figure 10.10: Temporal evolution of the commands used during the walk experiment depicted in Figure 10.11. More precisely, the speed reference v_{ref} and the heading reference h_{ref} control respectively the biped forward speed and its curved motion.

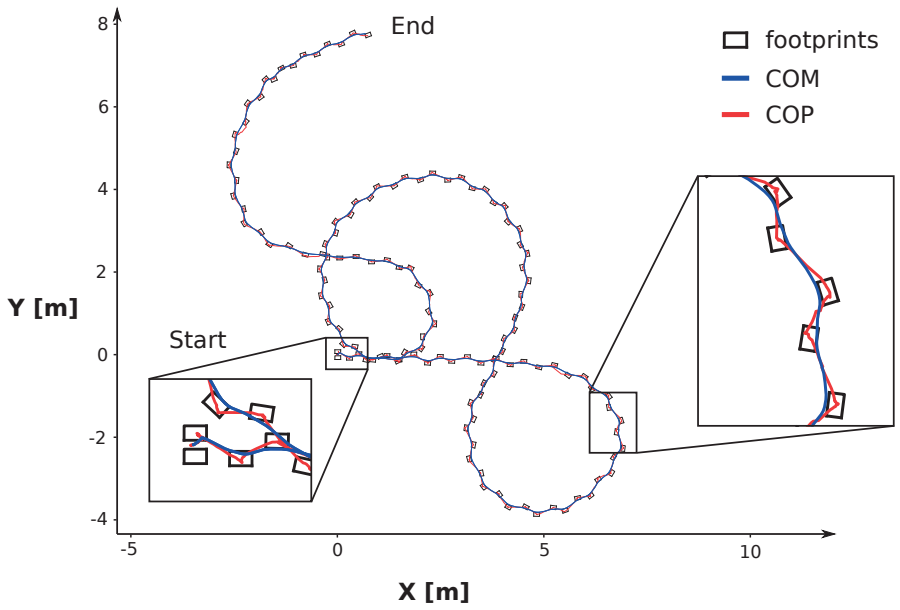
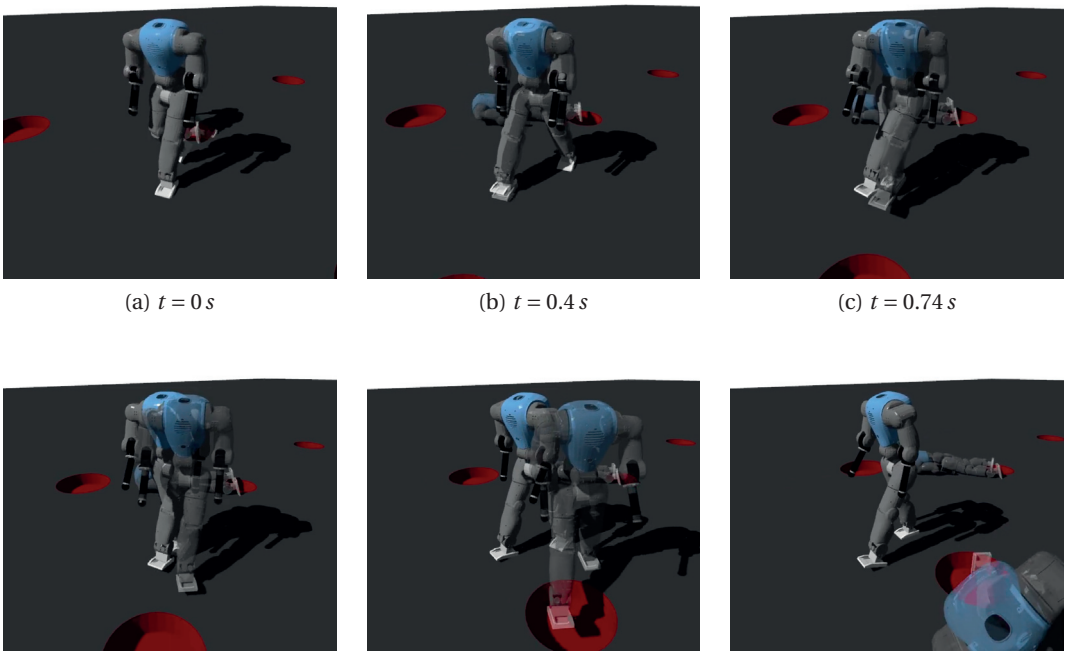


Figure 10.11: Footprints of COMAN on a planar ground, when walking while being teleoperated by a human. This corresponds to the walk experiment presented in Figure 10.10. The evolution of the center of mass (COM) and of the center of pression (COP) are also depicted.

Finally, this deliberate steering can also be used to avoid stepping into holes, as presented in Figure 10.12. In that last experiment, different commands were sent to the walker before stepping into different holes. According to the commands received, the biped either fell or succeeded to avoid the hole. In (Van der Noot et al., 2015b), we presented a similar experiment for a 2D scenario (i.e. with the biped waist artificially constrained to stay in the sagittal plane) where the modulation of v_{ref} resulted in a successful crossing of a hole.

10.6 Discussion

In this contribution, we presented an extension of the 3D straight walking controller we developed in Chapter 9. By embracing the concept of the limit cycle, it relaxes constraints inherent to more traditional walker controllers, therefore achieving faster, more energetically efficient (see Chapter 9) and more human-like (e.g. straight knee walking) gaits. However, the walker still features on-line adaptation capabilities. On top of its forward speed modulation, the biped is now capable of controlling its steering direction and curvature. The resulting walking controller is therefore capable of fully navigating in a cluttered environment (with a nearly-flat ground), by avoiding obstacles.



(a) $t = 0\,s$ (b) $t = 0.4\,s$ (c) $t = 0.74\,s$

(d) $t = 0.91\,s$ (e) $t = 1.41\,s$ (f) $t = 1.91\,s$

Figure 10.12: COMAN receives changing steering commands (speed and heading) while walking. Regularly, different steering commands are sent to two simulations starting from the same state. In the snapshots presented here (with time indications), the semi transparent COMAN and the fully visible one receive different steering commands, while starting from the same state (see panel (a)). As can be seen, body motion differs between these two models, allowing the fully visible one to escape the holes, while the semi transparent one falls. This experiment was repeated several times during COMAN walking (see the other semi transparent model, already on the floor in panel (b)).

10.6.1 Gait modulation

The whole controller, combining CPG and reflexes, can be obtained by two successive optimizations. In the first one, straight walking control is obtained as described in Chapter 9, while the second optimization provides turning capabilities. These two optimizations are performed on the whole speed range (i.e. from 0.4 to $0.9\,m/s$), therefore co-optimizing all the parameters, in order to increase the performances (energy efficiency, robustness, speed modulation, turning control) for any achievable speed.

Speed and curved motions can then be controlled by modulating the speed reference v_{ref} and the heading reference h_{ref} inputs, thus providing high-level inputs for on-line control. In sum, we managed to combine the benefits of limit cycle walkers (as shown in Chapter 9) with the capacity to steer the robot velocity and direction. Similar skills are usually achieved using inverse kinematics/dynamics approaches. Our framework thus offers to handily tele-operate the biped with few commands, while exhibiting human-like gait features.

10.6.2 Walker robustness

Humanoid robots are currently far from reaching the impressive robustness of real humans during locomotion. This is one of the main reasons preventing them from being used outside controlled environments like laboratories. In the blind walking experiments of Section 10.5.3, COMAN could naturally resist to pushes, both for straight walking and during curved motion. Interestingly, this was achieved without changing a single parameter of the controller. This is potentially related to the viscoelastic muscle properties, inducing gait adaptations to perturbed environments. However, the effects of these viscoelastic properties regarding the walking robustness remains to be quantified.

Importantly, additional strategies should probably be implemented to sustain stronger pushes. For instance, (Heremans et al., 2016) developed a neural controller progressively learning appropriate muscular stimulations to reject disturbances. Because this approach also relies on a musculo-skeletal model, it is compatible with the controller presented in this contribution.

Robustness can also be improved by using a more bio-inspired embodiment. For instance, the rigid foot used here is far from the flexible human one. In (Colasanto et al., 2015), we showed that replacing the robot rigid foot by a model of a human prosthesis led to more robust gaits. Indeed, in contrast to many approaches requiring to keep the feet flat on the ground, this constraint is not inherent to our bio-inspired approach.

10.6.3 Human steering strategies

While the recruitment of CPGs in locomotion control is widely accepted for many vertebrates, its involvement in human locomotion is still open to debate (Dzeladini et al., 2014). For instance, the work of (Geyer and Herr, 2010), further extended in (Song and Geyer, 2015a), generated human-like gaits with speed modulation and steady turning motions, although they implemented only reflex pathways (i.e. without CPG).

Here, speed and heading modulations were achieved by controlling two reference inputs (v_{ref} and h_{ref}), resulting in linear or quadratic adaptations of fifteen key control parameters. The modulation of this small set of parameters resulted in drastic gait modulations. Indeed, the speed could be modulated in the $[0.4; 0.9]\ m/s$ range, as a result of both step length and frequency adaption; while the footstep landing positions and heading modulation could steer the walker in a 3D environment to follow a given path or to avoid obstacles.

Among this set of fifteen key control parameters, only four act on reflexes (the others being CPG-related). Therefore, while the recruitment of CPG networks during human locomotion remains a matter open to debate, our work showed that they can decrease the complexity of gait modulation.

Regarding transverse hip joint angle, its motion was controlled to track a reference position reproducing observations of real human walking (Courtine et al., 2006; Courtine and Schieppati,

2003). Rather than a direct angle control, the controller could possibly be adapted by mixing the CPG turning outputs with new reflex signals. For instance, the contractile length l_{ce} of the HER and HIR muscles could be recruited to provide an indirect biological measurement of the hip transverse motion, similarly to the reflex driving the TA muscle (inspired from (Geyer and Herr, 2010)).

Other possible human strategies could drive the refinement of the proposed controller. For example, humans also move the hip internal rotation during straight walking. This moves the swing leg forward and thus increases the step length (Stokes et al., 1989). This observation could be used to achieve similar leg transverse motion during straight walking. During turning motions, the transverse hip control rules could also be adapted to mainly affect the stride length of the inner limb, as mentioned in Section 10.5.2.

10.6.4 Perspectives

The controller developed in this contribution can be used in various domains, like robotics, neuroscience (investigating human locomotion), and 3D animation, in order to generate physically plausible gaits with speed and turning control (Wang et al., 2012; Geijtenbeek et al., 2013).

Currently, the biped can be steered by a human operator to navigate in a cluttered environment, while avoiding obstacles like holes (see Section 10.5.4). Additional controllers could be developed to automatically command the robot. For instance, a higher-level layer could be in charge of finding the (v_{ref}, h_{ref}) commands to reach desired footstep locations.

The controller robustness can also be improved for sharp turns, by developing new stabilization strategies. The reachable curvatures could be extended, but this is less relevant. Indeed, the controller is already capable of reaching steering radii which are approximately twice the size of the walker leg length. Below this limit, curved motion becomes unnatural and could possibly be replaced by side-stepping strategies.

As pointed in Section 10.5.1, the hip lateral position could sometimes bring the swing leg too close to the stance one, possibly causing collisions between both legs. In our experiments, this could mainly happen for extreme speed, high heading references, or during perturbed walking. A first solution would be to adapt the optimization fitness (see Section 10.3.3) by rewarding solutions without collision. However, this issue is more related to the optimization of the straight walking, as discussed in Chapter 9. As pointed in that contribution, a future perspective would be to increment the lateral hip control during swing phase to avoid collisions. Importantly, this might also change as a function of the walker embodiment (i.e. robot or human model) being used.

Last but not least, the controller developed here could be tested on a real robotic device. Indeed, all the experiments in this contribution were performed using a faithful simulation model of the COMAN platform (including its actuator dynamics and noisy torque sensing).

Despite the huge gap between human walking capabilities and humanoid robot ones, this contribution illustrates that bio-inspiration can help to progressively close this gap.

11 Conclusion

In the previous chapters, we progressively developed a neuromuscular controller capable of steering a humanoid robot in a 3D environment. Starting from a reflex-based model, the controller was progressively iterated and transformed to add new features. In particular, the addition of a central pattern generator provided new possibilities to steer the gait, resulting in speed and heading adaptation.

In this last chapter, we conclude the thesis by reviewing its main contributions. We also provide clues to answer the seven questions detailed in the introduction. Finally, we discuss the main results obtained in the thesis and present future directions to extend the work of this Ph.D. and to possibly guide the development of neuromuscular controllers for humanoid robots.

11.1 Original contributions

The main contributions of this thesis are summarized here, for each chapter.

Hill muscle model time integration (Chapter 3)

The neuromuscular model developed in (Geyer and Herr, 2010) recruits Hill-muscle models, updated by integrating (relatively to time) the muscle contractile lengths. However, this integration requires to use small and/or adaptive time steps, due to the stiff and strongly non-linear state equations, which are not always compatible with the controller real-time constraints. In this thesis, we solved the problem by iterating several times the muscle equations during each controller step call. However, this solution does not work on embedded controllers with limited capabilities.

Therefore, we studied the effects of neglecting the muscle dynamics, by replacing each muscle contractile length by its steady-state value obtained when considering constant joint position and activation (i.e. signal related to the neural stimulation). Moreover, we compared three

methods to compute these steady-state approximations.

It appeared that the three approximations could be used for muscles with fast dynamics. For slower muscles, the full equations (i.e. with time integration) do usually not generate numerical issue and could thus be used. In fact, for the extreme case of slow muscles integrated on controllers with a small frequency (i.e. long time between two iterations of the main loop), we developed a solution combining the steady-state approximations and the full dynamics. Interestingly, when tested on COMAN, the steady-state approximations preserved the walking gaits for controllers with a time step call of 3 ms or less.

Experimental validation of a reflex-based walking controller (Chapter 4)

We ported the reflex-based neuromuscular model of (Geyer and Herr, 2010) to a real device. While this controller was already extensively studied in simulations, we brought it to a real full-body humanoid robot. In contrast to studies limited to simulation environments, new constraints appeared on the real device: (i) working with real hardware required to cope with the world non-idealities, (ii) the implemented reflex-based model did not include lateral balance; and (iii) the experimental procedure was more likely to damage the robot and more difficult to automate.

To solve these constraints, we first optimized the gait in a simulation environment with actuator dynamics modeling. Then, we ported it to the real device, without further re-tuning of the controller parameters. The main issue was however related to the lack of lateral balance because the gait was tuned in a 2D simulation environment. In other words, lateral constraints were applied in simulation, which was not possible to perfectly reproduce on the real device. Therefore, we developed a specific upper-body controller capable of providing lateral stability with a limited (ideally non-existent) effect on the sagittal plane.

Using this framework, the robot managed to perform a 50 steps walk experiment, when lateral stability was provided from an external operator thanks to this specific upper-body controller. The gait exhibited human-like features, but appeared to be rather different from the one optimized in simulation, in particular regarding the walker speed. Despite this discrepancy between simulation and reality, the robot still managed to walk. This stresses the fact that these neuromuscular controllers have some robustness to non-modeled effects like joint friction.

Feet with human-like compliance (Chapter 5)

Humanoid robots are usually equipped with rigid feet, in contrast to the flexible feet of humans. Indeed, these robots generally exhibit very different gaits than humans, as most of them try to maintain their feet mostly parallel to the ground. In this context, rigid feet are more adapted. However, this is not the case for the human-like gaits targeted by this thesis.

Therefore, we wanted to know if the neuromuscular controllers could use a similar flexibility to possibly improve their gait. Because feet prostheses are designed to help humans regain natural walking, we carried out experiments with child feet prostheses (in order to match the robot size, similar to the one of a five-year old child). More precisely, we extracted the physical characteristics of these prosthetic feet and modeled them in simulation.

After re-tuning of the controller parameters, the gaits obtained with these prostheses were compared to the ones equipped with rigid feet. It appeared that the prostheses helped the feet to adapt to rough ground, therefore resulting in more robust gaits. However, the gaits equipped with these prostheses consumed more energy.

Forward speed modulation during 2D walking gaits (Chapter 6)

Gait modulation is at the heart of this thesis. We developed it mainly through the inclusion of a central pattern generator (CPG). Indeed, CPGs feature very interesting properties, among which the possibility to adapt the gait by modulating a minimum set of parameters, and to recover previous limit cycles after external perturbations.

We first developed this gait modulation for the COMAN humanoid platform, during walking experiments in a 2D simulation environment. The CPG was mainly in charge of controlling the proximal muscles (i.e. close to the hip), while the distal muscles (i.e. close to the feet) were controlled by reflexes. This proximo-distal gradient control scheme thus offered to adapt the gait through proper hip control, while the reflexive control of the ankle mainly adapted the foot to the ground external interactions.

Interestingly, the gait could be modulated by adapting only five parameters as linear functions of the target speed. Among these parameters, only the torso reference angle was recruited by the reflex rules. Indeed, the four other parameters all impacted the CPG control. As a result, the gait could be continuously modulated in a range from $0.4\,m/s$ to $0.9\,m/s$, thus corresponding to normal human walking once scaled to the size of the robot.

Forward speed modulation during 2D running gaits (Chapter 7)

We further extended the CPG control to running gaits (still for 2D scenarios). To achieve this purpose, we first added some compliance to the feet with a torsion spring connecting two rigid plate, in order to roughly reproduce the effects of the toes during foot push-off.

As for walking, the CPG mainly affected the proximal muscles, while the distal muscles were only driven by reflexes. However, in contrast to all the gait modulations developed in this thesis, the speed adaptation also impacted some parameters related to the ankles and the knees, in order to better propel the biped for the flying phases.

Using this neuromuscular control, the COMAN was capable of running while continuously adapting its speed in a range from 1.3 m/s to 1.6 m/s. During short periods, the speed could reach a maximal value of 1.7 m/s, but this faster speed appeared to be less stable.

Bio-inspired balance controller (Chapter 8)

We developed a framework capable of automatically learning appropriate neural stimulations, in order to resist to random pushes while standing upright. This work was initially based on the full-body compliant impedance controller introduced in (Hyon et al., 2007). The original contribution of this thesis was to automatically learn muscular stimulations capable of reproducing the effects of this impedance-based controller. However, the proposed framework can be adapted to other tasks, as long as torque references are provided during the learning process.

The approach was the following. First, torque references were decomposed into muscular forces by means of a linear optimization, in order to solve the over-actuation of the musculo-skeletal model. Then, an inverse muscular model was in charge of finding appropriate neural stimulations reproducing similar forces at the muscle level. Finally, a regression engine was recruited to learn the stimulation patterns (based on sensory information), so that an appropriate muscle control could be achieved without the impedance-based controller.

Interestingly, the approach was capable of controlling the center of mass (COM) position to follow a given reference. Therefore, it was used to initiate the gait in the 3D scenarios of Chapters 9 and 10 by moving the COM on top of one foot. In this way, the other foot could be lifted from the ground without causing the biped to directly fall laterally.

Forward speed modulation during 3D straight walking gaits (Chapter 9)

We further extended our neuromuscular walking controller to 3D scenarios. In other words, no constraint was applied in the simulation environment, so that the motion was not limited to the sagittal plane. This mainly requested to develop an appropriate lateral balance controller.

We first incremented the musculo-skeletal model with the inclusion of new muscles, so that all the biped joints were controlled by virtual Hill-type muscles. In particular, new muscles were added to the upper-body joints, together with appropriate stimulation rules. The leg sagittal muscles control was mainly kept intact (i.e. similar to the one developed for the 2D scenarios), even if some small alterations were done. The most important part was the development of appropriate rules to command the leg lateral and transverse muscles.

We adapted the forward speed parameters modulation by changing nine parameters as linear or quadratic functions of the target speed. Among these parameters, seven were in charge of controlling the CPG frequency and output scaling, while the two remaining parameters were related to the reflex rules. The resulting walking gait could be continuously modulated from

0.4 m/s to 0.9 m/s, so in a range similar to the one obtained for the 2D case. Finally, we also showed that the biped was robust to different perturbations during blind walking, as it was capable of resisting to random pushes and could walk while facing small stairs, slopes and irregular grounds.

Steering control in a 3D environment (Chapter 10)

Finally, we extended the 3D straight walking gaits to control the heading direction. This was achieved by introducing a new scalar input: the heading reference. On top of the steering direction (i.e. left or right), this input also controlled the motion curvature.

Similarly to the 3D straight walking, the transverse hip muscles were controlled so that the hip transverse joint angle could follow a reference angle. However, this reference angle was adapted according to the requested motion, in contrast to the constant value used for straight walking. In particular, this reference was updated through CPG control, in order to synchronize the transverse hip motion with the gait. On top of this, the hip lateral control was also adapted to the turning motion, in order to move laterally the COM and to generate accelerations towards the center of the curve.

Six parameters were updated as linear or quadratic functions of the target forward speed. Two of them were related to reflexes, the others impacted the CPG control. With this framework, the walker could follow circular paths whose minimal radius was equal to 1.26 m.

11.2 Questions and answers

In the introduction, we detailed seven main questions to be addressed in the context of the Ph.D. (see Section 1.5.2). We review them in light of the experiments performed in this thesis. Note however that the answers provided here are not definitive, but mainly give elements to guide future developments of neuromuscular biped controllers.

A *Are neuromuscular controllers viable solutions to control the locomotion of humanoid robots ?*
In Chapter 4, we ported the reflex-based neuromuscular model of (Geyer and Herr, 2010) to the real COMAN platform. Because lateral stability was not implemented yet, we developed a specific upper-body controller to let an operator provide stability in the lateral plane only. Using this framework, the robot managed to perform a 50 steps walk experiment. On top of that, its gait exhibited stretched legs and foot roll at some points of the gait, two human walking features hard to achieve with most robot controllers.
Yet, it is important to note that the resulting gait was quite different from the one optimized in simulation. In particular, the leg was not stretched at the end of the swing phase. This was probably due to the different setup used to provide lateral stability (i.e. constraining the lateral motion of the waist in simulation or providing stability at the writs level on the

real robot) and to joint friction at the knee level (absent from the simulation environment). This impacted the step length and frequency, so that the resulting speed was drastically reduced (i.e. 0.4 m/s to 0.2 m/s). However, despite this huge gait difference, the robot was still capable of walking, therefore demonstrating some robustness when transferred from a frictionless joint model simulator to real hardware, without any controller re-tuning.
Because these first hardware result are promising, they call for future developments. In particular, the knee non-stretching issue could be solved with new knee muscle activations, while the 3D walking gaits of Chapters 9 and 10 could be tested, removing the need for an external operator to provide lateral balance.
On top of that, we also conducted many experiments in simulation, resulting in robust gaits with steering capabilities. Because the biped modeling in simulation was done to reduce the reality gap, these results also tend to show that neuromuscular controllers are viable solutions to drive the locomotion of humanoid robots.
This contribution thus illustrated that it is possible to take advantage from human walking mechanisms to develop new robotics gaits. In particular, we pointed out interesting features compared to more traditional approaches relying on inverse kinematics/dynamics: fast computational rate, and similarities to human gait features like stretched leg during stance phase, foot roll and higher waist position, which could potentially lead to more energy-efficient robots.

B *How do the biped gaits adapt on rough terrain and when subjected to unknown perturbations ?*
The biped robustness was mainly quantified in Chapters 5, 8, 9 and 10. When studied during blind walking, it appeared that the biped could resist to perturbations, despite the lack of higher commands modulation. Importantly, the gait was not optimized to resist to these perturbations.
There is thus an intrinsic robustness coming from the viscoelastic muscle properties, and their corresponding drive signals (i.e. CPG and reflexes). However, other strategies are needed to resist to larger perturbations. This is one of the main topic of Chapter 8. In that chapter, stimulations could be automatically learned to resist to random pushes, as a function of sensory information about the center of mass (COM) and joint positions.
As shown in Chapter 9, some grounds could affect the biped gait without leading to its fall. For instance, rising slopes reduced the step length of the walker, and so its forward speed. This gait adaptation resulted from the intertwining between the neuromuscular controller and the environment, and was thus not commanded by an external operator. Interestingly, when recruiting a CPG, the biped could recover its previous gait a short time after the perturbation disappeared. This capability to return to the previous limit cycle is mainly related to the CPG entrainment. This was observed for both rough terrain and external pushes.

C *How can we modulate the biped gait by means of a minimal set of high-level parameters ?*
This question is central to Chapters 6, 7, 9 and 10. In these chapters, the speed and/or the heading of the robot were modulated by adapting a reduced number of parameters as linear

or quadratic functions of a scalar input (i.e. either the speed or the heading reference). Among these parameters, most of them were related to the CPG. In particular, the gait frequency was adapted by modulating the CPG time constant (affecting the CPG frequency), which also impacted the gait synchronization with feet strikes. Muscle stimulations were mainly adapted by scaling the corresponding CPG stimulations. Finally, a reduced number of parameters were related to reflexes, like the torso target orientation.
Except for running gaits, the parameters being modulated all affected the proximal muscles, i.e. the ones close to the hips. In contrast, the distal muscles were mainly driven by reflexes, in order to adapt to the ground external forces. In sum, gait steering was mainly controlled by the proximal muscles, while distal ones were generally kept intact.

D *Which range of motion can be achieved with proper parameters modulation ?*
During walking, we managed to continuously adapt the biped forward speed in a range from 0.4 m/s to 0.9 m/s, both for the 2D and 3D scenarios (see Chapters 6 and 9). Once scaled to the size of the robot, this corresponds to the normal human walking speed range. Interestingly, these speed transitions are quite fast as it is possible to go from one speed extremum to the other in less than two strides (both for accelerations and decelerations).
As presented in Chapter 7, the forward speed of running gaits could be continuously adapted from 1.3 m/s to 1.6 m/s. It was even possible to reach a speed of 1.7 m/s for a short period (i.e. steady-state gaits were not obtained at this speed). This range is smaller than the one obtained during walking, and could thus possibly be improved by refining the model and re-tuning the unknown parameters. In particular, it might be worth to achieve faster stable running gaits.
On top of this, it might also be interesting to obtain faster walking speeds and slower running speeds to help transitions from walking to running and opposite. Indeed, human speed ranges for walking and running slightly intersect (Farris and Sawicki, 2011). Importantly, the speed at which the transition from human walking to running naturally occurs is still not well defined and might be related to other factors than metabolic energy efficiency (Saibene and Minetti, 2003).
Finally, Chapter 10 extended the straight walking gaits of Chapter 9 to heading control. The change in heading was quantified while the biped was walking around a circle. The smallest radius of such a circle (corresponding to the sharpest turns) was equal to 1.26 m (equivalent to a curvature of 0.79). This radius is a bit more than twice the size of COMAN leg length. This already corresponds to very sharp turns. Consequently, achieving sharper turns is maybe less relevant as other motions like side-stepping would be more adapted. Similarly, the trajectories of human walking can be divided into two major classes: the nonholonomic ones during which the direction of motion is supported by the body orientation (e.g. straight walking) and the holonomic ones during which lateral velocities are observed (e.g. side-stepping) (Truong et al., 2010). Note however that the walker robustness was deteriorated and that leg collisions were observed during these sharp turns. Therefore, it would be more valuable to improve the robustness for these small steering radii, rather than obtaining sharper turns.

E *Can the neuromuscular controllers work with flexible feet, and how do they affect the gait in terms of robustness and energy efficiency ?*
This added compliance was studied in Chapter 5. In that chapter, real prostheses were characterized and modeled in simulation. After re-tuning of the controller parameters, the biped managed to walk with these prostheses, both on flat and rough grounds. The gaits of COMAN equipped with these flexible feet was later compared to its gaits obtained with rigid feet (different morphologies of rigid feet were tested). It appeared that the walker equipped with flexible feet was more robust when walking on rough terrains, but at the cost of a higher energetic consumption. This illustrates the classical trade-off between robustness and energy efficiency. In this case, the flexible feet thus favor robustness, which is important to move in unknown environments.

F *Which method can be designed to automatically learn neural stimulations for a specific task ?*
Designing appropriate neural stimulation rules is not trivial and might require a lot of experiments to progressively refine the model. In Chapter 8, we showed that it is possible to take advantage from other humanoid control methods, in order to automatically learn neural networks capable of reproducing the task to learn.
More specifically, torque references were obtained from a more traditional approach and later converted to muscular stimulations through an inverted muscular model. Finally, these stimulations were progressively learned (based on the robot sensory information) using autonomous learning. Two concurrent approaches were implemented to perform this autonomous learning: a cerebellar model (CMAC) and a support vector regression (SVR) algorithm. In our tests, the SVR autonomous learning produced the best results.
This was tested with an impedance controller in charge of controlling the COM position when the robot was standing on its two feet, while receiving random pushes. Progressively, the neural network was autonomously learned, so that the biped was capable of resisting to these pushes without recruiting the impedance controller. Other tasks could be similarly learned, provided correct torque references could be found.

G *Do humans recruit a central pattern generator to control their locomotion ?*
The recruitment of a CPG during real human walking is still a controversial matter (Minassian et al., 2017). In Section 1.4, we presented several studies, most of them involving individuals with spinal cord injury, supporting the existence of a CPG in the human spinal cord. However, none of these studies was capable of providing a definite answer to this question. In the same section, we pointed out that numerical simulations and tests on biped robots are useful tools to test and refine conceptual models of locomotor circuits. This includes investigations about the possible recruitment of a CPG during human walking and/or running. In this thesis, we showed that the inclusion of a CPG is valuable for human locomotion, especially regarding gait modulation. In particular, this gait modulation was achieved by adapting a reduced number of parameters as linear or quadratic functions of a scalar input (i.e. the speed or the heading reference). Thus, the inclusion of a CPG can be worthwhile to reduce the number of parameter adaptations during gait modulation. How-

ever, the work of (Song and Geyer, 2015a) presented a neuromuscular model for bipedal walking with steering capabilities, while implementing only reflex pathways (i.e. without CPG). Therefore, the recruitment of CPG networks during human locomotion remains a matter open to debate.

11.3 Discussion and future directions

In this thesis, we explored how neuromuscular controllers could be used to obtain more human-like gaits with humanoid robots, while preserving the modulation capabilities common to more traditional approaches like the ones recruiting inverse kinematics/dynamics modules. In this work, we managed to adapt the walker forward speed and its heading direction (including the steering curvature).

While being far from embracing the large panel of human motions during walking, these two gait modulations showed that neuromuscular controllers are capable of adapting the gait on-line, similarly to what is observed with more traditional walkers. In particular, these modulations were obtained by adapting a small number of parameters as linear or quadratic functions of two scalar inputs. The modulation of this small set of parameters resulted in drastic gait modulations. Indeed, the speed could be continuously modulated in a range close to the normal human one, as a result of both step length and frequency adaption; while the footstep landing positions and heading modulation could steer the walker in a 3D environment to follow a given path or to avoid obstacles.

On top of this controller, a high-level module could thus be developed to automatically compute these scalar inputs (i.e. the speed and heading references), in order to reach desired footstep locations (or inversely to avoid dangerous positions). This is particularly relevant to navigate in cluttered environments, especially if the robot is equipped with cameras to detect the obstacles. This would also require specific algorithms to extract the affordances (i.e. the possibilities of action on objects or on the environment), in order to compute appropriate footstep locations, according to the destination and to the environment.

Moreover, new control parameters could be added, for instance, to separately control the step length and height. Similar developments could also be performed to achieve other walking modulation tasks like side-stepping, stairs climbing or backward stepping. Interestingly, (Song and Geyer, 2015a) managed to obtain other walking patterns, also by recruiting a neuromuscular model (in that case, only driven by reflexes).

Designing new gait modulations is not trivial, due to the lack of straightforward mathematical background. Such developments usually require many iterations to progressively refine the stimulation rules for a specific task. In this respect, the inclusion of a CPG can help to develop these new modulations, in order to better synchronize the motion with the gait. Indeed, most walking alterations must be performed at a precise timing, which can be provided by the CPG. This strategy was for instance used to update the transverse hip joint angle during steering,

therefore ensuring that the leg rotation and realignment happened at the right moment (see Chapter 10).

Another strategy to develop new stimulation rules is to use the inverted muscular model and the learning process developed in Chapter 8. More precisely, the following process could be used. First, torque references must be obtained, corresponding to the desired task (e.g. stairs climbing). These references could possibly be acquired from human data. Yet, it is unlikely that directly using these human torques in a simulation environment would result in the desired motion (due to the reality gap and to the non-perfect human torque measurements). Instead, pre-existing controllers could be recruited to provide these torque references (similarly to the impedance controller of Chapter 8). Another possibility is to use optimization methods like the ones based on optimal control (Todorov and Jordan, 2002; Todorov, 2009; Mordatch et al., 2013). Once the requested motion is obtained in simulation, the learning process described in Chapter 8 could be used to progressively build a neural controller. Some statistical analyses could then be run on this resulting neural controller to extract its most important components, and so, to progressively simplify it. Finally, reflex rules and/or CPG networks could be developed to produce stimulation patterns matching the ones obtained from this process. This last step requires a bit more intuition but should be facilitated by the use of the developed stimulation references (possibly using a new optimization phase).

Robustness is critical to bring humanoid robots in our everyday environment. In this work, the walker robustness mainly emerged from the muscle viscoelastic properties and from their control through appropriate reflexes and CPG signals. This allowed the robot to resist to unknown perturbations during blind walking. Interestingly, when tested on real hardware (see Chapter 4), the neuromuscular controller managed to walk despite the lack of re-tuning on the real robot, resulting in a gait drastically different from the one optimized in simulation. This indicates that the neuromuscular robustness can also be found during hardware experiments.

However, larger perturbations require additional reflexes, like the ones learned from an impedance controller, as presented in Chapter 8. Yet, this chapter only studied robustness for straight postures. During walking, new strategies need to be developed for these bigger perturbations. Interestingly, Chapter 8 developed reflex rules in charge of maintaining the center of mass (COM) close to a (possibly constant) position reference. This was studied for straight postures, but could be adapted to walking in case an appropriate moving position reference could be provided (for the COM). A possible strategy consists in developing a kind of internal model of the biped. In other words, a simulation model of the robot could be run inside its controller, mimicking the robot locomotion. At precise moments during the gait (e.g. shortly after a foot strike), the robot could reset this internal model according to its own state (i.e. physical and controller variables). From this model, it should then be possible to compute the COM position and to compare it with the one of the real biped. When there is no external perturbation, the difference (i.e. a vector) of COM positions between the robot and its internal model should have an amplitude close to zero. In contrast, this amplitude should grow in case of disturbances. Indeed, only the real walker should be affected by the perturbations.

This vector could then be used to detect possible falls, but also to trigger additional reflexes. These new reflex rules could be obtained similarly to the ones developed in Chapter 8, i.e. by computing joint torque contributions based on the COM position error. This strategy could also benefit from the development of an odometry module to compute the kinematics of the biped (especially its speed), relative to a fixed inertial frame.

Four different gait initiations were presented in this thesis. In the hardware experiment of Chapter 4, the COMAN platform was suspended above a running treadmill with its reflex-based controller already running. Then, a initial contact with the treadmill was enough to trigger walking. In the 2D walking simulation scenarios of Chapters 5 and 6, the robot was tilted forward to move the COM in the front part of its feet. The neuromuscular model was later triggered, initiating walking. In the 3D walking scenarios of Chapters 9 and 10, the COM was deported above one foot using the algorithm developed in Chapter 8. Walking was later triggered so that the other foot could initiate its swing phase. Finally, the running gaits of Chapter 7 were obtained by optimizing the initial position of the biped during a flying phase (i.e. the robot was initially not touching the ground). In contrast, gait finalization was never studied and would be worth being investigated. A possible solution would be to develop a mechanism capable of progressively switching from the walking controller to the upright standing controller developed in Chapter 8.

Also, the achievable forward speed covers normal human walking but could not reach very low speeds. Yet, this slow motion is potentially relevant for some tasks like accurate positioning. Therefore, slow walking is another interesting avenue to explore, in order to further extend the neuromuscular controllers presented in this thesis. This could be done with the aforementioned strategy, i.e. developing new stimulation rules through the recruitment of optimal control and human data, in order to obtain stimulation references which could be learned by a neural network.

In this Ph.D., we took inspiration from real human locomotion to develop our neuromuscular controllers. However, the work presented here can also be used to investigate real human walking and running. For instance, the inclusion of a CPG in our algorithms was also used to study its possible recruitment during human locomotion. No definite answer was provided to this question, but we showed that the recruitment of a CPG is valuable to decrease the gait modulation complexity.

Refining the work of this thesis according to human experiments could thus possibly offer to better understand human locomotion features. Different strategies are possible to achieve this purpose. First, an evaluation of the matching with human data (positions, torques...) could be included in the cost function (i.e. during the optimization). Yet, the comparisons of the obtained results with human experiments would be less sound because human-likeness was included in the training. Another path consists in carefully analyzing the divergences between the simulated model motion and the human literature. This would allow to better understand what generated these differences, and then to address the source of these divergences.

Refinements can be done in the controller but can also be applied on the simulation model. In this regard, using the model of a realistic human is relevant, as well as improving the musculo-skeletal model. In particular, the virtual muscle groups are extremely simplified compared to the complex human muscular system. For instance, the virtual muscles implemented in this thesis only act in a single plane, in contrast to real human muscles. In turn, these developments could probably help in the design of prostheses, ortheses and exoskeletons. Moreover, this could also be used in the fields of rehabilitation robotics and surgical operations related to body deformities affecting walking.

Last but not least, the controllers developed here could be tested on a real device, similarly to the hardware experiment of Chapter 4. These experiments could be performed with COMAN, but also with other humanoid robots whose joints can be controlled to track torque references. Note however that the gait parameters would need to be re-tuned to adapt the controller to different robot morphologies. These experiments could also possibly refine the simulation model to close the gap between simulations and reality. This would offer to better optimize the real gait, and thus to improve the walking performances on the real device. Many developments of this thesis could be tested on a real robot, like the capability to steer the gait on-line by modulating the speed and/or heading references. In particular, the 3D developments of Chapters 8, 9 and 10 are worth being tested on the real robot as they do not require to artificially constrain the robot motion to stay in the sagittal plane.

Many other developments could be performed to further extend the results presented in this thesis. Some of them were detailed at the end of the different chapters, while we only presented the most promising ones in this last section. Despite the important gap between robots and humans in terms of locomotion capabilities, this thesis showed that neuromuscular controllers hold the potential to make a step towards the development of more efficient robotics walkers, in particular regarding robustness, energy consumption and locomotion richness. Indeed, the gaits obtained with these neuro-muscular controllers are closer to the human ones, and so potentially more adapted to our surroundings. In the future, robots might thus be able to adapt to our environment, rather than us having to adapt our environment to the robot limited skills.

A Robotran simulation environment

Most results in this thesis were obtained in simulation, using the Robotran software (Fisette and Samin, 1993; Samin and Fisette, 2003; Docquier et al., 2013). It is a symbolic environment for multi-body systems (MBS) developed within the Université catholique de Louvain (UCL), inside the Center for Research in Mechatronics (CEREM).

This appendix provides a global overview of the simulator and list publications relevant for the particular implementation of COMAN. New features developed within this thesis for Robotran are later presented.

A.1 COMAN model in Robotran

Publications

The implementation of the COMAN humanoid robot in the Robotran simulator is notably addressed in these two papers (conference paper, later extended to journal):

Zobova AA, Habra T, Van der Noot N, Dallali H, Tsagarakis NG, Fisette P and Ronsse R (2017) Multi-physics modelling of a compliant humanoid robot. Multibody System Dynamics, 39 (1-2), pp. 95-114. DOI: 10.1007/s11044-016-9545-4.

Zobova AA, Habra T, Van der Noot N, Dallali H, Tsagarakis NG, Fisette P and Ronsse R (2015) Multi-physics modelling of a compliant humanoid robot. In: ECCOMAS Thematic Conference Multibody Dynamics 2015, Barcelona, 29 June-02 July 2015.

My contributions in these publications mainly include the development of the initial framework, as well as guidance and help in the development of the simulator, in particular for the contact model implementation. I also helped during paper writing for revisions.

The developments related to the specific implementation of COMAN can be found in (Dallali et al., 2013; Zobova et al., 2015, 2017). A special attention was given to reduce the reality

gap between simulation and the real robot, especially regarding the motor dynamics and the ground contact modeling.

A.2 A symbolic generator

Robotran is a symbolic generator, which means that it generates the analytical form of the motion equations, for a given multi-body system. Interestingly, this approach allows drastic simplifications, such that simulation time can be five to ten times faster than a purely numerical computation (Docquier et al., 2013).

In our case, the COMAN model includes 29 degrees of freedom (DOFs) and uses a fourth order Runge-Kutta integration scheme with a $250\,\mu s$ time step (i.e. 16 evaluations for $1\,ms$), when being used in a 3D environment. As presented in Chapter 9, an average time of $307\,ms$ is required to simulate $1\,s$ on a quad-core Intel(R) Core(TM) i7-4790 CPU, 3.6 GHz and 16 Go RAM (using a single core). Such speed would probably not be reached for such a small time step without this symbolic approach.

Another important feature of Robotran is that it recruits relative coordinates to represent the multi-body systems (MBS). In other words, a minimal set of relative coordinates is used to determine the state of the MBS, therefore removing many algebraic constraints. Consequently, this can improve the model integration accuracy, as shown in (Zobova et al., 2017).

The description of the multi-body system is stored in an XML file whose extension is *.mbs* (called *MBS file*). The *MBsysPad* graphical editor can be used to design such a model (body masses, inertia, anchor points...). In Figure A.1, the model of COMAN in this graphical editor is presented.

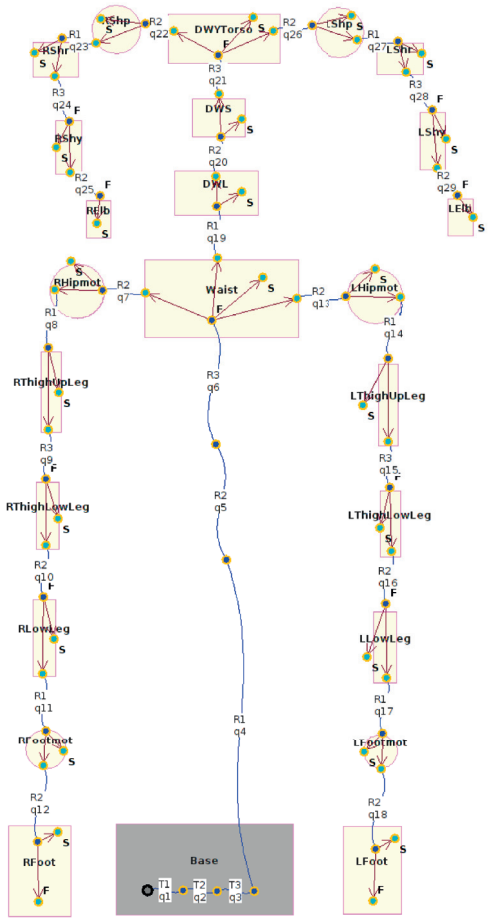


Figure A.1: MBS model of the COMAN robot, in the *MBsysPad* graphical editor.

More information about Robotran is available on the Robotran website[1]. The remaining parts of this Appendix detail new features added to Robotran since 2013, which are thus available

[1] http://www.robotran.be/

for any project. More specifically, only the modules for which I have made a significant contribution are presented. All these modules are developed for the C/C++ version of the simulator. In fact, Robotran was mainly developed for Matlab applications in 2013 (when starting this thesis), i.e. only basic C/C++ usage was possible at that time.

A.3 Project configuration through CMake

C/C++ projects combining different modules and libraries (some of them being introduced in the next sections) can be difficult to configure, and highly dependent on the operating system (OS). Therefore, a tool capable of automatically building the project on different OS is a valuable feature.

CMake is an open-source, cross-platform family of tools designed to build, test and package softwares[2]. Using it, Robotran can create a project on any of the three main OS (Linux, Windows and Mac OS). Different options are available to configure the project according to the user needs, and to the requested features. Interestingly, the code developed on one computer can then be shared with other collaborators without a single code alteration, even if they are using another OS.

Efforts are also currently being devoted to subdivide the whole project into smaller libraries, corresponding to the different features (MBS files loading, numerical functions, real-time functions...).

A.4 Pseudo real-time

Initially, no user interaction was possible during the simulation. More specifically, the process required to first simulate the MBS, before extracting the results to analyze them (as text files), or to visualize a 3D animation, in post-process.

While this feature is still available, Robotran can now also run simulations in pseudo real-time (i.e. real-time constraints are not guaranteed), while the user can interact with these simulations. This feature is mostly implemented using the *Simple DirectMedia Layer* (SDL) library[3].

The implementation is mainly performed as follows. The simulation time is compared to the real elapsed time, in order achieve correct synchronization between the simulation and the real time. More specifically, before reaching the next time when the visualization frame is expected to be updated, the simulation runs as fast as possible. Then, the simulation process is released until this time constraint is fulfilled. Therefore, the simulation can run at the same speed as the real time, provided the central processing unit (CPU) is capable of performing

[2] https://cmake.org/
[3] https://www.libsdl.org/

the requested computations in time (otherwise, the simulation is slowed down).

The user can also control the simulation time, by slowing down the simulation speed, speeding it up (up to the CPU limits), or temporarily stopping it. It is also possible to visualize previous simulated moments, as described in Appendix A.5.

To control the simulation, a graphical window (visible on the left part of Figure A.2) is generated (using to the SDL library), together with the 3D animation. When the focus is put on this SDL window, the user has access to many controls with the mouse or the keyboard. Most of these commands are summarized at the bottom of this window. Joysticks interactions are also possible (with or without focus on the SDL window). Finally, the user can increment the default simulation commands (simulation break, speeding up...) with its own commands. This is potentially relevant to control a robot in real-time.

A.5 Signals temporal evolution

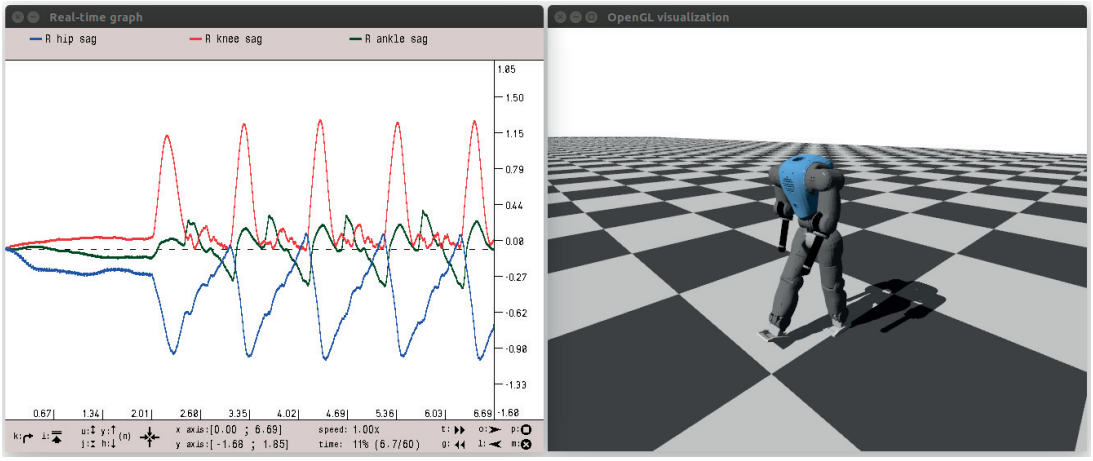
Extracting various signals (joint positions, torques, custom controller variables...) is an important feature of most projects.

During a simulation run, a new tool (using the SDL window introduced in Appendix A.4) can be used to visualize the evolution of any signal in real-time (i.e. their value being displayed and updated in real-time). This can be seen in Figure A.2a, where the signals corresponding to the positions of the right leg sagittal joints are updated in real-time. The evolution of these signals is displayed simultaneously with the 3D animation (see Appendix A.6 for more information about this 3D visualization).

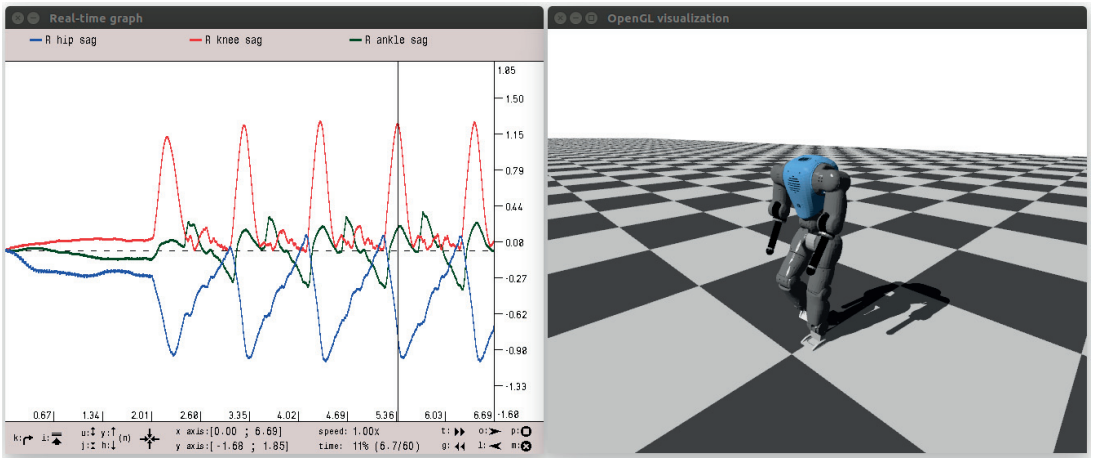
The real-time plotting features can be configured by calling a dedicated function in any file. During the simulation run, the 3D animation and the signals temporal evolution are recorded. After a break in the simulation run, it is therefore possible to visualize the past motion and the related plots, see Figure A.2. For a humanoid robot, this feature is potentially relevant to analyze the reasons leading to a fall.

During a break, different manipulations can be done on the real-time graphs, by using the mouse or the keyboard (or a combination of both). This is illustrated in Figure A.3. The mouse can be used for graph translations and zooms (in/out). Moreover, the zooming feature can be constrained to act only on one axis (X or Y) or on both axes at the same time. Keyboard commands allow to apply the same translations and zooming features.

On top of it, special keys can be used to automatically adapt the ordinate axis limits or to scale all signals in a range similar to the first one. In Figure A.4a, the right hip sagittal joint evolution can hardly be compared to the length of the contractile element (l_{ce}) of the right HFL muscle (because of the large difference in their range magnitudes). In Figure A.4b, these two scaling features are automatically applied (by pressing on the dedicated keys): the limits of the Y axis are modified and the l_{ce} signal is scaled so that its range is similar to the one of the hip



(a) Simulation running up to time $t = 6.69\,s$



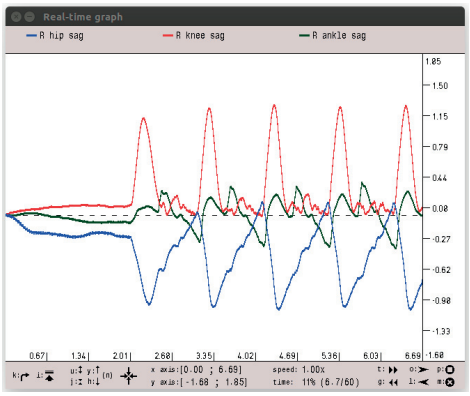
(b) Past analysis of the graph and 3D visualization, at time $t = 5.4\,s$

Figure A.2: In both panels, the evolution of the right sagittal joints (hip in blue, knee in red and ankle in green) is displayed on the left, while the corresponding 3D visualization is visible on the right. Graphs are updated in real-time when the simulation is running, before a break is called (panel (a)). In panel (b), the user goes back in time, with the 3D visualization reflecting the state of the multi-body system at that time. The vertical black line in the graph indicates the corresponding moment (right knee maximal flexion here).

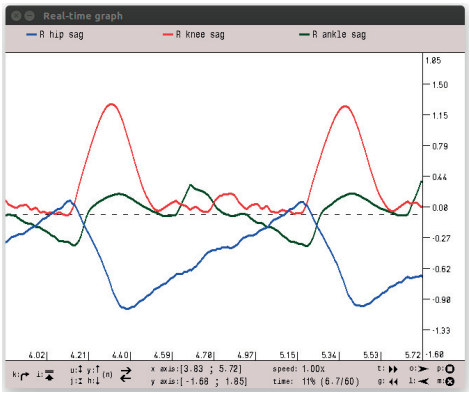
joint. The scaling applied to the l_{ce} signal is written on the upper part of the SDL window. Correlations between these two signals can then be better analyzed.

All these commands are summarized at the bottom of the SDL window. More information is available on the Robotran real-time features tutorial[4].

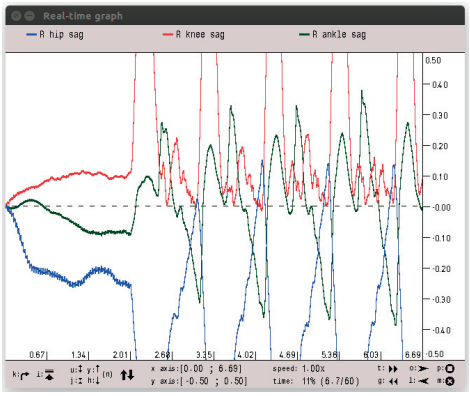
[4] http://www.robotran.be/tutorial/realtime/



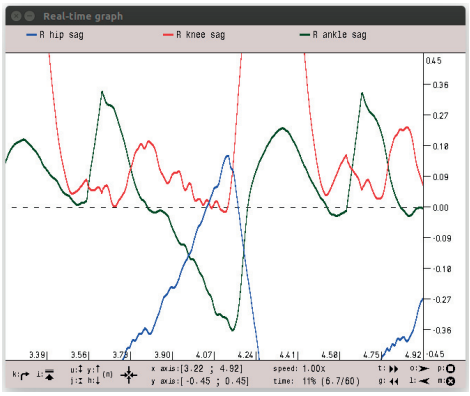
(a) initial graph (no zoom)



(b) zoom along the X axis

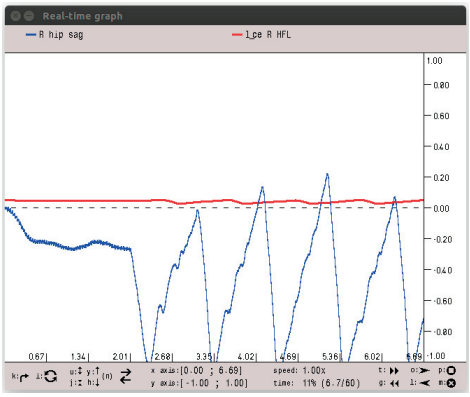


(c) zoom along the Y axis

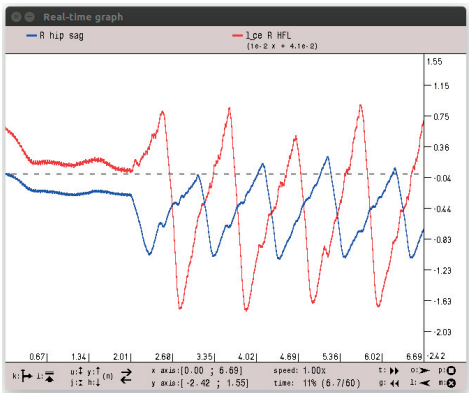


(d) zoom along both the X and Y axes

Figure A.3: Using the mouse or the keyboard keys, the user can zoom/dezoom on some parts of the graphs. Starting from an initial graph (panel (a)), it is possible to zoom only along the X axis (panel (b)) or the Y one (panel (c)), or along both axes at the same time (panel (d)).



(a) no scaling



(b) automatic scaling

Figure A.4: The sagittal hip joint position (blue) and the length l_{ce} of the HFL muscle (red) are displayed without scaling in panel (a). In panel (b), the following scaling is applied on the red signal: $10^{-2}x + 4.1 \cdot 10^{-2}$, so that its evolution can be compared to the one of the blue signal.

A.6 OpenGL 3D visualization

Initially, the only possibility to visualize the 3D animation resulting from a simulation was to use the graphical Pad (*MBsysPad*, see Figure A.1), including a post-process visualization tool based on a Java framework. Figure A.5 displays the COMAN robot using this Java 3D visualization. This tool is now integrated in the C/C++ project, in order to be used during real-time simulation runs.

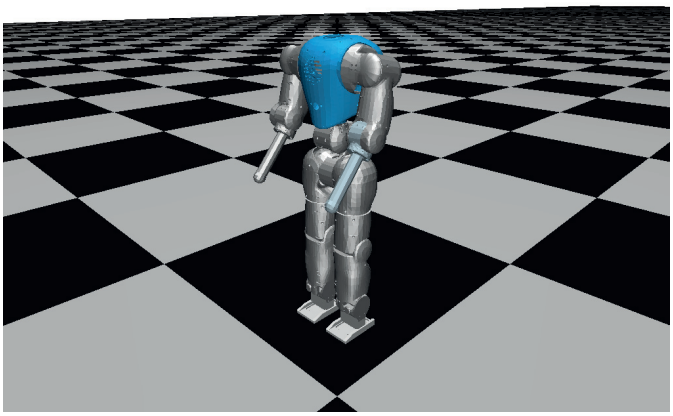


Figure A.5: COMAN visualization in its 3D environment, using Java.

However, this Java tool is quite outdated and has some compatibility issues, especially with the latest versions of Mac OS. Therefore, a new 3D visualization tool based on *OpenGL*[5] was developed and is now available. This new tool can be used either to generate 3D animations in post-process (similarly to the Java tool in MBsysPad) or to display the animation in real-time.

This OpenGL tool can represent any multi-body system (with rigid bodies), by computing the absolute position and orientation of these different bodies at each visualization frame. The multi-body system is described as a tree of rigid shapes, connected by anchor points and relative joints. This tree can be encoded manually or extracted from a MBS file, as the one of COMAN designed in MBsysPad (see Figure A.1).

Each rigid body can be made of different shapes, among which boxes, spheres, cones and cylinders. Wavefront .obj files can also be imported to describe the geometry and colors of any shape. These files can be generated by many computer-aided design (CAD) tools. The structure of the project is designed such that it can be extended to accept other file formats.

The OpenGL visualization can be configured, in order to adapt its available features to the computer capabilities, mainly dependent on its graphics processing unit (GPU). This mainly impacts the shaders, which are programs being run on the GPU to deal with different graphical effects, like body colors, lights, shadows... As illustrated in Figure A.6, four different shaders are available.

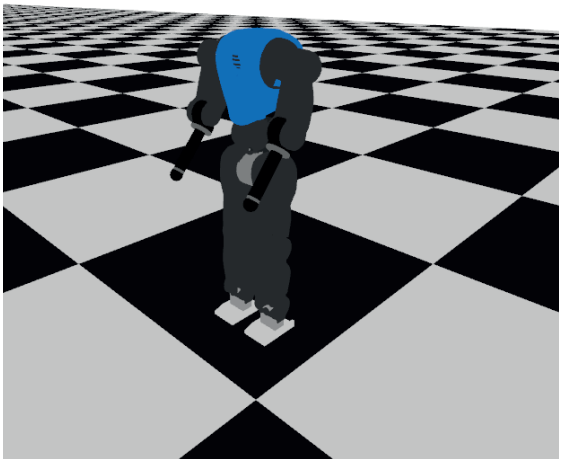
[5] *https://www.opengl.org/*

The most basic one (see Figure A.6a) only applies the body primitive color, without lighting effect. This solution is recommended for computers with low graphical capabilities. The second shader adds lighting effects, as can be seen in Figure A.6b. The next shader level adds specular effects. This effect is more subtle, and can be mainly seen in Figure A.6c, close to the COMAN left shoulder (small patches of light). The shininess and the specular color can be configured for any body. This effect is mainly relevant when dealing with metallic surfaces. Finally, the most advance shader adds shadows to the model, as depicted in Figure A.6d. The introduction of new features (and so the selection of more advanced shaders) impacts the graphical frame rate performances and mainly depends on the computer GPU capabilities. Importantly, the specular effects and shadows gestion are two features not available with the Java visualization tool.

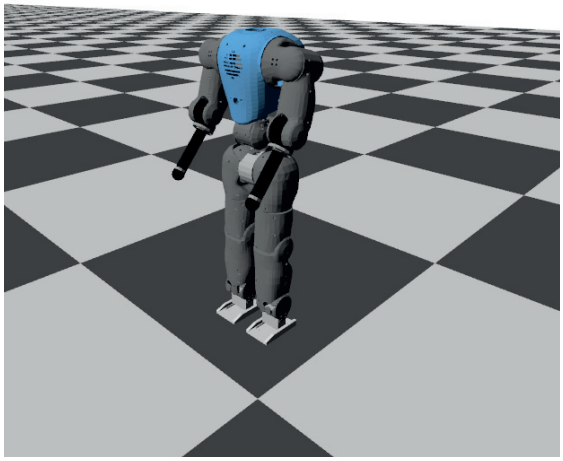
Shadows are computed using the shadow mapping method. The basic principle is the following. A specific texture (called *depth texture*) is generated from the point of view of the light source, storing the distance of the closest body point to this source. Later, a test is applied on each pixel to know if it is visible from the light source. This is done by computing its position in the depth texture, together with its distance to the light source. If this distance is greater than the one stored in the depth texture, this means that the pixel is hidden from the light by another object, in which case its color is darkened. Otherwise, the pixel is visible from the light source and keeps its color (which is still affected by the light).

The main advantage of this method is that it does not require intense graphical computations. However, each shadow must be carefully tuned. In particular, the depth textures need to be correctly calibrated. Typical problems include shadow acne (shadow fragments appearing on sunny surfaces, due to the limited resolution of the depth texture) and Peter Panning (shadows not attached to the body, the name refers to the "Peter Pan" fantasy character)[6]. All the shadow tunning parameters can be calibrated by the user, in order to prevent these issues.

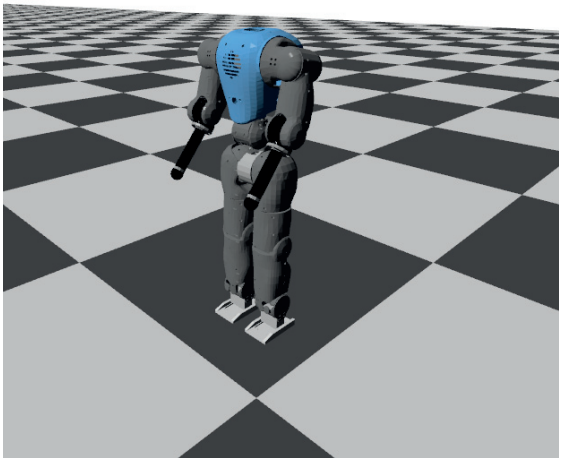
[6] for more details, see http://www.opengl-tutorial.org/intermediate-tutorials/tutorial-16-shadow-mapping/



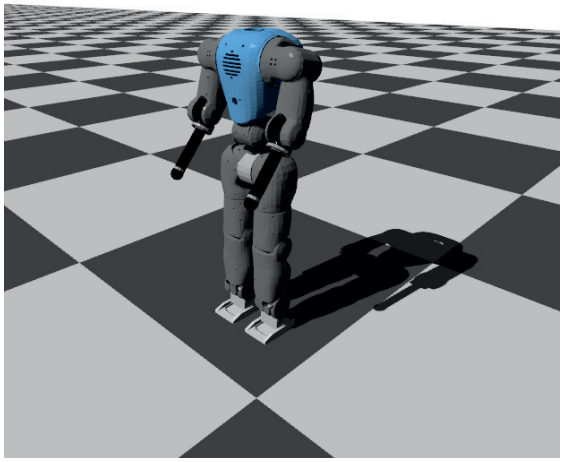
(a) basic shaders (no light)



(b) lights without specular effect



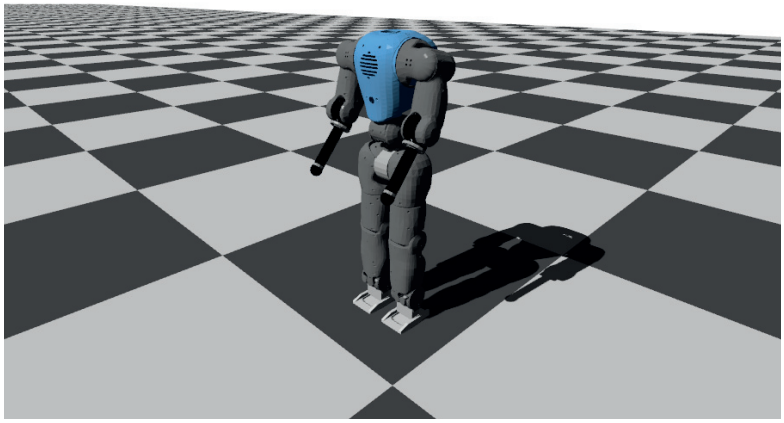
(c) lights with specular effects



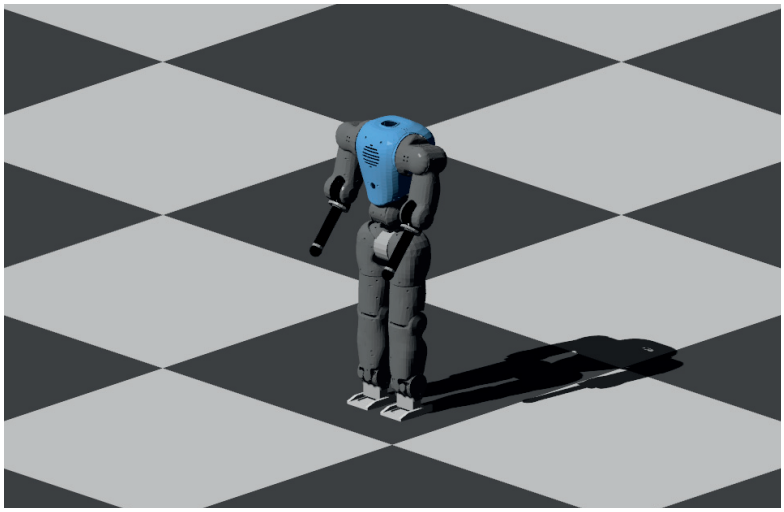
(d) lights (with specular effects) and shadows

Figure A.6: Different shaders can be selected for the whole multi-body system. From panels (a) to (d), new features/effects are added, at the cost of higher GPU computations.

To properly render a 3D scene, the viewpoint and the projection type (from the 3D scenario to the 2D computer screen) must be carefully selected. Two types of projections are implemented: the perspective projection (see Figure A.7a) and the parallel one (see Figure A.7b). Perspective projections are more realistic, in the sense that they reproduce human normal vision: objects located further away from the viewpoint appear smaller. In the parallel projection, shapes are not deformed, whatever their distance to the camera. The parallel view is especially relevant for 2D scenarios (i.e. motion constraint in a single plane). All viewpoint can be controlled with the mouse, in order to apply translations, rotations and/or zooms.



(a) perspective projection



(b) parallel projection

Figure A.7: Two types of viewpoints can be configured, depending on their projection method: (a) perspective or (b) parallel projection.

As can be seen when comparing Figures A.6a and A.6b, the lights drastically modify the rendering. Many attributes can be selected for the lights, among which their ambient component or their colors, as depicted in Figure A.8.

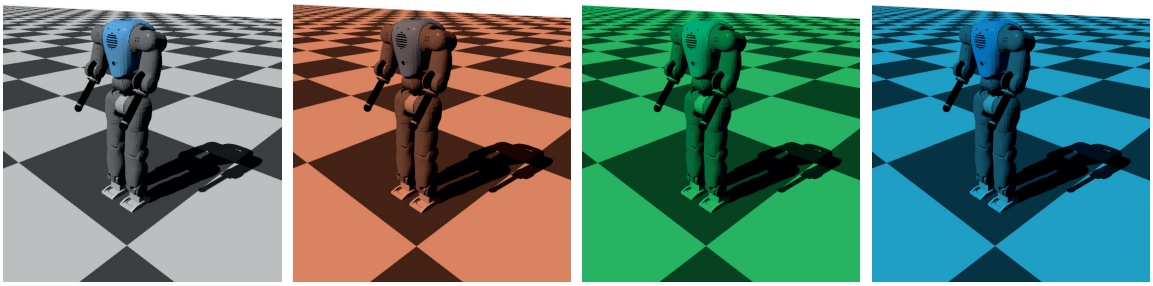


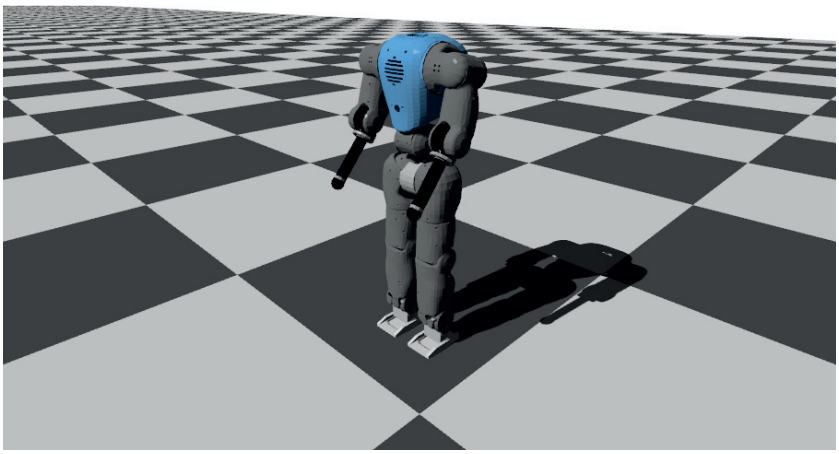
Figure A.8: Lights can be set to any color.

Three types of lights are available. The directional light projects parallel rays of light on the whole scene (see Figure A.9a). This is mainly relevant for distant light sources, like the sun. The point light emits light in all directions, as depicted in Figure A.9b. Consequently, points located closer to the light appear brighter than more distant ones. Another consequence is that shadows are distorted (contrary to directional lights). This is particularly visible in Figure A.9b, where the shadow of the upper body is much larger than the one of the lower body. Finally, the spot light is similar to the point one, except that it does not emit light outside a restricted cone (see Figure A.9c).

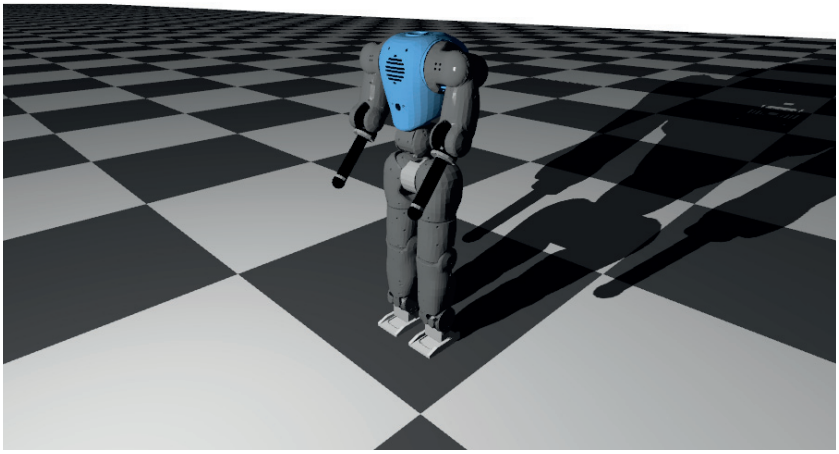
Several lights can be combined. In Figure A.10a, the combination of two directional lights generates two brighter shadows. This is due to the fact that each shadow is mostly in the sunny area covered by the other light source. In contrast, the shadows produced by the two spot lights in Figure A.10b are more visible, due to the restricted intersection area between these two lights.

Finally, the developments done here are currently being ported to *WebGL*, a web tool based on OpenGL to generate 3D graphics from a web browser[7].

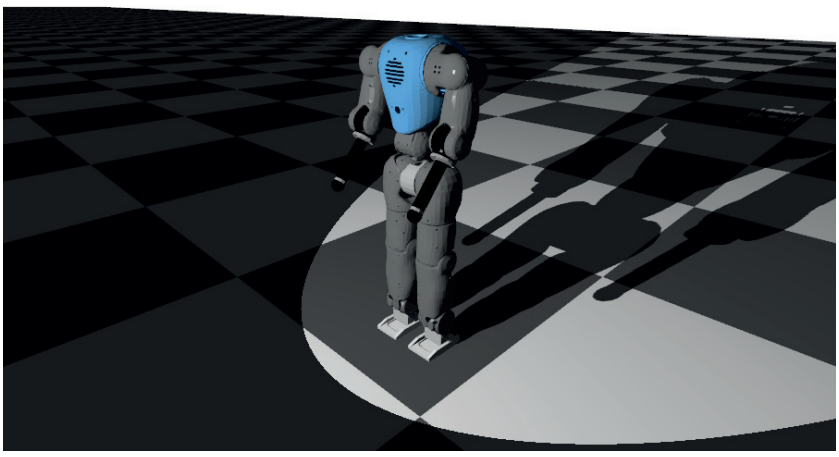
[7] https://www.khronos.org/webgl/



(a) directional light

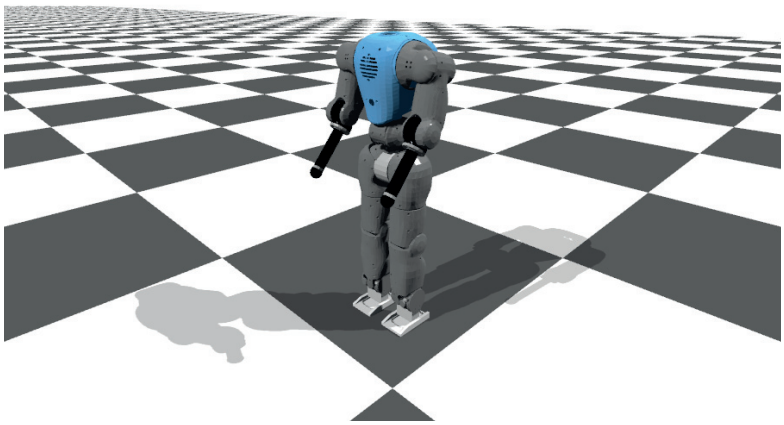


(b) point light

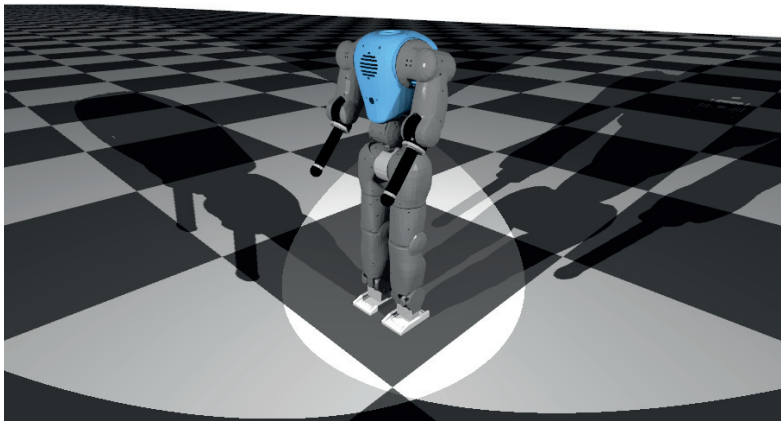


(c) spot light

Figure A.9: Three types of lights can be used: (a) directional, (b) point and (c) spot ones.



(a) two directional lights



(b) two spot lights

Figure A.10: Different lights can be combined, and so multiple shadows can emerge. In panel (a), two directional lights are used, while panel (b) displays two spot lights.

A.7 Primitive shapes contacts

This last module is not directly included in default Robotran projects, but is a library designed to be interfaced with any Robotran project. More specifically, this module computes contact forces (and resulting torques) between primitive shapes. This library was used in Chapter 9 when testing the walking controller robustness to pushes, in order to compute the contact forces between COMAN and the flying balls thrown to it (see Section 9.6.1).

This library computes contact forces and torques between three types of primitive shapes:

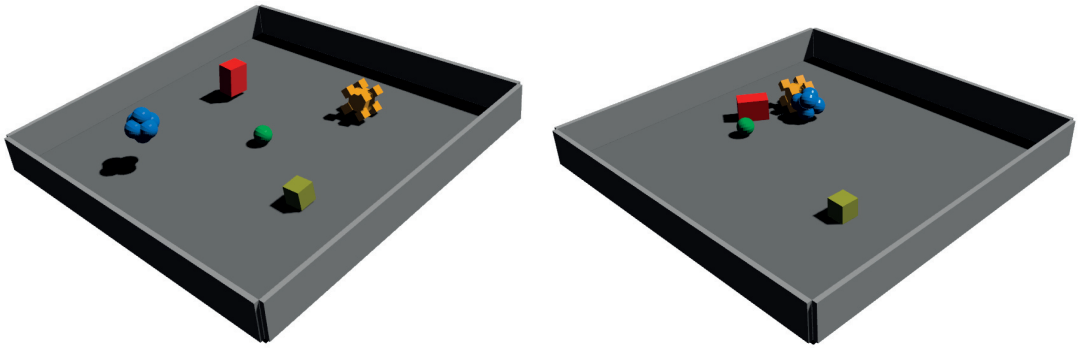
- spheres (radius defined by the user)
- boxes (width, depth and height defined by the used)
- infinite plans (like the ground)

More complex shapes can be obtained by combining these different contact primitives. The basic principle of this library is the following. When two shapes are considered, the penetration volume (i.e. the volume of intersection) is computed, together with its time derivative, based on kinematics. Normal forces are obtained as a function of these penetration measures. Then, some points at the intersection of the two shapes are selected. Their relative velocity and distributed normal forces are computed and aggregated, resulting in friction forces and torques. The contact force and torque components are finally provided to Robotran, in order to compute the resulting dynamics.

Configuration files are provided to the user, where the contact shapes (and their attachments to the different bodies) are defined. The user can also adapt the rules governing the normal contact forces (i.e. repulsive forces) and the tangent ones (i.e. friction forces), as functions of the penetration volume, its derivative and the relative motion between the bodies. For example, the contact model corresponding to the robustness experiment of Section 9.6.1 is described in Appendix G.5.2.

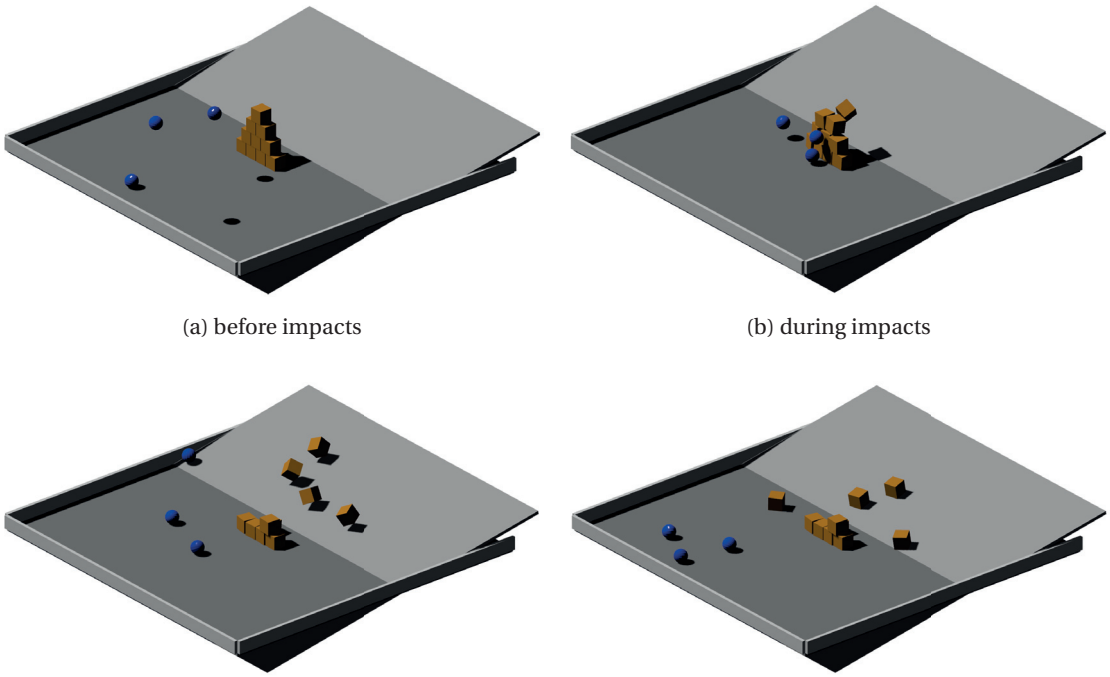
Two other examples are provided. In Figure A.11, a plane with flat boundaries is moving according to a user defined motion. On top of this plane, different shapes are released and move accordingly, as a result of the contacts between these shapes and with the moving plane.

In Figure A.12, a stack of ten boxes is destroyed with three flying balls thrown to them. In contrast to the moving plane example of Figure A.11, this experiment is computationally intensive because the compact stack of boxes involves high forces changing quickly with time. An adaptive time step strategy is therefore used, which results in a slow-running simulation.



(a) objects released on the moving plane (b) positions of the objects after after a few seconds

Figure A.11: Five different objects composed of spheres or cubes are released on a moving plane (equipped with flat borders). The initial positions of these objects is displayed in panel (a), while panel (b) displays their respective positions a few seconds later.



(a) before impacts (b) during impacts

(c) after impacts (d) boxes in standstill position

Figure A.12: Three flying balls impact a stack of ten boxes located on an horizontal ground, just in front of an inclined flat ground. The different panels show the position evolution of these balls and boxes.

B Zero-moment point compatibility

Publication

The material presented in this appendix is adapted from:

Van der Noot N and Barrea A (2014) Zero-Moment Point on a bipedal robot under bio-inspired walking control. In: MELECON 2014 - 17th IEEE Mediterranean Electrotechnical Conference, Beirut, 13-16 April 2014, pp. 85-90. DOI: 10.1109/MELCON.2014.6820512.

The neuromuscular developments presented in this thesis are far from the more traditional controllers recruiting the zero-moment point (ZMP) as an indicator of gait feasability (Vukobratovic and Borovac, 2004). In this appendix, we give a brief overview of the ZMP concept and develop a computationally efficient method to compute it. Then, we study the possibility to recruit it inside neuromuscular controllers.

More precisely, the gait generation is controlled using the neuromuscular reflex-based controller developed in (Geyer and Herr, 2010). The ZMP is computed in parallel, in order to possibly detect when the locomotion stability is deteriorated. The purpose of the current appendix is therefore to present a computationally-efficient ZMP computation, and then to investigate if the ZMP computation is compatible with the neuromuscular gaits targeted in this thesis.

Two gaits are considered here: a 2D walking gait (i.e. waist constrained to stay in the sagittal plane) and a 3D walking one (i.e. no simulation constraint), both using only reflexes (i.e. no CPG component). It is important to note that the rules controlling the lateral joints are adapted from (Wang et al., 2012) and (Yin et al., 2007), and not from our CPG-based 3D controller developed in Chapters 9 and 10.

B.1 Introduction

Simplest methods to achieve robot walking neglect the robot dynamics, leading to so-called *static walking*. In this case, stability is insured as long as the center of mass stays within the support polygon. The price to pay for this simplified approach is a drastic limitation of the maximum walking speed (Hobbelen and Wisse, 2007). Indeed, humans rely on their body dynamics to control their gait, thus allowing a higher walking speed and a lower energy consumption.

In contrast, more advanced methods focus on dynamic walking, and the most popular one is the so-called zero-moment point (ZMP) criterion, an indicator of dynamic stability (Vukobratovic and Borovac, 2004; Dekker, 2009). Many experimental validations were already conducted to perform ZMP-based dynamic walking with humanoid robots such as ASIMO (Chestnutt et al., 2005) or HRP-2 (Kaneko et al., 2002). However, there are some drawbacks associated with ZMP-based controllers: energy-inefficient walking, limited walking speed, poor resistance against external perturbations and no appropriate reaction when the equilibrium is lost (Dallali, 2011). On top of that, these methods are computationally greedy, rely on perfect knowledge of the robot parameters and of the environment, and exhibit non human-like walking features, like constant knee bending (Kurazume et al., 2005).

In particular, ZMP-based methods rely on full local controllability (i.e. each point of the gait cycle is stable), which is not necessary to ensure a stable walking gait, then leading to a higher energy consumption (Dallali, 2011). The emerging concept called 'Limit Cycle Walking' considers the gait as a limit cycle whose global stability is prevalent to the local stability (Hobbelen and Wisse, 2007), leading to more energy-efficient walkers. Bio-inspired controllers are emerging as a promising way to implement such limit cycle walking, even if they have been mainly studied in simulation so far.

In this contribution, we implement such a bio-inspired controller. More precisely, we adapt the reflex-based approach developed by (Geyer and Herr, 2010) to make a humanoid robot, namely the COMAN, walk dynamically. The purpose is then to extend it to compute the robot ZMP position in real-time, while walking in a human-like fashion. Indeed, the ZMP is a good indicator of dynamic stability. Then, even if not used as an input of the gait controller, it can detect when the robot dynamic stability is compromised, leading to the activation of additional recovery modules. In this contribution, we focus on simulation only, but keeping in mind the idea of transferring these results to the real robot. This brings up new challenges in terms of computation time (the real robot controller runs at 1 kHz) and on available inputs (using only data coming from the real robot sensors).

The two main contributions of this publication are (i) the presentation of a computationally-efficient ZMP computation and (ii) the illustration of this computation for a robot walking with a human-like gait. This type of gait induces some rough heel strikes, which are absent from most ZMP-based walking robots, thus decreasing the accuracy of the ZMP computation. We will illustrate that computing the ZMP on a bio-inspired gait is valuable to quantify the gait

stability, mainly in the lateral direction.

This appendix is organized as follows. In Section B.2, the COMAN robotic platform and its bio-inspired gait are briefly outlined. In Section B.3, the ZMP concept is introduced and in Section B.4, the efficient computation method is described. In Section B.5, simulation results obtained from two different bio-inspired gaits are presented. Section B.6 discusses the possibility of using such a method on real human-like walking robots. Finally, Section B.7 concludes the appendix.

B.2 COMAN platform and gait controller

The robotic platform used to compute the ZMP is the COmpliant HuMANoid platform (called COMAN) developed by the Italian Institute of Technology (IIT), within the AMARSI European project. The 95 cm tall COMAN has 23 actuated degrees of freedom (DOFs), equipped with series elastic actuators and position, velocity and torque sensors (Pratt and Williamson, 1995). An inertial measurement unit (IMU) is located in the robot waist and 6-DOF force and torque sensors in each ankle can measure the ground reaction forces. Further information about COMAN can be found in (Tsagarakis et al., 2013), (Tsagarakis et al., 2011), and (Dallali et al., 2013). Figure B.1 shows the basic planes used to describe the robot motion. The controller design was done in the Robotran simulation environment (Samin and Fisette, 2003; Dallali et al., 2013), accurately modeling the physics of the COMAN and its environment.

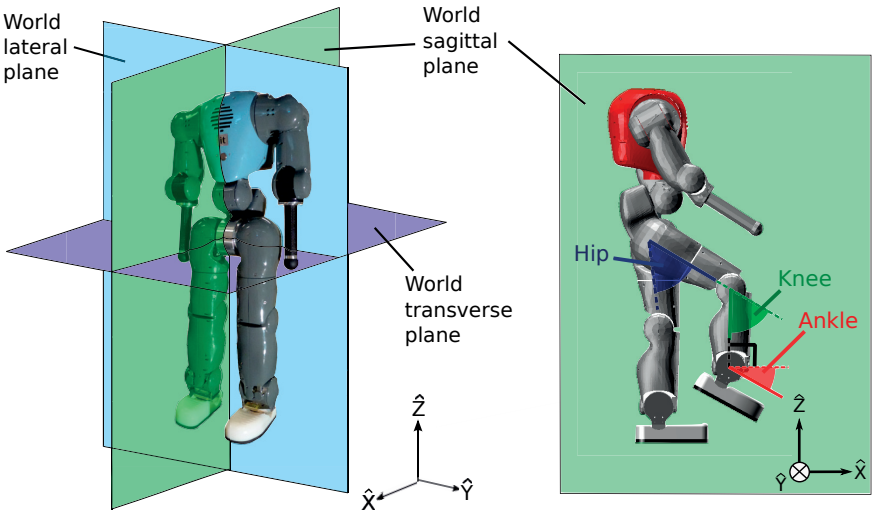


Figure B.1: Real COMAN, along with the related world planes and the inertial frame (left) and simulated COMAN in the world sagittal plane, along with the inertial frame and the angles describing the leg sagittal joints (right).

Two kinds of walking gaits are considered. First, the *2D gait* artificially constrains the robot waist to stay in the world sagittal plane. Second, the *3D gait* relaxes this constraint. In both cases, the design of the leg sagittal joints controller (6 DOFs) is based on the bio-inspired

reflex rules described in (Geyer and Herr, 2010). These are the most important joints for walking since they propel the body forward. The remaining DOFs are controlled under the rules described in (Wang et al., 2012) and (Yin et al., 2007). Finally, the whole controller is tuned via a 'Particle Swarm Optimization' (PSO) algorithm (Kennedy and Eberhart, 1995; Clerc and Kennedy, 2002).

B.3 Zero-moment point

B.3.1 Zero-moment point overview

The bipedal gait is composed of several phases. The *single support* phase happens when only one foot contacts the ground, while the *double support* phase happens when both feet are in contact with the ground. Their corresponding support polygons can be seen in Figure B.2.

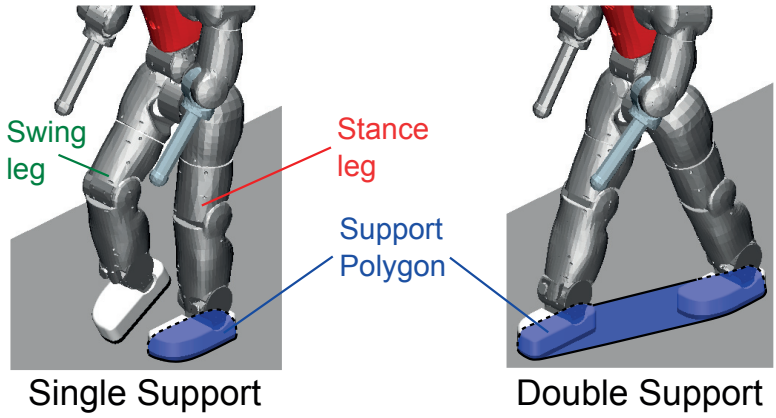


Figure B.2: Single support (left) and double support (right) phases.

For static bodies, the stability criterion is quite straightforward: the vertical projection of the center of mass (COM) on the ground must stay within the support polygon (Hobbelen and Wisse, 2007). Nevertheless, this is not true for moving bodies where linear and angular accelerations must be taken into account. This is the purpose of the zero-moment point (ZMP), which can be viewed as the generalization of the COM concept in dynamic conditions. More formally, the ZMP is the point on the ground where the tipping moment (i.e. the component of the moment that is tangential to the supporting surface) acting on the biped, due to gravity and inertia forces, equals zero (Sardain and Bessonnet, 2004a). Consequently, the body does not fall as long as its ZMP does not reach the boundaries of the support polygon. Otherwise, the robot would start falling by rotating around the edge of the support polygon where the ZMP is located. Thus, the distance between the ZMP and the nearest border of the support polygon is a good indicator of dynamic stability. The purpose of ZMP-based controllers is thus to maximize this distance, as described in Figure B.3.

The ZMP criterion is a sufficient condition for dynamic stability but not a necessary one, as it requires full local controllability (Dallali, 2011). For instance, a ZMP at the front edge of the

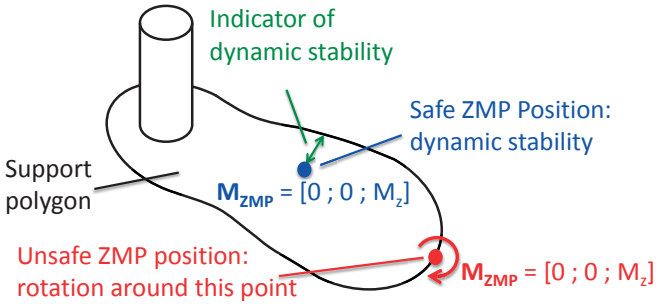


Figure B.3: The ZMP position is the point where the net moment M_{ZMP} of inertial gravity forces has only a vertical component, i.e. along the z-axis. When the ZMP lies within the support polygon (blue point), the distance to the nearest support polygon edge is an image of the dynamic stability. Otherwise, when the ZMP lies on the support polygon boundary (red point), the robot is dynamically unstable and starts to rotate around the point where the ZMP is located.

stance foot causes the body to start rotating around this edge, but then, it is possible to recover stable walking when the swing leg strikes the ground (see Figure B.4). The strict application of the ZMP criterion constrains the stance foot to remain flat on the floor at all time, decreasing the performances in terms of efficiency, disturbance handling, and natural appearance as compared to human walking (Hobbelen and Wisse, 2007).

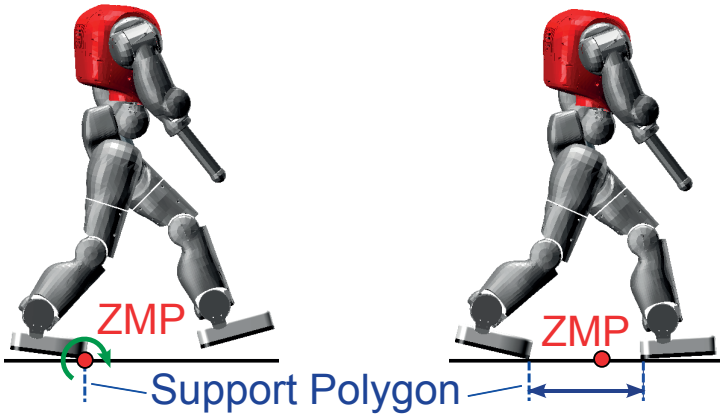


Figure B.4: On the left-hand side, the ZMP is located at the edge of the support polygon, causing the foot to start rotating around it. On the right-hand side, after the strike of the swing leg, the ZMP is no longer located at the edge of the support polygon (ZMP stability criterion recovered).

Among the numerous ZMP-based biped robots built around the world, few are provided with anthropomorphic soles (Sardain and Bessonnet, 2004a), which prevents them from walking with a human-like gait, especially during the double support phases (Sardain and Bessonnet, 2004b). Indeed, most of these biped robots walk with bended knees. This is because controlling the ZMP becomes tedious when the knee is fully stretched, i.e. when the leg is in a singular configuration (Kurazume et al., 2005). Importantly, even for limit cycle

walking, the ZMP should not be located on the left or right side of the support polygon, usually indicating a lateral fall.

B.3.2 Center of pressure

The ZMP is equivalent to another concept widely used in bio-mechanics: the center of pressure (COP). When the field of pressure forces exerted by the feet on the ground is replaced by a single resultant force, the COP is the point where the resultant moment is zero, as defined by (Sardain and Bessonnet, 2004a). Based on this definition, they proved that the ZMP and the COP are strictly equivalent in a balanced gait. Interestingly, they are computed in a different way: the ZMP is computed from the body kinematics while the COP is generally referred to as the point computed from measured forces. So, the COP can be used to validate the ZMP computation, since both should coincide. Moreover, in simulations, the COP can be easily computed as a weighted sum of the interaction forces created by the contact model between the robot feet and the ground.

B.4 Zero-moment point computation

B.4.1 Main assumptions

The ZMP computation uses forward kinematics, relying on the following assumptions (Dekker, 2009):

a) *The robot is made of n rigid links*;

b) *All time-invariant quantities (inertia…) are known:*
in simulation, these values are obtained from the CAD files of the real robot;

c) *All kinematics information is perfectly measurable:*
this is **challenging after heel strike** because of the high accelerations it induces;

d) *The floor is rigid and motionless*;

e) *The feet do not slide over the floor:* there is a **small sliding** after some rough heel strikes.

These assumptions are valid, except *c*) and *e*) after heel strike.

B.4.2 Symbolic equations

There are very tight constraints on the time allocated for the ZMP computation, as it must be fast enough to let the controller fulfil its real-time constraints. A first naive approach to compute the ZMP would be to numerically compute the kinematics of all bodies (absolute

position, velocity and acceleration) as well as the linear and angular momenta. However, this method requires solving several numerical loops, which is not efficient enough to be used in practice. In contrast, the proposed approach allows to get the exact ZMP position (i.e. without model simplification) while being fast enough in the computation.

The approach of this contribution consists in getting the *symbolic* ZMP equations (as if they were computed by hand), automatically generated by a custom Matlab script. This approach is called the symbolic approach[1] and is at the heart of the Robotran simulator (see (Samin and Fisette, 2003) for more insights). All time-independent bodies constants (masses, inertia matrices...) are gathered. Symbolic variables are then defined and the ZMP is symbolically computed. The corresponding symbolic equations (computed in the Matlab script) can further be printed in a C file which can be integrated to the robot controller. Section B.4.4 details the method used to generate this script.

To get the ZMP position, all forces acting on the COMAN are considered, namely the gravitational force $\boldsymbol{F_g}$ applied at the whole-body center of mass (COM) and the ground contact force $\boldsymbol{F_{fl}}$ (Dekker, 2009). The position vector of the COM is denoted $\boldsymbol{r_{COM}}$, $\boldsymbol{x_i^G}$ is the position vector of a single body i center of mass COM_i and r_{ZMP} is the position vector of the ZMP, as shown in Figure B.5.

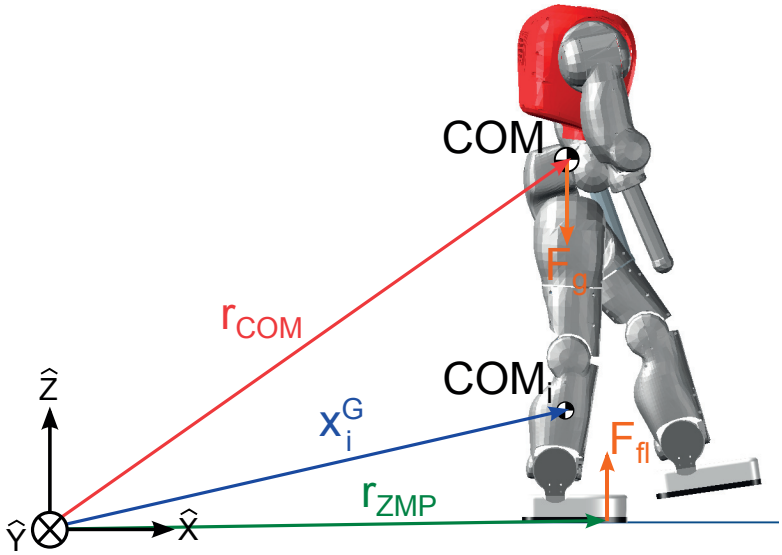


Figure B.5: The absolute position vectors r_{COM}, x_i^G and r_{ZMP} are computed with respect to the inertial frame. The only forces acting on the system are due to gravity (F_g) and contacts with the floor (F_{fl}).

Then, the net moment M_{ZMP} of inertial and gravitational forces is computed. Its x and y components (i.e. parallel to the ground as illustrated in Figure B.5) must then be equal to zero.

[1] The symbolic approach, which appeared in the eighties, is a powerful tool to drastically simplify mathematical expressions and to confer the equations a high portability towards other scientific disciplines like control, optimization, dimensioning, etc. (Samin and Fisette, 2003).

The ZMP position equations are presented in (B.1), where $\boldsymbol{r_{ZMP}} = [r_{ZMP,x}, r_{ZMP,y}, r_{ZMP,z}]^T$, m_{tot} is the total mass of COMAN, g_z is the gravitational acceleration, $\dot{\boldsymbol{N}}$ is the time derivative of the linear momentum and $\dot{\boldsymbol{H}}$ is the time derivative of the angular momentum.

$$\begin{cases} r_{ZMP,x} = \dfrac{m_{tot} \cdot r_{COM,x} \cdot g_z + r_{ZMP,z} \cdot \dot{N}_x^0 - \dot{H}_y^0}{\dot{N}_z^0 + m_{tot} \cdot g_z} \\ r_{ZMP,y} = \dfrac{m_{tot} \cdot r_{COM,y} \cdot g_z + r_{ZMP,z} \cdot \dot{N}_y^0 + \dot{H}_x^0}{\dot{N}_z^0 + m_{tot} \cdot g_z} \end{cases} \tag{B.1}$$

B.4.3 Inputs to the ZMP computation

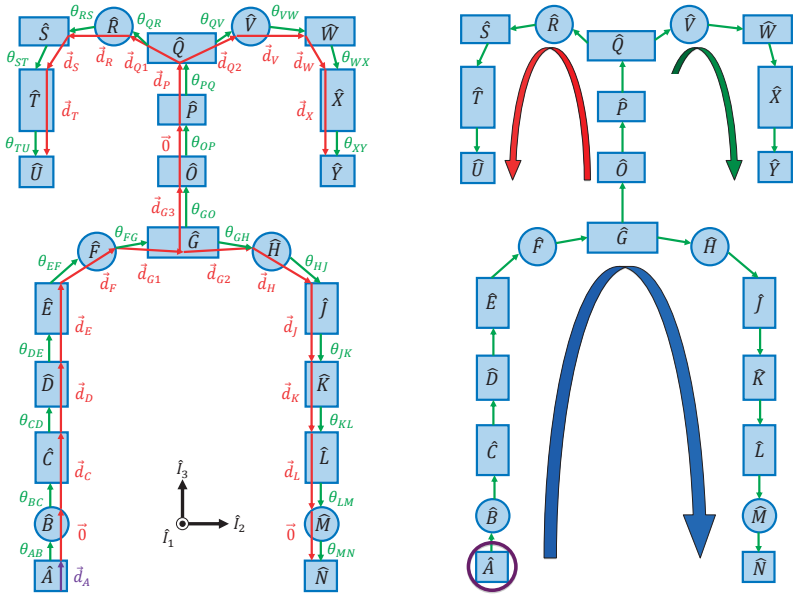
All quantities presented in (B.1) must be described relative to an inertial frame (visible in Figure B.5). There are two different ways to get an inertial frame with the COMAN sensors: (i) using the inertial measurement unit (IMU) located in the robot waist or (ii) assuming that the foot with the highest measured ground reaction force (called the *supporting foot*) is flat on the ground. Because the ZMP computation relies on sharp absolute acceleration variations, using the IMU might not be accurate enough, so the second possibility was chosen, i.e. there is always one foot assumed to be motionless and flat on the ground.

Then, the relative joint accelerations need to be computed while only the relative joint positions and velocities are accessible on the real robot. Because the relative joint velocities are quite noisy, numerical differentiation must be rejected. To filter and differentiate these signals, a third-order polynomial fitting the velocity is computed, and then analytically differentiated. Like any filter, this approach induces a small delay (about 25 ms in this case).

B.4.4 Recursive forward kinematics

To compute all the quantities in (B.1), the recursive forward kinematics method (Samin and Fisette, 2003) is used. To do so, a proper view of all body relationships in the robot is needed. The schematic on the left panel of Figure B.6 gives an overview of the 24 bodies of the COMAN with appropriate notations.

The computation of the forward kinematics starts from the supporting foot, denoted by $\hat{A}$ and supposed to be motionless. Then, a recursive path is defined (visible on the right panel of Figure B.6) to list all the bodies in a precise order: first the leg bodies (blue arrow), then the trunk and first arm bodies (red arrow) and finally the other arm bodies (green arrow).



(a) Joint and positions

(b) Forward kinematics path

Figure B.6: Left: schematic representation of the 24-bodies COMAN, along with reference frames, joints and relative positions. Right: same representation, but with the forward kinematic path being indicated.

When the forward kinematics of a body $i-1$ is computed, the forward kinematics of the next body i is obtained by computing first its absolute angular velocity $\boldsymbol{\omega_i}$ and acceleration $\dot{\boldsymbol{\omega}}_{\boldsymbol{i}}$ based on its relative angular velocity $\boldsymbol{\Omega_i}$ and acceleration $\dot{\boldsymbol{\Omega}}_{\boldsymbol{i}}$ and on the body $i-1$ kinematics, according to the following recursive equations:

$$\begin{cases} \boldsymbol{\omega_i} = & \boldsymbol{\omega_{i-1}} + \boldsymbol{\Omega_i} \\ \dot{\boldsymbol{\omega}}_{\boldsymbol{i}} = & \dot{\boldsymbol{\omega}}_{\boldsymbol{i-1}} + \dot{\boldsymbol{\Omega}}_{\boldsymbol{i}} + \boldsymbol{\omega_i} \times \boldsymbol{\Omega_i} \end{cases} \tag{B.2}$$

Thereafter, the absolute position $\boldsymbol{x}_{\boldsymbol{i}}^{\boldsymbol{G}}$ and acceleration $\ddot{\boldsymbol{x}}_{\boldsymbol{i}}^{\boldsymbol{G}}$ vectors of any body i are computed (velocities are not necessary to compute the ZMP). $\boldsymbol{d_i}$ is the relative position vector between COM_{i-1} and COM_i while R_i (recursively computed) is the absolute rotation matrix between the relative frame of body i and the inertial frame.

$$\begin{cases} \boldsymbol{x}_{\boldsymbol{i}}^{\boldsymbol{G}} = \boldsymbol{x}_{\boldsymbol{i-1}}^{\boldsymbol{G}} + R_i \boldsymbol{d_i} \\ \ddot{\boldsymbol{x}}_{\boldsymbol{i}}^{\boldsymbol{G}} = \ddot{\boldsymbol{x}}_{\boldsymbol{i-1}}^{\boldsymbol{G}} + R_i(\dot{\boldsymbol{\omega}}_{\boldsymbol{i}} \times \boldsymbol{d_i} + \boldsymbol{\omega_i} \times (\boldsymbol{\omega_i} \times \boldsymbol{d_i})) \end{cases} \tag{B.3}$$

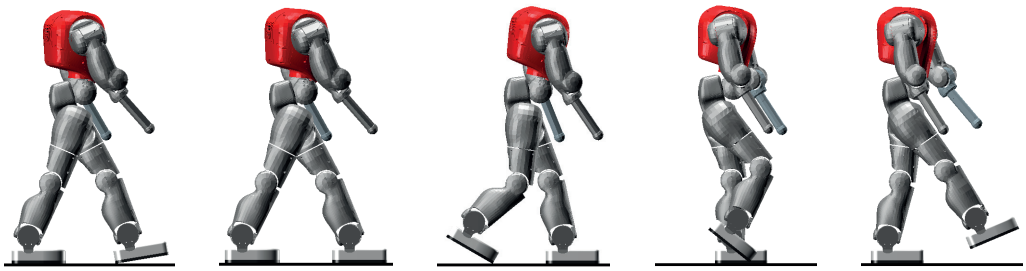
Once this recursive forward kinematics step is done, the rate of change of angular and linear momentum ($\dot{\boldsymbol{H}}^{\boldsymbol{0}}$ and $\dot{\boldsymbol{N}}^{\boldsymbol{0}} = \sum_{i=1}^{N} m_i \cdot \ddot{\boldsymbol{x}}_{\boldsymbol{i}}$) are computed, along with the whole-body COM position ($\boldsymbol{r_{COM}}$). m_i is the mass of body i while I_i^G (recursively computed) is its absolute inertial matrix.

$$\dot{\boldsymbol{H}}^{\boldsymbol{0}} = \sum_{i=1}^{N} (\boldsymbol{x}_{\boldsymbol{i}}^{\boldsymbol{G}} \times (m_i \, \ddot{\boldsymbol{x}}_{\boldsymbol{i}}^{\boldsymbol{G}}) + I_i^G \, \dot{\boldsymbol{\omega}}_{\boldsymbol{i}} + \boldsymbol{\omega}_{\boldsymbol{i}} \times (I_i^G \, \boldsymbol{\omega}_{\boldsymbol{i}})) \tag{B.4}$$

The last step consists in computing the ZMP with Equation (B.1). During this recursive process, each symbolic equation computed with Matlab is printed in a C file. At the end, thousands of lines written in C are generated, which would have been impossible to derive by hand. In sum, this produces a computationally-efficient C file allowing to compute the ZMP in real-time.

B.5 Results

The ZMP computation was tested in simulation on two different kinds of gait. The first one is the *2D walking gait*. A few snapshots of this gait are shown in Figure B.7. It can be observed that the robot exhibits straight knees at some phases of the gait (in contrast to many existing ZMP-based walkers) and strikes the ground with the heel first, similarly to humans.



(a) t=0.42 s (b) t=0.45 s (c) t=0.61 s (d) t=0.79 s (e) t=0.99 s

Figure B.7: Snapshots of the 2D walking gait of the COMAN in the Robotran simulator, corresponding to panels (a) and (b) of Figure B.9.

The other gait is the *3D gait*. Some corresponding snapshots are presented in Figure B.8. This second gait is clearly less robust, more jerky and less human-like than the 2D one.

B.5.1 Computation time

One of the requirements of the ZMP computation was to be fast enough to be computable by the real COMAN controller without compromising its realtimeness. The computation time was evaluated around 0.014 ms, which is fast enough given the controller sampling rate,

(a) t=0.38 s (b) t=0.47 s (c) t=0.52 s (d) t=0.66 s (e) t=0.83 s

Figure B.8: Snapshots of the 3D walking gait of the COMAN in the Robotran simulator, corresponding to panels (c) and (d) of Figure B.9 and to Figure B.10.

namely 1 ms. This was tested on a Dell OptiPlex 7010 computer with quad-core Intel(R) Core(TM) i7-3770 CPU, 3.4GHz and 8 Go RAM.

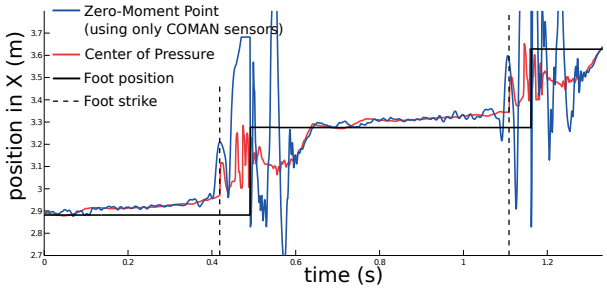
B.5.2 2D gait

Because the robot waist is constrained to stay within the world sagittal plane during the 2D walking gate, it is nonsense to compute the Y component of the ZMP in this case. Panel (a) in Figure B.9 shows the X position (i.e. along the world sagittal plane) of the ZMP relative to an inertial frame. To compute this position, only sensors available on the real COMAN were used. To analyze the accuracy of this ZMP computation, the computed center of pressure (COP) absolute position is shown in the same figure. This COP position is considered to be errorless as it is computed from a weighted sum of the ground reaction forces. As the ZMP and COP are equivalent in balanced gaits (see Section B.3.2), the matching between these two signals shows the accuracy of the ZMP computation proposed here.

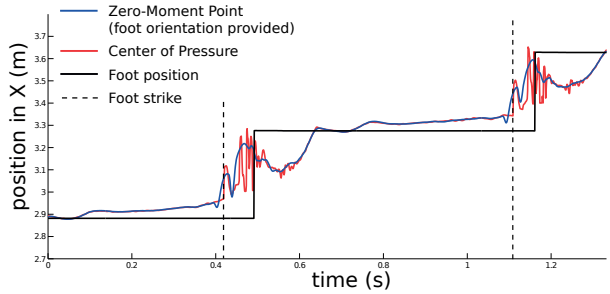
Figure B.9(a) shows that the matching between both signals is quite good, except just after a heel strike. This is due to the assumption that the supporting foot is perfectly flat on the ground (see Section B.4.3), which is not always the case with the gait obtained from our bio-inspired controller, mainly after heel strike. Coherently, providing the COMAN with the supporting foot absolute orientation (along with its acceleration), an information unavailable on the real COMAN, gives the results shown in Figure B.9(b). The blue curve in Figure B.9(b) corresponds to the blue one in Figure B.9(a), but with the foot orientation provided. The matching in Figure B.9(b) appears to be much better as the ZMP and the COP coincide, except that the ZMP is smoother (indeed, filters are used on the ZMP inputs).

B.5.3 3D gait

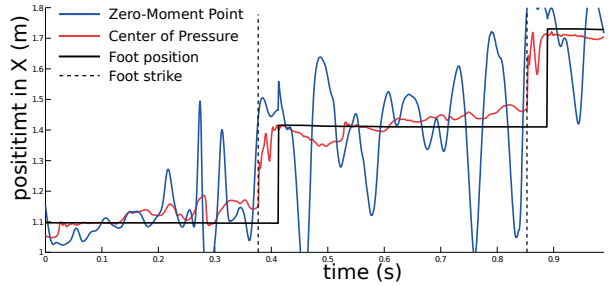
The generated 3D gait was much more jerky, due to the lack of stability of our bio-inspired controller. Consequently, larger accelerations were induced and the ZMP computation was much more prone to errors. Therefore, only results where the absolute foot orientation was



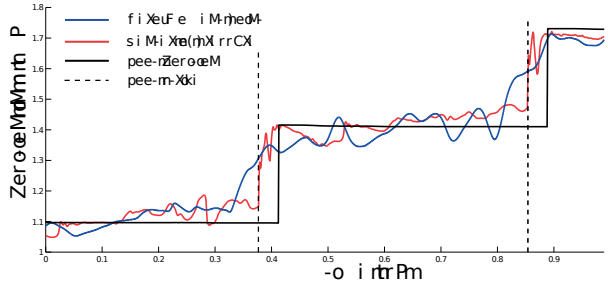
(a) ZMP computed for the 2D walking gait, using only sensors available on the real COMAN, without post-process filtering.



(b) ZMP computed for the 2D walking gait, with the foot orientation provided, without post-process filtering.



(c) ZMP computed for the 3D walking gait, with the foot orientation provided, without post-process filtering.



(d) ZMP computed for the 3D walking gait, with the foot orientation provided, post-processed with a 100 ms-wide running average.

Figure B.9: The ZMP position in X relative to an inertial frame is presented, along with the COP position, the supporting foot position and the foot strike instants. The ZMP position is shifted to compensate the 25 ms delay introduced by the filters.

used in the computation (i.e. relaxing the "flat foot" hypothesis) are presented, a situation which does not make sense on the real robot. The X position of the ZMP is shown in Figure B.9(c). As expected, the ZMP still matches the COP position, although with much more noise. Processing the ZMP with a 100 ms-wide running average post-process filtering, gives the signal shown in Figure B.9(d). This shows that the filtered version of the ZMP matches the COP. Their positions in the transverse plane are shown in Figure B.10.

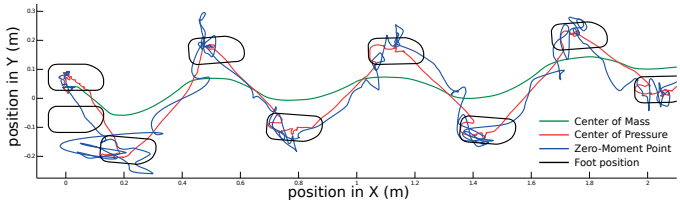


Figure B.10: Position of the COM, the COP, the ZMP and the feet in the transverse world plane for the 3D walking gait. To improve the graph clarity, the COP position was low-pass filtered with a 100 ms-wide running average. The ZMP is filtered like in Figure B.9(d).

B.6 Discussion

For gaits being smooth enough like the 2D one, the method presented here exhibits very good results using only the sensors available on the real robot, except just after heel strikes, since the "flat foot" hypothesis is strongly violated at that moment. Nevertheless, this is not a real problem as the final purpose of this ZMP computation on this bio-inspired walker is to provide a measure of the gait stability in order to monitor possible falls (and trigger corresponding reactions). Because heel strike initiates the beginning of the double support phase, i.e. the one with the largest support polygon, the ZMP-based stability monitoring can be disregarded during this phase.

For jerky gaits, like the 3D one, results are deteriorated. So, Figure B.10 shows that even after post-process filtering the ZMP position sometimes leaves the support polygon surface, although the COP does not. This happens especially for the first steps, i.e. the jerkiest ones. Consequently, the robot would detect a fall, although this is not the case. Different solutions exist to overcome this problem. The first one would be to investigate more efficient filters for the ZMP inputs. Another one would be to use the COP position (computed thanks to the 6-DOF force and torque sensors included in the robot ankles) instead of the ZMP. Nevertheless, the problems related to the sharp acceleration variations would also alter the COP position. Then, a last method would be to use a simplified version of the robot model to compute the ZMP, but with no guarantee that this would improve the results. Finally, further research might be carried out to test this ZMP computation on the real robot.

B.7 Conclusion

In this contribution, we presented a computationally-efficient method to compute the zero-moment point on a robot walking with a human-like gait (i.e. with straight knees and heel strikes) obtained from a bio-inspired controller. Because our final objective was to evaluate the possibility to implement this method on a real robot, two important requirements were (i) to minimize the computation time and (ii) to only use inputs corresponding to real robot sensors. In the present contribution, we presented simulation studies.

As for the first requirement, the symbolic approach led to a very short computational time given realistic controller time constraints. Following this approach, we were able to automatically generate a custom C-code file, which would have been impossible to produce manually.

As for the second requirement, we established that the exact ZMP computation requires knowing the absolute orientation of the robot bodies, a measurement which is difficult to obtain with actual sensors. However, we proposed a second method, assuming that one foot is flat on the ground, which provided good matching with the real ZMP, except just after heel strikes. This is however not the most critical gait phase, since it corresponds to the largest support polygon.

However, this contribution showed that new difficulties appeared when computing the ZMP on a gait closer to the human one (i.e. with straight legs and feet strikes), in contrast to the "safe" (feet flat on the ground...) walking approaches common to many traditional walkers. In particular, the heel strikes strongly deteriorated the ZMP accuracy, mainly due to the high accelerations involved. Using the ZMP as a stability criterion for human-like walking seems therefore not ideal. This is however not a problem because the ZMP is not respected either during real human locomotion. Because of these two reasons (ZMP inaccuracy and its non application in real human walking), the ZMP criterion was not further studied in this thesis.

C Particle swarm optimization

Most walking and running gaits obtained in this thesis recruited a heuristic optimization called *particle swarm optimization* (PSO), in order to tune the controllers unknown parameters. PSO is thoroughly described in (Kennedy and Eberhart, 1995) and (Clerc and Kennedy, 2002). This appendix summarizes its core principles.

PSO is a cooperative, population-based optimization method, inspired from bird flocking and relying on the concept of "swarm intelligence". The parameters to optimize are represented by a set of particles (the population). Each particle lives in a n-dimensional search space, where n is the number of parameters to optimize, in order to maximize a given fitness function. Basically, the fitness function can be seen as a map guiding the particles in their travel through the search space. The particles with higher fitness values attract the other particles to them while each particle also tends to go towards the position corresponding to its best fitness.

More precisely, each particle state is composed by its position and its velocity, both represented by n-dimensional vectors. At each iteration, the velocity of a particle is updated according to three criteria: (i) the particle current velocity; (ii) the particle best position achieved so far (i.e. its position scoring the highest fitness function); and (iii) the best position of all particles so far. Then, all particle positions are updated according to their respective velocities.

We detail the different steps of the algorithm.

Initially, the particles are randomly spread over the whole search space, and their related fitness functions are computed. At each iteration i, the velocity of each particle is updated for the next iteration i' according to Equation (C.1), where i and i' are iteration indexes, j is the particle index, x_{ij} is the particle current position and v_{ij} its current velocity (the other parameters are detailed later). Next, the particle positions in the search space are updated as detailed in Equation (C.2). After this process, the fitness functions are computed again for each particle, then each particle position is updated, and so on for a given number of iterations (i.e. number of generations).

$$v_{i'j} = \omega\, v_{ij} + R_1\, \varphi_1 \left(P_{ij} - x_{ij}\right) + R_2\, \varphi_2 \left(P_{i,b} - x_{ij}\right) \tag{C.1}$$

$$x_{i'j} = x_{ij} + v_{ij} \tag{C.2}$$

In Equations (C.1) and (C.2), the following parameters control the behavior of the optimization:

- ω is the inertia, related to the weight of the previous velocity.
- P_{ij} is the position with the best fitness in the particle history.
- $P_{i,b}$ is the position with the best fitness in the population history.
- φ_1 is the cognitive factor, giving the relative importance to the particle history.
- φ_2 is the social factor, giving the relative importance to the population history.
- R_1 and R_2 are two pseudo-random numbers in the range $[0, 1]$.

These parameters influence the trade-off between exploration and exploitation, thus affecting the capability of the optimizer to find appropriate solutions. These rules can be incremented, for instance to define the behavior of particles reaching the boundaries of the search space, or to limit the particle velocities.

D Low-level impedance controller

The simulation environment models the physical actuators of COMAN. As mentioned in Section 2.1, most joints are equipped with series elastic actuators (SEAs). Their full implementation is reported in (Dallali et al., 2013) and in (Zobova et al., 2017).

In this appendix, we overview the control of these motors when appropriate position or torque references are provided. In other words, we detail the low-level impedance controller (one for each joint) in charge of computing the appropriate voltage V (commanding the motor) when receiving a position reference (q_{ref}) and a torque reference (τ_{ref}). This low-level controller is adapted from (Mosadeghzad et al., 2012) and is depicted in Figure D.1.

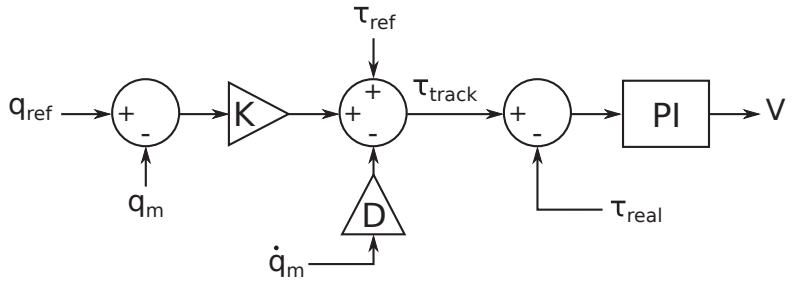


Figure D.1: When receiving a position reference q_{ref} and a torque reference τ_{ref}, the low-level impedance controller computes the motor voltage V. This requires the following sensory information: the joint motor position q_m and its time derivative $\dot{q}_m$, as well as the real torque measurement τ_{real}. K and D are PD gains for the position control. The PI module is in charge of computing the voltage V to track the torque τ_{track}.

The first stage of the low-level impedance controller consists in a PD (proportional-derivative) module constraining the motor position q_m ($\dot{q}_m$ being its time derivative) to reach the reference position q_{ref}. This is obtained with the following rule, where K and D are proportional and derivative gains:

$$\tau_{track} = K\,(q_{ref} - q_m) - D\,\dot{q}_m + \tau_{ref} \tag{D.1}$$

An additional torque reference τ_{ref} is added to the result of this PD module, in order to obtain the torque to track (τ_{track}), see Equation (D.1). This torque is later compared to the real torque measurement τ_{real}. Importantly, a uniform noise with a maximal amplitude of $0.4\,Nm$ is added to τ_{real} (in simulation only). Finally, a PI (proportional-integral) controller is used to compute the appropriate voltage V, using the following rule (with P being the proportional gain and I the integral gain):

$$V = P\,(\tau_{track} - \tau_{real}) + I \int (\tau_{track} - \tau_{real}) \tag{D.2}$$

This generalist controller was developed to obtain either position or torque tracking, although both could never be perfectly achieved at the same time (Spong et al., 2005). Indeed, a position controller can approximatively be obtained by using large gains K and D, with low gains P and I, while the torque reference τ_{ref} is set to 0. In contrast, setting the gains K and D to 0 results in a classical torque PI controller. This last configuration was the main one used in the algorithms developed in this thesis, especially to reproduce the torques generated by the virtual muscles. Yet, the position control was also recruited in some contributions, mainly for the upper-body joints.

E Forward speed modulation during 2D walking gaits

Publication

The material presented in this appendix is related to Chapter 6 and is adapted from:

Van der Noot N, Ijspeert AJ and Ronsse R (2015) Biped gait controller for large speed variations, combining reflexes and a central pattern generator in a neuromuscular model. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, 26-30 May 2015, pp. 6267-6274. DOI: 10.1109/ICRA.2015.7140079.

E.1 CPG full equations

The equations governing the firing rate x_i of each neuron N_i are presented in Equation (E.1), self-inhibition equations in Equation (E.2).

$$\begin{aligned}
\dot{x_1} &= \frac{1}{\tau}\left(-x_1 - \beta_A v_1 - \eta_A [x_4]^+ - \eta_D [x_2]^+ - \eta_E [x_5]^+ + u_1\right)\\
\dot{x_2} &= \frac{1}{\tau}\left(-x_2 - \beta_B v_2 - \eta_B [x_5]^+ - \eta_D [x_1]^+ - \eta_E [x_4]^+ + u_2\right)\\
\dot{x_3} &= \frac{1}{\tau}\left(-x_3 - \beta_C v_3 - \eta_C [x_6]^+ - \eta_F [x_2]^+ - \eta_G [x_5]^+ + u_3\right)\\
\dot{x_4} &= \frac{1}{\tau}\left(-x_4 - \beta_A v_4 - \eta_A [x_1]^+ - \eta_D [x_5]^+ - \eta_E [x_2]^+ + u_4\right)\\
\dot{x_5} &= \frac{1}{\tau}\left(-x_5 - \beta_B v_5 - \eta_B [x_2]^+ - \eta_D [x_4]^+ - \eta_E [x_1]^+ + u_5\right)\\
\dot{x_6} &= \frac{1}{\tau}\left(-x_6 - \beta_C v_6 - \eta_C [x_3]^+ - \eta_F [x_5]^+ - \eta_G [x_2]^+ + u_6\right)
\end{aligned} \tag{E.1}$$

$$\begin{aligned}
\dot{v_1} &= \frac{1}{\gamma_A \tau}\left(-v_1 + [x_1]^+\right) \qquad & \dot{v_4} &= \frac{1}{\gamma_A \tau}\left(-v_4 + [x_4]^+\right)\\
\dot{v_2} &= \frac{1}{\gamma_B \tau}\left(-v_2 + [x_2]^+\right) & \dot{v_5} &= \frac{1}{\gamma_B \tau}\left(-v_5 + [x_5]^+\right)\\
\dot{v_3} &= \frac{1}{\gamma_C \tau}\left(-v_3 + [x_3]^+\right) & \dot{v_6} &= \frac{1}{\gamma_C \tau}\left(-v_6 + [x_6]^+\right)
\end{aligned} \tag{E.2}$$

E.2 Optimization parameters

Table E.1 gathers all the optimization parameters along with their bounds. Some of these parameters are used to get the trunk angle reference θ_{ref}, the oscillator time constants τ, and the stimulations gains k_{HFL}, $k_{HAM,1}$ and $k_{HAM,2}$, according to the rules described in Section E.3.

Table E.1: Optimization parameters and their bounds

	min	max		min	max		min	max
speed			**init**			β		
P_θ	0.01	0.3	κ	0	0.13	β_A	4.5	6.5
P_τ	0.08	0.2	**reflex**			β_B	4	6
P_{HFL}	2.2	4	$S_{0,VAS}$	0.01	0.03	β_C	3	6
$P_{HAM,1}$	1.3	3.2	G_{SOL}	0.7	1.6	η		
$P_{HAM,2}$	0.5	2	G_{VAS}	0.6	20	η_A	3	6
p_θ	0	1	ϕ_{off}	2.5	π	η_B	4	7
p_τ	-0.2	0	ξ_1	0.3	10	η_C	3	6
p_{HFL}	0	4.5	ξ_2	0.004	0.15	η_D	2.5	4
$p_{HAM,1}$	0	4.5	γ			η_E	2.5	5
$p_{HAM,2}$	-4	0	γ_A	0.5	2.5	η_F	2.5	5
const			γ_B	0.5	2.5	η_G	3	5.5
k_{GLU}	0.8	2	γ_C	0.5	3			

E.3 Speed dependent parameters

The trunk angle reference θ_{ref}, the oscillator time constant τ, and the stimulations gains k_{HFL}, $k_{HAM,1}$ and $k_{HAM,2}$ are computed as simple linear functions of the target speed v_t, according to Equation (E.3). v^* is an arbitrary reference speed set to 0.6 m/s. Speed modulation is then simply obtained by modifying the target speed v_t. Finally, k_{GLU} is kept constant for all speeds (see Table E.1).

$$\begin{aligned}
\theta_{ref} &= P_\theta + p_\theta\,(v_t - v^*)\\
\tau &= P_\tau + p_\tau\,(v_t - v^*)\\
k_{HFL} &= P_{HFL} + p_{HFL}\,(v_t - v^*)\\
k_{HAM,1} &= P_{HAM,1} + p_{HAM,1}\,(v_t - v^*)\\
k_{HAM,2} &= P_{HAM,2} + p_{HAM,2}\,(v_t - v^*)
\end{aligned} \tag{E.3}$$

F Forward speed modulation during 2D running gaits

The material presented in this appendix is related to Chapter 7, about the developments of a controller to achieve running locomotion on a humanoid robot.

F.1 CPG equations

The full CPG network is presented in Figure 7.1b. Each firing rate x_i (corresponding to a neuron N_i) is integrated based on Equation (7.1). The full development is presented in (F.1). Similarly, the self-inhibitions equations (i.e. related to v_i) are governed by (7.2) and fully developed in (F.2).

$$
\begin{aligned}
\dot{x_1} &= \frac{1}{\tau}\left(-x_1 - \beta_A\, v_1 - \eta_A[x_4]^+ - \eta_D[x_2]^+ - \eta_E[x_5]^+ + u_1\right)\\
\dot{x_2} &= \frac{1}{\tau}\left(-x_2 - \beta_B\, v_2 - \eta_B[x_5]^+ - \eta_D[x_1]^+ - \eta_E[x_4]^+ + u_2\right)\\
\dot{x_3} &= \frac{1}{\tau}\left(-x_3 - \beta_C\, v_3 - \eta_C[x_6]^+ - \eta_F[x_2]^+ - \eta_G[x_5]^+ + u_3\right)\\
\dot{x_4} &= \frac{1}{\tau}\left(-x_4 - \beta_A\, v_4 - \eta_A[x_1]^+ - \eta_D[x_5]^+ - \eta_E[x_2]^+ + u_4\right)\\
\dot{x_5} &= \frac{1}{\tau}\left(-x_5 - \beta_B\, v_5 - \eta_B[x_2]^+ - \eta_D[x_4]^+ - \eta_E[x_1]^+ + u_5\right)\\
\dot{x_6} &= \frac{1}{\tau}\left(-x_6 - \beta_C\, v_6 - \eta_C[x_3]^+ - \eta_F[x_5]^+ - \eta_G[x_2]^+ + u_6\right)
\end{aligned}
\tag{F.1}
$$

$$
\begin{aligned}
\dot{v_1} &= \frac{1}{\gamma_A\,\tau}\left(-v_1 + [x_1]^+\right) \qquad & \dot{v_4} &= \frac{1}{\gamma_A\,\tau}\left(-v_4 + [x_4]^+\right)\\
\dot{v_2} &= \frac{1}{\gamma_B\,\tau}\left(-v_2 + [x_2]^+\right) & \dot{v_5} &= \frac{1}{\gamma_B\,\tau}\left(-v_5 + [x_5]^+\right)\\
\dot{v_3} &= \frac{1}{\gamma_C\,\tau}\left(-v_3 + [x_3]^+\right) & \dot{v_6} &= \frac{1}{\gamma_C\,\tau}\left(-v_6 + [x_6]^+\right)
\end{aligned}
\tag{F.2}
$$

F.2 Hill muscles parameters

Each muscle group is modeled by a set of equations called the Hill muscle model (Hill, 1938), whose implementation is described in (Geyer et al., 2003) and (Geyer and Herr, 2010). The different characteristics of our virtual muscles (see Figure 7.1a) are listed in Table F.1.

Table F.1: The fixed MTU parameters of the 12 types of muscles for COMAN are reported in this table. When a leg MTU acts on different joints, its values are explicitly reported as (a) for ankle, (k) for knee and (h) for hip (except if they are equal). These values were extracted from (Geyer and Herr, 2010) and (Song and Geyer, 2015a) for the leg muscles. The arm muscles were estimated with the *OpenSim* simulator (Delp et al., 2007) using the human model developed in (Rajagopal et al., 2016). The masses m_{mtu} were obtained using the method proposed in (Wang et al., 2012), while the λ values were obtained from (Yamaguchi et al., 1990). These values were scaled to the size of COMAN by using dynamic scaling methods, being described in (Bejan and Marden, 2006) and (Schepelmann et al., 2012).

	F_{max} [N]	v_{max} [l_{opt}/s]	l_{opt} [mm]	l_{slack} [mm]	r_o [mm]	φ_{max} [deg]	φ_{ref} [deg]	ρ [-]	m_{mtu} [g]	λ [%]
SOL	1415	9	17	110	21	20	-10	0.5	240	81
TA	285	18	26	100	17	-10	20	0.7	70	70
GAS	530	18	21	170	21	20 (a) 40 (k)	-10 (a) 15 (k)	0.7	110	54
VAS	2125	18	34	98	26	15	55	0.7	720	50
RF	425	18	34	149	26 (k) 34 (h)	135 (k) - (h)	55 (k) 10 (h)	0.5 (k) 0.3 (h)	140	45
HAM	1060	18	43	132	21 (k) 34 (h)	0 (k) - (h)	0 (k) -25 (h)	0.7	450	44
GLU	530	18	47	56	43	-	-30	0.5	250	50
HFL	710	18	47	43	43	-	0	0.5	330	50
SET	180	18	59	38	18	-70	-120	0.6	110	42
SFL	525	18	43	42	14	-30	-15	0.7	230	57
EET	460	18	53	51	10	-25	-60	0.8	240	32
EFL	390	18	50	72	16	-70	-60	1	190	46

F.3 CPG excitations

The CPG excitations u_i are usually set to a tonic constant $u = 1$. However, they can be modulated, according to the feet strikes, so that N_2 starts firing just after left strike and N_5 just after right strike. Interestingly, these excitations modulations are very short (i.e. $u_i = 1$ most of the time).

In case the oscillators are too fast, all excitations are cut to 0. More precisely, this correction is applied during the flying phase, when x_2 becomes positive while the last leg to touch the ground was the right one, or when x_5 becomes positive while the last leg to touch the ground

was the left one. This correction is applied until the next strike happens.

In contrast, the neurons are too slow when N_2 is still negative after left strike or when N_5 is still negative after right strike. In that case, the corresponding neuron receives an extra contribution, while the excitation of the neuron firing before is reduced. On top of that, an extra safety is added so that neurons only fire during their corresponding leg reference phase. The left leg reference phase starts at the left foot strike and ends at the right foot strike (similarly for the right leg reference phase, lasting from right foot strike to left foot strike). This behavior is achieved with the equations presented in (F.3).

$$\begin{aligned} u_1 &= u - [x_1]^+_{Ref,L} - [x_1]^+_{Str,L} & u_4 &= u - [x_4]^+_{Ref,R} - [x_4]^+_{Str,R} \\ u_2 &= u - [x_2]^+_{Ref,R} + [x_2]^-_{Str,L} & u_5 &= u - [x_5]^+_{Ref,L} + [x_5]^-_{Str,R} \\ u_3 &= u - [x_3]^+_{Ref,L} & u_6 &= u - [x_6]^+_{Ref,R} \end{aligned} \tag{F.3}$$

The $[\bullet]_{Str,L}$ function always saturates its argument to zero, except after the left foot strike if the firing rate x_2 is still negative. In this case, it keeps its argument intact as long as x_2 is negative (similar for $[\bullet]_{Str,R}$ with the right leg and x_5). The function $[\bullet]_{Ref,L}$ keeps its argument intact during the left leg reference phase, and saturates to 0 otherwise (similarly for the right leg reference phase with $[\bullet]_{Ref,R}$). The functions are combined with the functions $[\bullet]^+ = max(0,\bullet)$ and $[\bullet]^- = -min(\bullet,0)$.

To help CPG initiation, all excitations u_i are set to 0 during the first 10 ms, except u_5 which is set to 1 if the right leg is the first one to impact the ground. Symmetrically, u_2 is the only excitation set to 1 if the left leg impacts the ground first.

F.4 Fitness function stages

The global fitness function is computing by aggregating the separate scores of the different stages run in parallel. All these scores are computed at the end of the simulation run (i.e. 70 s) or earlier if the robot fell.

Most stages fitness functions f are computed using the Gaussian function presented in Equation (F.4), where x is the studied parameters, x^* its target value and α, β are two weight parameters. In particular, β represents the maximum value of f, and is used to balance the corresponding stage weight.

$$f = \beta\, e^{-\alpha (x-x^*)^2} \tag{F.4}$$

A first stage constraints the resulting running speed to match a target speed v_{ref}. Using (F.4), x is set to the actual speed, x^* to v_{ref} and $(\alpha;\beta)$ to (85;700). This stage receives the largest β, therefore contributing most to the resulting fitness.

Two other stages study the gait robustness by rewarding the running time and distance traveled before falling. More precisely, a linear function is computed proportionally to the time without falling, such that a score of 0 corresponds to a theoretical fall when starting the simulation (i.e. $t = 0\,s$) and a reward of 300 is granted at the end of the simulation time (i.e. $t = 70\,s$). Similarly, a linear function increases from 0 to 300 and corresponds to the traveled distance before falling, from 0 to $50\,m$. Over this maximal value of $50\,m$, no extra reward is granted (i.e. maximal reward of 300 given). This stage prevents the robot from reaching a standstill position.

Another stage encourages the synchronization of the neuron firing rates with the strikes. In fact, x_2 is supposed to fire after left strike, and x_5 after right strike. The average error between the moment when these neurons start to fire and the actual strikes is computed as the CPG prediction error. The stage function is computed using (F.4), with x set to this prediction error, x^* to 0 and $(\alpha;\beta)$ to (1100;500).

Running is favored over walking by constraining the flying phase ratio to last at least 10 % of the cycle duration. More precisely, Equation (F.4) is used with x set to the flying phase ratio (natural unit, no %), x^* to 0.1 and $(\alpha;\beta)$ to (450;250). If x exceeds 0.1, the maximal reward (i.e. 250) is granted.

Similarly to (Wang et al., 2012), the optimizer also aims at maintaining an upright posture. This is done by computing the mean torso orientation angle in the sagittal plane (with $0\,rad$ corresponding to straight posture). This is done using Equation (F.4) with x set to this mean torso orientation, x^* to 0 and $(\alpha;\beta)$ to (20;350). A maximal reward (i.e. 350) is granted if the torso orientation is lower than $0.1\,rad$.

Finally, a last stage minimizes the equivalent metabolic energy consumption in virtual muscle contraction per unit distance walked. This energy is computed as detailed in (Bhargava et al., 2004). The fitness stage is computed again with Equation (F.4) where x is the metabolic energy consumption of both legs per unit distance walked and normalized by the walker mass, x^* is set to 0 and $(\alpha;\beta)$ to (0.01;200).

F.5 Optimization parameters

Table F.2: The parameters to be optimized in the controller, and their ranges are reported in this table. The speed dependent parameters are computed as follows: $\tau = K_\tau + L_\tau\, v_* + M_\tau\, v_*^2$; $k_{HFL,1} = K_{HFL,1} + L_{HFL,1}\, v_*$; $k_{HFL,2} = K_{HFL,2} + L_{HFL,2}\, v_* + M_{HFL,2}\, v_*^2$; $k_{HAM} = K_{HAM} + L_{HAM}\, v_* + M_{HAM}\, v_*^2$; $G_{VAS} = K_{G,VAS} + L_{G,VAS}\, v_*$; $G_{GAS} = K_{G,GAS} + L_{G,GAS}\, v_*$; $G_{SOL} = K_{G,SOL} + L_{G,SOL}\, v_* + M_{G,SOL}\, v_*^2$; $G_{S-T} = K_{G,S-T} + L_{G,S-T}\, v_*$, where $v_* = v_{ref} - 1.5$ and v_{ref} is the target forward speed.

	min	max		min	max		min	max		min	max
CPG			$S_{0,HAM,st}$	0.01	0.35	$k_{d,sp,GLU}$	0	0.1	$K_{G,VAS}$	0.75	1.75
β_a	4.5	6.5	$S_{0,GLU,st}$	0.01	0.35	$k_{p,sp,HFL}$	0	4	$K_{G,GAS}$	5	13
β_b	4	6	$S_{0,HFL,st}$	0.01	0.35	$k_{d,sp,HFL}$	0	0.1	$K_{G,SOL}$	2	3.5
β_c	3	6	$S_{0,SOL,sw}$	0.01	0.05	G_{HAM}	0	3	$K_{G,S-T}$	2	6
γ_a	1	2	$S_{0,TA,sw}$	0.01	0.05	G_{GLU}	0	2	L_τ	-0.1	0.1
γ_b	2	3	$S_{0,GAS,sw}$	0.01	0.2	$G_{TA,sw}$	0	5	$L_{HFL,1}$	-4	-1
γ_c	2	3	$S_{0,VAS,sw}$	0.01	0.1	$G_{TA,st}$	0	5	$L_{HFL,2}$	-10	20
η_a	5	6	$S_{0,RF,sw}$	0.01	0.5	$l_{off,TA,sw}$	0.4	0.8	L_{HAM}	-3	3
η_b	4	5.5	$S_{0,HAM,sw}$	0.01	0.1	$l_{off,TA,st}$	0.4	0.8	$L_{G,VAS}$	1	3
η_c	4.5	6.5	$S_{0,GLU,sw}$	0.01	0.05	d_{sp}	-0.3	0.2	$L_{G,GAS}$	-20	-5
η_d	3	4.5	$S_{0,HFL,sw}$	0.01	0.05	d_{si}	0.1	0.7	$L_{G,SOL}$	-1.5	1.5
η_e	3	4.5	$k_{p,HAM}$	0	6	si_{VAS}	0.2	1	$L_{G,S-T}$	5	15
η_f	3	5.5	$k_{d,HAM}$	0	0.5	si_{RF}	0	1	M_τ	-0.1	0
η_g	3	5.5	$k_{p,GLU}$	0	18	$\varphi_{k,th,sw}$	0	0.3	$M_{HFL,2}$	20	120
reflex			$k_{d,GLU}$	0	0.8	$\varphi_{k,th,st}$	0.02	0.2	M_{HAM}	10	50
$S_{0,SOL,st}$	0.01	0.05	$k_{p,HFL}$	0	10	**speed**			$M_{G,SOL}$	0	20
$S_{0,TA,st}$	0.01	0.05	$k_{d,HFL}$	0	0.6	K_τ	0.042	0.052	**const**		
$S_{0,GAS,st}$	0.01	0.05	$k_{p,sp,VAS}$	0	5	$K_{HFL,1}$	4	7	θ_{ref}	0.03	0.09
$S_{0,VAS,st}$	0.045	0.75	$k_{d,sp,VAS}$	0	0.1	$K_{HFL,2}$	2	8	$\varphi_{h,ref}$	-0.25	-0.05
$S_{0,RF,st}$	0.01	0.5	$k_{p,sp,GLU}$	0	5	K_{HAM}	2	5	$k_{\varphi,k}$	4	13

Table F.3: The parameters to be optimized for the biped initial dynamics state, and their ranges are reported in this table. z_w is the vertical position of the waist, θ_w its orientation (used to describe the floating base kinematics). φ_h, φ_k, φ_a and φ_{sh} are respectively the hip, knee, ankle and shoulder sagittal joint angles (with R and L standing respectively for right and left). The $\dot{\bullet}$ function denotes the time derivative of the quantity (i.e. its speed).

	min	max		min	max		min	max		min	max
floating			**position**			$\varphi_{a,R}$	-0.09	0.09	$\dot{\varphi}_{k,R}$	-5	5
z_w	0.46	0.58	$\varphi_{h,R}$	-0.87	-0.35	$\varphi_{a,L}$	0.09	0.45	$\dot{\varphi}_{k,L}$	-5	5
θ_w	0	0.3	$\varphi_{h,L}$	-0.09	0.35	**speed**			$\dot{\varphi}_{a,R}$	-5	5
$\dot{z}_w$	-0.5	0.5	$\varphi_{k,R}$	0	0.87	$\dot{\varphi}_{h,R}$	-5	5	$\dot{\varphi}_{a,L}$	-5	5
$\dot{\theta}_w$	-5	5	$\varphi_{k,L}$	0.35	1.75	$\dot{\varphi}_{h,L}$	-5	5	$\dot{\varphi}_{sh}$	-5	5

G Forward speed modulation during 3D straight walking gaits

Publication

The material presented in this appendix is related to Chapter 9 and is adapted from the the following submitted paper:

Van der Noot N, Ijspeert AJ and Ronsse R (conditionally accepted) Bio-inspired controller achieving forward speed modulation with a 3D bipedal walker. In: International Journal of Robotics Research (IJRR).

G.1 Muscle tendon unit

The full muscle model and its biological relevance is covered in (Geyer et al., 2003) and (Geyer and Herr, 2010). We report here the steps and equations to implement it.

G.1.1 MTU kinematics

First, some parameters can be computed from the joint angular positions φ, through the lever arm r_m and the MTU length l_{mtu}.

Each lever arm r_m obeys an equation like $r_m = \pm r_0\, cos(\varphi - \varphi_{max})$. It is thus maximal (and equal to $\pm r_0$) when φ is equal to φ_{max}, except for lever arms acting the hip joint which is kept constant (i.e. $r_m = \pm r_0$). The sign depends on the resulting torque contribution in the frames of Figure 9.2.

The MTU length is computed as $l_{mtu} = l_{opt} + l_{slack} + \sum_i \Delta l_{mtu,i}$, where l_{opt} is the CE optimum length, l_{slack} is the distance corresponding to SE being slack and i is the joint affected by the muscle (two joints for GAS and HAM, one otherwise). For the hip: $\Delta l_{mtu} = \pm \rho\, r_0(\varphi - \varphi_{ref})$, where ρ accounts for muscle pennation angles and φ_{ref} is the angle at which $l_{mtu} = l_{opt} + l_{slack}$. The sign can be deduced from the muscle length evolution with φ (e.g. positive when l_{mtu} increases with φ). For the other joints: $\Delta l_{mtu} = \pm \rho\, r_0\, (sin(\varphi - \varphi_{max}) - sin(\varphi_{ref} - \varphi_{max}))$.

All parameters used in these equations are reported in Table G.1. All the angle frames are consistent with the ones displayed in Figure 9.2 and equal zero in the homing position, i.e. the walker standing straight with the arms hanging vertically.

G.1.2 MTU forces

The following equations depend on the muscle state, which can be represented by a single variable: the CE length l_{ce} (the other variables can be computed from the kinematics and from l_{ce}). In the approach of (Geyer and Herr, 2010), l_{ce} is found by integrating its time derivative v_{ce}. However, this integration requires a small time step, due to the stiff and strongly non-linear derivative state equations, which is a strong issue for a robot controller sampled with a fixed time step (Van der Noot et al., 2014). A possible solution is to integrate these dynamic equations with a smaller time step than the one of the controller itself. For example, the controller of COMAN being sampled with a frequency of $1\,kHz$, the following equations were integrated with an explicit Euler integration scheme sampled five times in a row with a time step of $0.2\,ms$.

First, the muscle force $F_m = F_{se}$, i.e. the force in the series elastic element SE, is computed from the current value of l_{se} ($= l_{mtu} - l_{ce}$): $F_{se} = F_{max}([(l_{se} - l_{slack})/(l_{slack}\epsilon_{ref})]^+)^2$, where F_{max} is the muscle maximal force and ϵ_{ref} ($= 0.04$) is the reference strain (with $[\bullet]^+ = max(0, \bullet)$). Then, the BE force is computed as follows: $F_{be} = F_{max}([l_{min} - l_{ce}]^+/(l_{opt}\epsilon_{be}))^2$ where l_{min} ($= 0.44\ l_{opt}$) is the BE rest length and ϵ_{be} ($= 0.28$) is the BE reference compression. F^*_{pe} is the PE muscle force divided by f_v (the force-velocity relationship) and is obtained as follows: $F^*_{pe} = F_{pe}/f_v = F_{max}([l_{ce} - l_{opt}]^+/(l_{opt}\epsilon_{pe}))^2$, where ϵ_{pe} ($= 0.56$) is the PE reference strain.

Then, the force-length relationship is computed as $f_l = exp(c||(l_{ce} - l_{opt})/w||^3)$, where w ($= 0.56\ l_{opt}$) is the width and c ($= ln(0.05)$) is the residual force factor. f_l is finally saturated to a lower bound of 10^{-3}. The force-velocity relationship is computed as follows: $f_v = (F_{se} + F_{be})/(A_m F_{max} f_l + F^*_{pe})$, and then saturated between 0 and 1.5. A_m is the muscle activation, being computed using the following first-order low-pass filter: $\tau_m\, dA_m/dt = S_m - A_m$, where τ_m is a time constant of $10\ ms$ and S_m is the muscle stimulation (see Figure 9.1). Finally, this allows to compute the force of the CE element as $F_{ce} = [F_{se} + F_{be} - F^*_{pe} f_v]^+$.

At the end, the CE velocity v_{ce} is obtained as $v_{ce} = -v_{max}\, l_{opt}\, ((1 - f_v)/(1 + K f_v))$ if $f_v < 1$ and $v_{ce} = -v_{max}\, l_{opt}\, ((f_v - 1)/(7.56\, K\, (f_v - N) + 1 - N))$ otherwise. In these equations, K ($= 5$) is the shape factor of f_v and N ($= 1.5$) is the eccentric force enhancement. All constant parameters used in these equations are reported in Table G.1.

Iterating over all these equations, the value of l_{ce} is progressively updated by integrating v_{ce}. Finally, the generated torque reference is computed as $\tau_{ref} = r_m F_m$ (see Figure 9.1).

To prevent the joints from exceeding a physiological range, similar joint soft limits as the ones reported in (Geyer and Herr, 2010) are used. Note that these limits do usually not engage, expect for the knee joint in over-extension.

Table G.1: The fixed MTU parameters of the 27 types of muscles for COMAN are reported in this table. When a leg MTU acts on different joints, they are explicitly reported as (a) for ankle, (k) for knee and (h) for hip. The sign ± means positive for the right leg and negative for the left one. The sign ∓ means the opposite. These values were extracted from (Geyer and Herr, 2010) for the leg sagittal muscles and from (Song and Geyer, 2013) for the hip lateral muscles. Finally, other muscles were estimated with the *OpenSim* simulator (Delp et al., 2007) with the human models developed in (Arnold et al., 2010) and in (Rajagopal et al., 2016). The masses m_{mtu} were obtained using the method proposed in (Wang et al., 2012), while the λ values were obtained from (Yamaguchi et al., 1990). These values are scaled to the size of COMAN by using dynamic scaling methods, being described in (Bejan and Marden, 2006) and (Schepelmann et al., 2012).

	F_{max} [N]	v_{max} [l_{opt}/s]	l_{opt} [mm]	l_{slack} [mm]	r_o [mm]	φ_{max} [deg]	φ_{ref} [deg]	ρ [-]	m_{mtu} [g]	λ [%]
SOL	1415	9	17	110	21	20	-10	0.5	240	81
TA	285	18	26	100	17	-10	20	0.7	70	70
GAS	530	18	21	170	21 (a) 21 (k)	20 (a) 40 (k)	-10 (a) 15 (k)	0.7	110	54
VAS	2125	18	34	98	26	15	55	0.7	720	50
HAM	1060	18	43	132	21 (k) 34 (h)	0 (k) - (h)	0 (k) -25 (h)	0.7	450	44
GLU	530	18	47	56	43	-	-30	0.5	250	50
HFL	710	18	47	43	43	-	0	0.5	330	50
HAB	1060	18	38	30	26	-	∓10	0.7	404	50
HAD	1595	18	43	77	13	-	∓15	1	676	57
EVE	375	18	21	107	13	∓10	∓5	0.7	80	57
INV	480	18	21	128	9	±5	∓10	0.7	100	55
HER	530	18	24	21	17	-	±10	0.8	180	50
HIR	570	18	34	30	13	-	∓20	0.7	192	50
BTR	640	18	43	43	49	45	0	1	270	50
BTL	640	18	43	43	49	-45	0	1	270	50
BET	1060	18	51	13	23	-45	0	1	540	57
BFL	830	18	48	53	35	40	5	1	390	50
BRR	560	18	47	45	15	-35	20	1	260	51
BRL	560	18	47	45	15	35	-20	1	260	51
SET	180	18	59	38	18	-70	-120	0.6	110	42
SFL	525	18	43	42	14	-30	-15	0.7	230	57
SAB	810	18	44	43	16	∓80	∓60	0.7	350	57
SAD	140	18	59	56	21	∓20	∓155	0.6	80	42
SER	430	18	32	13	12	∓35	0	0.7	140	45
SIR	650	18	39	22	12	∓40	∓25	0.7	250	58
EET	460	18	53	51	10	-25	-60	0.8	240	32
EFL	390	18	50	72	16	-70	-60	1	190	46

G.1.3 Metabolic energy

The model of (Bhargava et al., 2004) is used to compute the virtual muscle metabolic energy. This requires two additional MTU properties: its mass (m_{mtu}) and the mass fraction of slow twitch fibres (λ), which can be found in Table G.1.

The total rate of energy consumption is computed as $\dot{E}_{MTU} = \dot{A}_{MTU} + \dot{M}_{MTU} + \dot{S}_{MTU} + \dot{B}_{MTU} + \dot{W}_{MTU}$. The different terms are detailed below.

The activation heat rate is computed as a function of the stimulation S_m: $\dot{A}_{MTU} = m_{mtu}\,(40\,\lambda\,sin(\frac{\pi}{2}\,S_m) + 133\,(1-\lambda)(1-cos(\frac{\pi}{2}\,S_m)))$. The maintenance heat rate $\dot{M}_{MTU}$ depends on the activation A_m and requires to define the function $g(\tilde{l}_{ce})$ to model the dependence on the normalized muscle length $\tilde{l}_{ce} = l_{ce}/l_{opt}$. The function $g(\tilde{l}_{ce})$ is set to 0.5 for $\tilde{l}_{ce} < 0.5$, to $\tilde{l}_{ce}$ for $0.5 \leq \tilde{l}_{ce} < 1$, to $-2\,\tilde{l}_{ce} + 3$ for $1 \leq \tilde{l}_{ce} < 1.5$ and to 0 otherwise. Then, we compute $\dot{M}_{MTU} = m_{mtu}\,g(\tilde{l}_{ce})\,(74\,\lambda\,sin(\frac{\pi}{2}\,A_m) + 111\,(1-\lambda)\,(1-cos(\frac{\pi}{2}\,A_m)))$.

The shortening heat rate $\dot{S}_{MTU}$ is set to $[-0.25\,F_m\,v_{ce}]^+$, the basal metabolic rate $\dot{B}_{MTU}$ is set to $0.0225\,m_{mtu}$ and the work rate $\dot{W}_{MTU}$ is set to $[-F_{ce}\,v_{ce}]^+$. Finally, $\dot{E}_{MTU}$ is simply integrated with time.

G.2 CPG full equations

The following equations report the time derivatives of the neurons firing rate. Most parameters are optimized, their range being provided in Table G.2.

$$\dot{x}_1 = \frac{1}{\tau}(-x_1 - \beta_a v_1 - \eta_a [x_2]^+ - \eta_f [x_3]^+ - \eta_g [x_4]^+ + u_1)$$
$$\dot{x}_2 = \frac{1}{\tau}(-x_2 - \beta_a v_2 - \eta_a [x_1]^+ - \eta_g [x_3]^+ - \eta_f [x_4]^+ + u_2)$$
$$\dot{x}_3 = \frac{1}{\tau}(-x_3 - \beta_b v_3 - \eta_f [x_1]^+ - \eta_g [x_2]^+ - \eta_b [x_4]^+ + u_3)$$
$$\dot{x}_4 = \frac{1}{\tau}(-x_4 - \beta_b v_4 - \eta_g [x_1]^+ - \eta_f [x_2]^+ - \eta_b [x_3]^+ + u_4)$$
$$\dot{x}_A = \frac{1}{\tau}(-x_A - \beta_a v_A - \eta_f [x_3]^+ - \eta_g [x_4]^+ - \eta_a [x_B]^+ + u_A)$$
$$\dot{x}_B = \frac{1}{\tau}(-x_B - \beta_a v_B - \eta_g [x_3]^+ - \eta_f [x_4]^+ - \eta_a [x_A]^+ + u_B)$$
$$\dot{x}_C = \frac{1}{\tau}(-x_C - \beta_c v_C - \eta_h [x_3]^+ - \eta_i [x_4]^+ - \eta_c [x_D]^+ + u_C)$$
$$\dot{x}_D = \frac{1}{\tau}(-x_D - \beta_c v_D - \eta_i [x_3]^+ - \eta_h [x_4]^+ - \eta_c [x_C]^+ + u_D)$$
$$\dot{x}_E = \frac{1}{\tau}(-x_E - \beta_d v_E - \eta_j [x_3]^+ - \eta_k [x_4]^+ - \eta_d [x_F]^+ + u_E)$$
$$\dot{x}_F = \frac{1}{\tau}(-x_F - \beta_d v_F - \eta_k [x_3]^+ - \eta_j [x_4]^+ - \eta_d [x_E]^+ + u_F)$$
$$\dot{x}_G = \frac{1}{\tau}(-x_G - \beta_e v_G - \eta_l [x_1]^+ - \eta_m [x_2]^+ - \eta_e [x_H]^+ + u_G)$$
$$\dot{x}_H = \frac{1}{\tau}(-x_H - \beta_e v_H - \eta_m [x_1]^+ - \eta_l [x_2]^+ - \eta_e [x_G]^+ + u_H)$$

The fatigue dynamic equations are as below:

$$\dot{v}_1 = \frac{1}{\gamma_a \tau}(-v_1 + [x_1]^+) \qquad \dot{v}_C = \frac{1}{\gamma_c \tau}(-v_C + [x_C]^+)$$
$$\dot{v}_2 = \frac{1}{\gamma_a \tau}(-v_2 + [x_2]^+) \qquad \dot{v}_D = \frac{1}{\gamma_c \tau}(-v_D + [x_D]^+)$$
$$\dot{v}_3 = \frac{1}{\gamma_b \tau}(-v_3 + [x_3]^+) \qquad \dot{v}_E = \frac{1}{\gamma_d \tau}(-v_E + [x_E]^+)$$
$$\dot{v}_4 = \frac{1}{\gamma_b \tau}(-v_4 + [x_4]^+) \qquad \dot{v}_F = \frac{1}{\gamma_d \tau}(-v_F + [x_F]^+)$$
$$\dot{v}_A = \frac{1}{\gamma_a \tau}(-v_A + [x_A]^+) \qquad \dot{v}_G = \frac{1}{\gamma_e \tau}(-v_G + [x_G]^+)$$
$$\dot{v}_B = \frac{1}{\gamma_a \tau}(-v_B + [x_B]^+) \qquad \dot{v}_H = \frac{1}{\gamma_e \tau}(-v_H + [x_H]^+)$$

G.3 Excitations modulation

Modulations of the CPG excitations u_i are mainly performed to achieve the synchronization between neurons and foot strikes. In particular, x_1 is expected to fire (i.e. to become positive) just after the right strike and x_2 after the left strike. Therefore, all excitations u_i are set to zero if x_1 is positive during the right swing phase or x_2 is positive during the left swing phase.

On top of that, x_A and x_E are expected to fire after the right strike, while x_B and x_F should fire after the left strike. Also, neurons firing rates are allowed to be positive only during their corresponding supporting phase (i.e. stance phase without last double support). Finally, some neurons are inhibited to prevent the PD control acting on the sagittal torso angle θ_t (tracking θ_{ref}) to start earlier than expected (and similarly for the lateral torso angle Ψ_t tracking Ψ_{ref}). This is achieved with the following equations:

$$
\begin{aligned}
u_1 &= u - [x_1]^+_{SL} + [x_1]^-_{Str,R} & u_C &= u - [x_C]^+_{SL} \\
u_2 &= u - [x_2]^+_{SR} + [x_2]^-_{Str,L} & u_D &= u - [x_D]^+_{SR} \\
u_3 &= u - [x_3]^+_{SL} - [x_3]^+_{Str,L} & u_G &= u - [x_G]^+_{SL} \\
u_4 &= u - [x_4]^+_{SR} - [x_4]^+_{Str,R} & u_H &= u - [x_H]^+_{SR} \\
u_A &= u - [x_A]^+_{SL} + [x_A]^-_{Str,R} - [x_A]^+ [\theta_{ref} - \theta_t]^+_{1/0} \\
u_B &= u - [x_B]^+_{SR} + [x_B]^-_{Str,L} - [x_B]^+ [\theta_{ref} - \theta_t]^+_{1/0} \\
u_E &= u - [x_E]^+_{SL} + [x_E]^-_{Str,R} - [x_E]^+ [\Psi_{ref} - \Psi_t]^-_{1/0} \\
u_F &= u - [x_F]^+_{SR} + [x_F]^-_{Str,L} - [x_F]^+ [\Psi_{ref} - \Psi_t]^+_{1/0}
\end{aligned}
$$

where $u = 1$ is a tonic excitation. The function $[x]_{SR}$ is equal to x during the right leg supporting phase, to 0 otherwise. The function $[x]_{SL}$ is equal to x during the left leg supporting phase, to 0 otherwise. The $[x]_{Str,R}$ function is always equal to zero, except if the firing rate x_1 is still negative after the right foot strike. In this case, it is equal to x as long as x_1 is not the only positive RG neuron. The function $[\bullet]_{Str,L}$ is similar for the left leg and x_2. These functions are combined with the previously defined $[\bullet]^+$ and $[\bullet]^-$ functions. The $[\bullet]^+_{1/0}$ function returns 1 if its argument is positive, 0 otherwise (similarly for $[\bullet]^-_{1/0}$ with negative arguments). Finally, only the positive values of all excitations u_i are used (i.e. $[u_i]^+$).

It should be noted that most of the time, these excitations are kept to the tonic excitation u. Indeed, their modulations are usually very short.

Finally, to guarantee that the CPG quickly converges to its requested state, different excitations are used during the first 0.2 s of the gait. More specifically, all u_i are set to 0, except u_2, u_B, u_D and u_F (if the right leg is the first to enter in swing phase) or u_1, u_A, u_C and u_E (otherwise), which are set to 1.

G.4 Muscles stimulations

The following sections detail the muscle stimulations implementation. As in (Geyer and Herr, 2010), time delays were applied to some reflex inputs, to capture long (t_l), medium (t_m) and short (t_s) neural signal delays. Stimulations are further bounded between $S_{MIN} = 0.01$ and $S_{MAX} = 1$. All parameters to be optimized are reported in Table G.2.

G.4.1 Leg proximal muscles

The leg proximal muscles (i.e. HFL, GLU, HAM, HAB and HAD) are the main ones in charge of adapting the gait speed.

Based on the CPG firing rates x_i, the CPG output signals y_i are computed. Similarly to (Van der Noot et al., 2015b), we use $y_i = [[x_a]^+ - [x_b]^+]^+$, where x_a is a PF neuron and x_b a RG neuron directly connected to x_a. The $[x_b]^+$ contribution purpose is to decrease the output strength when x_a and x_b are firing at the same time. However, its influence is rather small, i.e. $y_i \simeq [x_a]^+$:

$$
\begin{aligned}
y_1 &= [[x_A]^+ - [x_3]^+]^+ \qquad & y_5 &= [[x_E]^+ - [x_3]^+]^+ \\
y_2 &= [[x_B]^+ - [x_4]^+]^+ \qquad & y_6 &= [[x_F]^+ - [x_4]^+]^+ \\
y_3 &= [[x_C]^+ - [x_3]^+]^+ \qquad & y_7 &= [[x_G]^+ - [x_1]^+]^+ \\
y_4 &= [[x_D]^+ - [x_4]^+]^+ \qquad & y_8 &= [[x_H]^+ - [x_2]^+]^+
\end{aligned}
$$

The muscles stimulations of the proximal muscles (CPG contribution) are computed as follows:

$$
\begin{aligned}
S_{GLU,R} &= k_{GLU,1}\, y_1 + k_{GLU,2}\, y_8 \\
S_{GLU,L} &= k_{GLU,1}\, y_2 + k_{GLU,2}\, y_7 \\
S_{HAM,R} &= k_{HAM,1}\, y_1 + k_{HAM,2}\, y_2 + k_{HAM,3}\, y_8 \\
S_{HAM,L} &= k_{HAM,1}\, y_2 + k_{HAM,2}\, y_1 + k_{HAM,3}\, y_7 \\
S_{HFL,R} &= k_{HFL}\, y_4 \quad ; \quad S_{HAB,R} = k_{HAB}\, y_5 \\
S_{HFL,L} &= k_{HFL}\, y_3 \quad ; \quad S_{HAB,L} = k_{HAB}\, y_6
\end{aligned}
$$

The following equations are systematically doubled: one for the right leg, and the other for the left leg. To capture this, we used the $\{x, y\}$ notation: the first item refers to the right leg, and the second to the left one. In particular, $\{R, L\}$ stands for right or left leg. During the stance phase, the PD control applied to the torso sagittal lean angle is computed as follows: $\Delta_{\theta,\{R,L\}} = (k_{p,\theta}\,(\theta_{ref} - \theta_t(t_s)) - k_{d,\theta}\,\dot{\theta}_t(t_s))\,\tilde{F}_{gd,\{R,L\}}(t_s)$, where $k_{p,\theta}$, $k_{d,\theta}$ and θ_{ref} are parameters to be optimized. θ_t is the torso sagittal lean angle and $\dot{\theta}_t$ is its derivative. Finally, $\tilde{F}_{gd,\{R,L\}}$ is the vertical force below the corresponding foot, normalized to the walker weight. Then, the HFL stimulation is incremented by $[\Delta_{\theta,\{R,L\}}]^+$. The $[\Delta_{\theta,\{R,L\}}]^-$ signal is added to the GLU stimulation,

as long as the condition $[y_1 = 0 \;\&\; y_4 = 0]$ is met for the right leg or the condition $[y_2 = 0 \;\&\; y_3 = 0]$ is met for the left leg (to prevent contradictory signals between the CPG and reflexes).

In the following notations, δ equals 1 for the right leg and -1 for the left one. For HAB and HAD muscles, additional reflexes are added. The PD control applied during the supporting phase on the lateral lean angle is computed as follows: $\Delta_{\Psi,\{R,L\}} = (k_{p,\Psi}(\delta\,\Psi_{ref} - \Psi_t(t_s)) - k_{d,\Psi}\,\dot{\Psi}_t(t_s))\,\tilde{F}_{gd,\{R,L\}}(t_s)$, where $k_{p,\Psi}$, $k_{d,\Psi}$ and Ψ_{ref} are parameters to be optimized. Ψ_t is the torso lateral lean angle and $\dot{\Psi}_t$ is its derivative. Then, the stimulation S_{HAD} is computed as $S_{MIN} + [\Delta_{\Psi,\{R,L\}}]^{\{-/+\}}$. The $[\Delta_{\Psi,\{R,L\}}]^{\{+/-\}}$ signal is added to the HAB stimulation, provided the condition $[y_5 = 0]$ is fulfilled for the right leg or that $[y_6 = 0]$ is met for the left one.

During the contralateral leg supporting phase, the lateral hip reference position is computed as $\varphi_{h,l,ref,\{R,L\}} = -k_{p,\Lambda,h}\,(-\delta\,\Lambda_{ref,h} - \Delta_{com,\{L,R\}}(t_s)) + k_{d,\Lambda,h}\,\dot{\Delta}_{com,\{L,R\}}(t_s)$ where $k_{p,\Lambda,h}$, $k_{d,\Lambda,h}$ and $\Lambda_{ref,h}$ are control parameters to be optimized. $\Delta_{com,L}$ is the COM lateral position, relative to the left foot, $\dot{\Delta}_{com,L}$ its derivative. In order to decrease leg inter-penetration, $\varphi_{h,l,ref,R}$ is limited to an upper bound of 7.5° and $\varphi_{h,l,ref,L}$ to a lower bound of −7.5°. Then, the PD controller tracking this hip lateral position is computed as $\Delta_{h,sw,\{R,L\}} = k_{p,\varphi,h}\,(\varphi_{h,l,ref,\{R,L\}} - \varphi_{h,l,\{R,L\}}(t_s)) - k_{d,\varphi,h}\,\dot{\varphi}_{h,l,\{R,L\}}(t_s)$, where $k_{p,\varphi,h}$ and $k_{d,\varphi,h}$ are parameters to be optimized. $\varphi_{h,l,\{R,L\}}$ is the hip lateral position and $\dot{\varphi}_{h,l,\{R,L\}}$ is its derivative. HAB and HAD stimulations are then computed as $S_{HAB,\{R,L\}} = S_{MIN} + [\Delta_{h,sw,\{R,L\}}]^{\{-/+\}}$ and $S_{HAD,\{R,L\}} = S_{MIN} + [\Delta_{h,sw,\{R,L\}}]^{\{+/-\}}$.

To support gait initialization, special stimulations are sent to the HAB and HAD muscles. More specifically, during an initial time $T_{sw,in}$, the first leg to enter in the swing phase receives a stimulation $S_{sw,in}$ for the HAB muscle, while the HAD muscle only receives the minimal stimulation S_{MIN}. Similarly, during an initial time $T_{st,in}$, the other leg (first in stance) receives a stimulation $S_{st,in}$ for the HAB muscle and S_{MIN} for the HAD muscle. These parameters are reported in Table G.2.

Finally, the leg transverse muscles (i.e. HER and HIR) are actuated by the following PD controller: $\Delta_{trans,\{R,L\}} = -500\,\varphi_{h,t,\{R,L\}}(t_s) - 20\,\dot{\varphi}_{h,t,\{R,L\}}(t_s)$, where $\varphi_{h,t,\{R,L\}}$ is the hip joint transverse position and $\dot{\varphi}_{h,t,\{R,L\}}$ is its derivative. The corresponding stimulations are the following: $S_{HER,\{R,L\}} = S_{MIN} + [\Delta_{trans,\{R,L\}}]^{\{-,+\}}$ and $S_{HIR,\{R,L\}} = S_{MIN} + [\Delta_{trans,\{R,L\}}]^{\{+,-\}}$.

G.4.2 Leg distal muscles

In the sagittal plane, the leg distal muscles (i.e. VAS, GAS, TA and SOL) are mainly based on reflexes, as detailed in (Geyer and Herr, 2010). The following rules hold during stance phase: $S_{VAS} = S_{MIN} + G_{VAS}\,\tilde{F}_{VAS}(t_m)$; $S_{GAS} = S_{MIN} + G_{GAS}\,\tilde{F}_{GAS}(t_l)$; $S_{TA} = S_{MIN} + G_{TA,st}\,(\tilde{l}_{ce,TA}(t_l) - l_{TA,st}) - G_{SOL,TA}\,\tilde{F}_{SOL}(t_l)$; $S_{SOL} = S_{MIN} + G_{SOL}\,\tilde{F}_{SOL}(t_l)$. The parameters G_{VAS}, G_{GAS}, $G_{TA,st}$, $G_{SOL,TA}$, G_{SOL} and $l_{TA,st}$ are optimized. $\tilde{F}_m$ is the muscle force normalized by its maximal force F_{max} and $\tilde{l}_m$ is the muscle CE length l_{CE} normalized by its l_{opt} value. On top of this, the VAS muscle is inhibited (i.e. $S_{VAS} = S_{MIN}$) when close to knee over-extension, i.e. when $[\varphi_k(t_m) < \varphi_{k,th} \;\&\; \dot{\varphi}_k(t_m) < 0]$, where φ_k is the knee position and $\dot{\varphi}_k$ is its derivative. This inhibition is also applied during

the double support phase, detected by the CPG firing rate condition $[x_2 > 0.05]$ (for the right leg) and $[x_1 > 0.05]$ (for the left leg). During the swing phase, all leg distal sagittal muscles only receive S_{MIN}, except the TA, which gets an extra term to guarantee a proper foot clearance: $S_{TA} = S_{MIN} + G_{TA,sw}(\tilde{l}_{ce,TA}(t_l) - l_{TA,sw})$, where $G_{TA,sw}$ and $l_{TA,sw}$ are optimized.

During the supporting phase, the leg foot lateral muscles are activated by a PD controller on the COM lateral position, i.e. $\Delta_{f,sp,\{R,L\}} = k_{p,\Lambda,f}(\delta\Lambda_{ref,f} - \Delta_{com,\{R,L\}}(t_s)) - k_{d,\Lambda,f}\dot{\Delta}_{com,\{R,L\}}(t_s)$. $k_{p,\Lambda,f}$, $k_{d,\Lambda,f}$ and $\Lambda_{ref,f}$ are optimized parameters. Corresponding stimulations (i.e. acting on EVE and INV) are the following: $S_{EVE,\{R,L\}} = S_{MIN} + [\Delta_{f,sp,\{R,L\}}]^{\{-/+\}}$ and $S_{INV,\{R,L\}} = S_{MIN} + [\Delta_{f,sp,\{R,L\}}]^{\{+/-\}}$. During the other leg supporting phase, a PD controller on the foot lateral orientation is applied as $\Delta_{f,sw,\{R,L\}} = -k_{p,\Psi,f}\Psi_{f,\{R,L\}}(t_l) - k_{d,\Psi,f}\dot{\Psi}_{f,\{R,L\}}(t_l)$, where $k_{p,\Psi,f}$ and $k_{d,\Psi,f}$ are optimized, $\Psi_{f,\{R,L\}}$ is the foot lateral orientation (relative to the ground) and $\dot{\Psi}_{f,\{R,L\}}$ is its derivative. The corresponding stimulations are the following: $S_{EVE,\{R,L\}} = S_{MIN} + [\Delta_{f,sw,\{R,L\}}]^{\{-,+\}}$ and $S_{INV,\{R,L\}} = S_{MIN} + [\Delta_{f,sw,\{R,L\}}]^{\{+,-\}}$.

G.4.3 Upper-body muscles

Most torso muscles track a constant position reference q_{ref} using the following PD control rule: $f_{PD,t}(q_{ref}) = 500(q_{ref} - q) - 20\dot{q}$, where q is the joint position, and $\dot{q}$ is its derivative. Then, the torso muscles BET, BFL, BTL, BTR corresponding stimulations are the following: $S_{BET} = [f_{PD,t}(0°)]^-$, $S_{BFL} = [f_{PD,t}(0°)]^+$, $S_{BTL} = [f_{PD,t}(0°)]^-$ and $S_{BTR} = [f_{PD,t}(0°)]^+$. For the remaining torso muscles (i.e. BRL and BRR), the RG neurons are used to control the torso transverse joint: $S_{BRL} = k_{torso}[x_1]^+ + k_{torso}[x_3]^+$; $S_{BRR} = k_{torso}[x_2]^+ + k_{torso}[x_4]^+$, where k_{torso} is a unique parameter to be optimized.

Similarly, most arms muscles track a position reference q_{ref} with the following control: $f_{P,a}(q_{ref}) = 5(q_{ref} - q)$. Then, the resulting SAB, SAD, SER, SIR, EET and EFL stimulations are the following: $S_{SAB,\{R,L\}} = [f_{P,a}(-\delta 5°)]^{\{-,+\}}$, $S_{SAD,\{R,L\}} = [f_{P,a}(-\delta 5°)]^{\{+,-\}}$, $S_{SER,\{R,L\}} = [f_{P,a}(\delta 7.5°)]^{\{-,+\}}$, $S_{SIR,\{R,L\}} = [f_{P,a}(\delta 7.5°)]^{\{+,-\}}$, $S_{EET} = [f_{P,a}(-25°)]^+$ and $S_{EFL} = [f_{P,a}(-25°)]^-$. Finally, the RG neurons are used to control the arms remaining muscles SFL and SET: $S_{SFL,\{R,L\}} = k_{arms}[x_{\{1,2\}}]^+ + k_{arms}[x_{\{3,4\}}]^+$; $S_{SET,\{R,L\}} = k_{arms}[x_{\{2,1\}}]^+ + k_{arms}[x_{\{4,3\}}]^+$, where k_{arms} is set to an arbitrary value of 0.75.

Table G.2: The parameters to be optimized in the controller, and their ranges are reported in this table. The speed dependent parameters are computed as follows: $\tau = K_\tau + L_\tau\, v_* + M_\tau\, v_*^2$; $k_{HAB} = K_{HAB} + L_{HAB}\, v_* + M_{HAB}\, v_*^2$; $k_{HFL} = K_{HFL} + L_{HFL}\, v_*$; $k_{GLU,1} = K_{GLU,1} + L_{GLU,1}\, v_*$; $k_{HAM,1} = K_{HAM,1} + L_{HAM,1}\, v_*$; $k_{HAM,2} = K_{HAM,2} + L_{HAM,2}\, v_* + M_{HAM,2}\, v_*^2$; $k_{HAM,3} = K_{HAM,3} + L_{HAM,3}\, v_*$; $\theta_{ref} = K_\theta + L_\theta\, v_*$; $\Lambda_{ref,h} = K_{\Lambda,h} + L_{\Lambda,h}\, v_* + M_{\Lambda,h}\, v_*^2$, where $v_* = v_{ref} - 0.65$ and v_{ref} is the target forward speed. When only a single speed was optimized, all the terms related to v_* and v_*^2 were removed. On top of that, the remaining speed parameters (i.e. labelled as $K_\bullet$) received a higher range, close to the bounds of the vertical axes of Figure 9.6.

	min	max		min	max		min	max		min	max
β			η_j	3.5	4.5	$L_{HAM,1}$	3	7	**reflex (l)**		
β_a	5	6.5	η_k	3.5	5	$L_{HAM,2}$	-1.3	-0.3	$k_{p,\Psi}$	10	15
β_b	3	4.5	η_l	2.5	3.5	$L_{HAM,3}$	-0.35	-0.2	$k_{d,\Psi}$	1.5	2.5
β_c	2.5	5	η_m	3	4	L_θ	0.2	0.35	$k_{p,\Lambda,h}$	1	2.5
β_d	4	6.5	**const**			$L_{\Lambda,h}$	-0.04	0.06	$k_{d,\Lambda,h}$	0.1	0.4
β_e	3	4.5	$k_{GLU,2}$	0	0.15	M_τ	-0.08	0	$k_{p,\varphi,h}$	3.5	5.5
γ			Ψ, ref	0.03	0.05	M_{HAB}	-1.5	0	$k_{d,\varphi,h}$	0.2	0.5
γ_a	2	4	**speed**			$M_{HAM,2}$	1	3	$k_{p,\Psi,f}$	12	18
γ_b	2	3.5	K_τ	0.078	0.085	$M_{\Lambda,h}$	0	0.3	$k_{d,\Psi,f}$	0.5	1
γ_c	2.5	5.5	K_{HAB}	1.4	2.2	**reflex (s)**			$k_{p,\Lambda,f}$	70	120
γ_d	1	2	K_{HFL}	3.5	6	G_{SOL}	0.85	1.05	$k_{d,\Lambda,f}$	10	20
γ_e	2.5	4	$K_{GLU,1}$	2.5	3.5	$G_{SOL,TA}$	0.3	1	$\Lambda_{ref,f}$	0.03	0.06
η			$K_{HAM,1}$	2	3	$G_{TA,sw}$	1.5	4	**init**		
η_a	3.5	6	$K_{HAM,2}$	0.4	1	$G_{TA,st}$	1.5	2.5	$T_{st,in}$	0.1	0.4
η_b	4.5	7	$K_{HAM,3}$	0	0.1	G_{GAS}	0.2	0.8	$T_{sw,in}$	0	0.3
η_c	3.5	5.5	K_θ	0.18	0.25	G_{VAS}	25	35	$S_{st,in}$	0.6	1
η_d	5.5	7	$K_{\Lambda,h}$	0.04	0.09	$l_{TA,sw}$	0.8	0.9	$S_{sw,in}$	0	0.5
η_e	4	6.5	L_τ	-0.04	-0.01	$l_{TA,st}$	0.55	0.65	X_{init}	0.03	0.07
η_f	2	4	L_{HAB}	-1	0.4	$\varphi_{k,th}$	0	0.3	Y_{init}	0	0.03
η_g	3	4.5	L_{HFL}	2.5	4	$k_{p,\theta}$	4	10	**upper**		
η_h	3.5	5	$L_{GLU,1}$	0.2	1.5	$k_{d,\theta}$	0.2	0.8	k_{torso}	0.07	0.11
η_i	3.5	5									

G.5 External forces

Two types of custom-made contact models were used: (i) the mesh-based one (used for GCM) and (ii) the volume penetration one.

G.5.1 Mesh-based contact

The mesh-based CGM is very similar to the one implemented in (Geyer and Herr, 2010) and (Song and Geyer, 2013). More specifically, a regular mesh of 20 points is used under each foot. This number was selected as a compromise between computational cost and the accuracy of contacts with uneven grounds (see Experiment 9). Each foot point can reach three different states: (i) swing state (when it is not in contact with the ground), (ii) sliding state and (iii) stiction state.

Swing state is reached when the point is above the ground level. In such a case, no force is applied to it. When the point penetrates the ground, it first switches to the sliding state. Then, when the point tangential velocity gets lower than $1\,cm/s$, the point switches to the stiction state. In this state, if the tangential force norm $||\boldsymbol{F}_T||$ exceeds $\mu_{st}||\boldsymbol{F}_N||$ ($\mu_{st} = 0.9$ is the static friction coefficient, $\boldsymbol{F}_N$ is the point normal force), the point goes back to the sliding state.

When in sliding or stiction state, the normal force norm is computed as $||\boldsymbol{F}_N|| = -k_{p,N}\,\Delta_N\,[1 - k_{d,N}\,\dot{\Delta}_N]^+$ where $k_{p,N}$ is set to $81.5\,kN/m$ and $k_{d,N}$ is set to $30\,s/m$. Δ_N is the point penetration in the ground along its normal (negative according to the frames of Figure 9.2), $\dot{\Delta}_N$ is its time derivative.

During stiction, the vectorial tangential force is computed as $\boldsymbol{F}_T = -k_{p,T}\,\boldsymbol{\Delta}_T\,[1+k_{d,T}\,sgn(\boldsymbol{\Delta}_T)\,\boldsymbol{v}_T]^+$, where $k_{p,T}$ is set to $8.2\,[kN/m]$, and $k_{d,T}$ to $30\,[s/m]$. The $sgn(\bullet)$ function returns 1 when its argument is positive, and -1 otherwise. The vector $\boldsymbol{\Delta}_T$ contains the two tangential components of the distance between the point current position and the previous one when entering in stiction mode. Finally, $\boldsymbol{v}_T$ is the point tangential velocity.

During the sliding phase, the tangential force is computed as $\boldsymbol{F}_T = -\mu_{sl}\,F_N\,(\boldsymbol{v}_T/||\boldsymbol{v}_T||)$, where $\mu_{sl} = 0.8$ is the sliding friction coefficient.

G.5.2 Volume penetration contact

This contact model is only used in the experiment testing the walker robustness to flying balls (see Figure 9.13), in order to compute the contacts between the COMAN body and the balls thrown to it. The robot body is approximated by two types of volume primitives: spheres and cuboids. Then, iterating through the different volume primitives (COMAN bodies and balls), different volume penetrations V_i (i.e. intersection volume between two different bodies) and their time derivative $\dot{V}_i$ are computed.

For each $V_i \neq 0$, a normal repulsive force is computed as $||\boldsymbol{F}_{N,i}|| = l_{p,N}\,V_i\,[1 + l_{d,N}\,\dot{V}_i]^+$, where $l_{p,N}$ is set to $10^9\,[N/(m^3)]$ and $l_{d,N}$ is set to $10^3\,[s/(m^3)]$. The tangential force is computed as $\boldsymbol{F}_{T,i} = -\mu\,||\boldsymbol{F}_{N,i}||\,tanh(\beta_T\,||\boldsymbol{v}_{T,i}||)\,(\boldsymbol{v}_{T,i}/||\boldsymbol{v}_{T,i}||)$, where $\mu = 0.9$, $\beta_T = 10\,[s/m]$, $tanh$ is the hyperbolic tangent function and $\boldsymbol{v}_{T,i}$ is the relative tangential speed between the two bodies. Finally, these forces are applied (with opposite directions) at the center of the contact surface between the two bodies.

G.6 Lack of fit

The sum of squares due to lack of fit (Smith and Rose, 1995) analysis is presented here for one of the eleven key parameters displayed in Figure 9.6.

First, the polynomial approximation of orders 0, 1 and 2 are computed, based on the least squares method. For the n (= 10) target speeds, the corresponding sum of squares due to lack of fit is computed as $SSLF = \sum_{i=1}^{n} n_i(\bar{Y}_i - \hat{Y}_i)^2$, where n_i (= 10) is the number of trials performed for each speed, $\bar{Y}_i$ is the mean of these n_i trials and $\hat{Y}_i$ is the regression performed for this speed. Similarly, the sum of squares due to pure error is computed as $SSPE = \sum_{i=1}^{n}\sum_{j=1}^{n_i}(Y_{ij} - \bar{Y}_i)^2$, where Y_{ij} is the j^{th} measure performed for the target speed i.

Next, the corresponding F-distribution can be computed as $F = (SSLF/(n-p))/(SSPE/(N-n))$, where p is the number of parameters of the regression (1, 2 or 3, respectively for orders 0, 1 and 2) and N ($= n \cdot n_i$) is the total number of measures. Using the null hypothesis that the regression model is adequate, the corresponding p-value are computed based on this F-distribution value and on the following degrees of freedom: $n-p$ and $N-n$.

H Steering control in a 3D environment

The material presented in this appendix is related to Chapter 10, about the extension of the 3D walking neuromuscular controller (see Chapter 9) to control steering (i.e. heading and turning curvature).

H.1 Stimulations for heading control

The stimulations computation rules from Chapter 9 (and from their related appendices in Chapter G) affected by the curved motion updates (see Section 10.3) are summarized here. More details (e.g. time delays) are provided in Chapters 9 and G. Variables presented in Section 10.3 are not detailed here.

The hip lateral joints are controlled by the HAB and HAD muscles. First, the HAB muscles receive stimulations coming from the CPG. These stimulations are mainly proportional to $[x_E]^+$ (excited by u_E) for the right leg and to $[x_F]^+$ (excited by u_F) for the left leg.

During the leg supporting phase, the following proportional-derivative (PD) control is applied: $\Delta_{\Psi,\{R,L\}} = (k_{p,\Psi}\,(\delta\,\Psi^*_{ref,\{R,L\}} - \Psi_t) - k_{d,\Psi}\,\dot{\Psi}_t)\,\tilde{F}_{gd,\{R,L\}}$, where $k_{p,\Psi}$ and $k_{d,\Psi}$ are parameters to optimize, Ψ_t is the torso lateral lean angle and $\dot{\Psi}_t$ is its derivative. δ equals 1 for the right leg and -1 for the left one. Finally, $\tilde{F}_{gd,\{R,L\}}$ is the vertical force below the corresponding foot, normalized to the walker weight. Then, HAB and HAD muscles are mainly commanded by a stimulation equal to $[\Delta_{\Psi,\{R,L\}}]^+$ or $[\Delta_{\Psi,\{R,L\}}]^-$.

During the contralateral leg supporting phase, a hip lateral reference angle $\varphi_{h,l,ref,\{R,L\}}$ is computed as $-k_{p,\Lambda,h}\,(-\delta\,\Lambda^*_{ref,h,\{R,L\}} - \Delta_{com,\{L,R\}}) + k_{d,\Lambda,h}\,\dot{\Delta}_{com,\{L,R\}}$, where $k_{p,\Lambda,h}$ and $k_{d,\Lambda,h}$ are control parameters to optimize, $\Delta_{com,L}$ is the COM lateral position, relative to the left foot and $\dot{\Delta}_{com,L}$ its derivative (similar for $\Delta_{com,R}$ and $\dot{\Delta}_{com,R}$ relative to the right foot). The resulting local angle reference $\varphi_{h,l,ref,\{R,L\}}$ is later maintained by using a similar PD control rule as described above (i.e. for the supporting phase), with similar stimulations sent to HAB and HAD.

The hip transverse joints are controlled with the following PD computation: $\Delta_{trans,\{R,L\}} = 500\,(\varphi_{h,t,ref,\{R,L\}} - \varphi_{h,t,\{R,L\}}) - 20\,\dot{\varphi}_{h,t,\{R,L\}})$, where $\varphi_{h,t,\{R,L\}}$ is the hip joint transverse position and $\dot{\varphi}_{h,t,\{R,L\}}$ is its derivative. Stimulation equal to $[\Delta_{trans,\{R,L\}}]^+$ or $[\Delta_{trans,\{R,L\}}]^-$ are then sent to the HER and HIR muscles.

H.2 Optimization parameters

The parameters to be optimized in the controller, and their ranges are reported Table H.1: the transverse (t) and lateral (l) leg parameters, as well as the CPG-related parameters. The speed dependent parameters are computed as follows: $k_{y,in} = K_{y,in} + L_{y,in}\,v_*$; $k_{y,out} = K_{y,out} + L_{y,out}\,v_* + M_{y,out}\,v_*^2$; $\Delta_\Lambda = K_{\Delta,\Lambda} + L_{\Delta,\Lambda}\,v_*$; $\Delta_\Psi = K_{\Delta,\Psi} + L_{\Delta,\Psi}\,v_* + M_{\Delta,\Psi}\,v_*^2$; $\eta_o = K_{\eta,o} + L_{\eta,o}\,v_*$; $\nu_l = K_{\nu,l} + L_{\nu,l}\,v_* + M_{\nu,l}\,v_*^2$, where $v_* = v_{ref} - 0.65$ and v_{ref} is the target forward speed.

Table H.1: Optimization parameters and their bounds

	min	max		min	max
leg (t)			**leg (l)**		
$K_{y,in}$	0.1	0.4	$K_{\Delta,\Lambda}$	0.2	1
$K_{y,out}$	0.1	0.5	$K_{\Delta,\Psi}$	0	0.3
$L_{y,in}$	0	0.4	$L_{\Delta,\Lambda}$	0	2
$L_{y,out}$	-0.6	0.2	$L_{\Delta,\Psi}$	-0.2	0.6
$M_{y,out}$	0	6	$M_{\Delta,\Psi}$	0	4
CPG (η)			**CPG (ν)**		
η_n	1.6	3.2	$K_{\nu,l}$	0.05	0.35
$K_{\eta,o}$	0.4	1.4	$L_{\nu,l}$	-0.15	0.15
$L_{\eta,o}$	0	2	$M_{\nu,l}$	0	4

Bibliography

Albus JS (1971) A theory of cerebellar function. *Mathematical Biosciences* 10(1): 25–61. DOI: 10.1016/0025-5564(71)90051-4.

Anderson FC and Pandy MG (2001) Dynamic Optimization of Human Walking. *Journal of Biomechanical Engineering* 123(5): 381–390. DOI:10.1115/1.1392310.

Aoi S, Ogihara N, Funato T, Sugimoto Y and Tsuchiya K (2010) Evaluating functional roles of phase resetting in generation of adaptive human bipedal walking with a physiologically based model of the spinal pattern generator. *Biological Cybernetics* 102(5): 373–387. DOI: 10.1007/s00422-010-0373-y.

Aoi S and Tsuchiya K (2005) Locomotion Control of a Biped Robot Using Nonlinear Oscillators. *Autonomous Robots* 19(3): 219–232. DOI:10.1007/s10514-005-4051-1.

Arnold EM, Ward SR, Lieber RL and Delp SL (2010) A model of the lower limb for analysis of human movement. *Annals of biomedical engineering* 38(2): 269–79. DOI:10.1007/s10439-009-9852-5.

Bejan A and Marden JH (2006) Unifying constructal theory for scale effects in running, swimming and flying. *The Journal of experimental biology* 209(Pt 2): 238–48. DOI: 10.1242/jeb.01974.

Bhargava LJ, Pandy MG and Anderson FC (2004) A phenomenological model for estimating metabolic energy consumption in muscle contraction. *Journal of Biomechanics* 37(1): 81–88. DOI:10.1016/S0021-9290(03)00239-2.

Bovi G, Rabuffetti M, Mazzoleni P and Ferrarin M (2011) A multiple-task gait analysis approach: Kinematic, kinetic and EMG reference data for healthy young and adult subjects. *Gait & Posture* 33(1): 6–13. DOI:10.1016/j.gaitpost.2010.08.009.

Bussel B, Roby-Brami A, Azouvi P, Biraben A, Yakovleff A and Held JP (1988) Myoclonus in a patient with spinal cord transection. Possible involvement of the spinal stepping generator. *Brain: A Journal of Neurology* 111 (Pt 5): 1235–1245.

Calancie B (2006) Spinal Myoclonus After Spinal Cord Injury. *The Journal of Spinal Cord Medicine* 29(4): 413–424.

Cavanagh PR and Kram R (1989) Stride length in distance running: velocity, body dimensions, and added mass effects. *Medicine and Science in Sports and Exercise* 21(4): 467–479.

Chang CC and Lin CJ (2011) LIBSVM: A Library for Support Vector Machines. *ACM Trans. Intell. Syst. Technol.* 2(3): 27:1–27:27. DOI:10.1145/1961189.1961199.

Chestnutt J, Lau M, Cheung G, Kuffner J, Hodgins J and Kanade T (2005) Footstep Planning for the Honda ASIMO Humanoid. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. IEEE, pp. 629–634. DOI:10.1109/ROBOT.2005.1570188.

Chung JH, Yi BJ and Han CS (2009) Intelligent Mobility Playing the Role of Impulse Absorption. In: Asama H, Kurokawa H, Ota J and Sekiyama K (eds.) *Distributed Autonomous Robotic Systems 8*. Springer Berlin Heidelberg, pp. 109–119. DOI: 10.1007/978-3-642-00644-9_10.

Clerc M and Kennedy J (2002) The particle swarm - explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation* 6(1): 58–73. DOI:10.1109/4235.985692.

Colasanto L, Van der Noot N and Ijspeert AJ (2015) Bio-inspired walking for humanoid robots using feet with human-like compliance and neuromuscular control. In: *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 26–32. DOI:10.1109/HUMANOIDS.2015.7363518.

Collins S and Ruina A (2005) A Bipedal Walking Robot with Efficient and Human-Like Gait. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. IEEE, pp. 1983–1988. DOI:10.1109/ROBOT.2005.1570404.

Collins SH, Adamczyk PG and Kuo AD (2009) Dynamic arm swinging in human walking. *Proceedings. Biological sciences / The Royal Society* 276(1673): 3679–3688. DOI:10.1098/rspb.2009.0664.

Courtine G, Papaxanthis C and Schieppati M (2006) Coordinated modulation of locomotor muscle synergies constructs straight-ahead and curvilinear walking in humans. *Experimental brain research* 170(3): 320–35. DOI:10.1007/s00221-005-0215-7.

Courtine G and Schieppati M (2003) Human walking along a curved path. I. Body trajectory, segment orientation and the effect of vision. *The European Journal of Neuroscience* 18(1): 177–190.

Daley MA, Felix G and Biewener AA (2007) Running stability is enhanced by a proximo-distal gradient in joint neuromechanical control. *The Journal of experimental biology* 210(Pt 3): 383–394. DOI:10.1242/jeb.02668.

Dallali H (2011) *Modelling and dynamic stabilization of a compliant humanoid robot, CoMan*. PhD Thesis, University of Manchester.

Dallali H, Mosadeghzad M, Medrano-Cerda Ga, Docquier N, Kormushev P, Tsagarakis N, Li Z and Caldwell D (2013) Development of a dynamic simulator for a compliant humanoid robot based on a symbolic multibody approach. In: *2013 IEEE International Conference on Mechatronics, ICM 2013*. IEEE, pp. 598–603. DOI:10.1109/ICMECH.2013.6519110.

Danner SM, Hofstoetter US, Freundl B, Binder H, Mayr W, Rattay F and Minassian K (2015) Human spinal locomotor control is based on flexibly organized burst generators. *Brain: A Journal of Neurology* 138(Pt 3): 577–588. DOI:10.1093/brain/awu372.

Davis S and Caldwell DG (2010) The design of an anthropomorphic dexterous humanoid foot. In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2200–2205. DOI:10.1109/IROS.2010.5649756.

de Rugy A and Sternad D (2003) Interaction between discrete and rhythmic movements: reaction time and phase of discrete movement initiation during oscillatory movements. *Brain Research* 994(2): 160–174.

Dekker MHP (2009) Zero-Moment Point Method for Stable Biped Walking. Technical report, Eindhoven, University of Technology.

Delp SL, Anderson FC, Arnold AS, Loan P, Habib A, John CT, Guendelman E and Thelen DG (2007) OpenSim: open-source software to create and analyze dynamic simulations of movement. *IEEE transactions on bio-medical engineering* 54(11): 1940–50. DOI:10.1109/TBME.2007.901024.

Desai R and Geyer H (2013) Muscle-reflex control of robust swing leg placement. In: *2013 IEEE International Conference on Robotics and Automation*. IEEE, pp. 2169–2174. DOI:10.1109/ICRA.2013.6630868.

Dimitrijevic MR, Gerasimenko Y and Pinter MM (1998) Evidence for a spinal central pattern generator in humans. *Annals of the New York Academy of Sciences* 860: 360–376. DOI:10.1111/j.1749-6632.1998.tb09062.x.

Docquier N, Poncelet A and Fisette P (2013) ROBOTRAN: a powerful symbolic generator of multibody models. *Mechanical Sciences* 4(1): 199–219. DOI:10.5194/ms-4-199-2013.

Dominici N, Ivanenko YP, Cappellini G, d'Avella A, Mondì V, Cicchese M, Fabiano A, Silei T, Paolo AD, Giannini C, Poppele RE and Lacquaniti F (2011) Locomotor Primitives in Newborn Babies and Their Development. *Science* 334(6058): 997–999. DOI:10.1126/science.1210617.

Dzeladini F, van den Kieboom J and Ijspeert A (2014) The contribution of a central pattern generator in a reflex-based neuromuscular model. *Frontiers in Human Neuroscience* 8(June): 1–18. DOI:10.3389/fnhum.2014.00371.

Eilenberg MF, Geyer H and Herr H (2010) Control of a powered ankle-foot prosthesis based on a neuromuscular model. *IEEE transactions on neural systems and rehabilitation engineering : a publication of the IEEE Engineering in Medicine and Biology Society* 18(2): 164–173. DOI:10.1109/TNSRE.2009.2039620.

Faisal AA, Selen LPJ and Wolpert DM (2008) Noise in the nervous system. *Nature Reviews Neuroscience* 9(4): 292–303. DOI:10.1038/nrn2258.

Faraji S, Pouya S, Atkeson CG and Ijspeert AJ (2014a) Versatile and robust 3d walking with a simulated humanoid robot (Atlas): A model predictive control approach. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1943–1950. DOI: 10.1109/ICRA.2014.6907116.

Faraji S, Pouya S and Ijspeert A (2014b) Robust and Agile 3d Biped Walking With Steering Capability Using a Footstep Predictive Approach. Robotics: Science and Systems Foundation. DOI:10.15607/RSS.2014.X.028.

Farris DJ and Sawicki GS (2011) The mechanics and energetics of human walking and running: a joint level perspective. *Journal of The Royal Society Interface* : rsif20110182 DOI:10.1098/rsif.2011.0182.

Fisette P and Samin JC (1993) ROBOTRAN: Symbolic Generation of Multi-Body System Dynamic Equations. In: Schiehlen W (ed.) *Advanced Multibody System Dynamics*, number 20 in Solid Mechanics and Its Applications. Springer Netherlands, pp. 373–378. DOI: 10.1007/978-94-017-0625-4_21.

Fitzpatrick P, Harada K, Kemp CC, Matsumoto Y, Yokoi K and Yoshida E (2016) Humanoids. In: Siciliano B and Khatib O (eds.) *Springer Handbook of Robotics*. Springer International Publishing, pp. 1789–1818.

Frigon A (2012) Central pattern generators of the mammalian spinal cord. *The Neuroscientist: A Review Journal Bringing Neurobiology, Neurology and Psychiatry* 18(1): 56–69. DOI: 10.1177/1073858410396101.

Geijtenbeek T, van de Panne M and van der Stappen AF (2013) Flexible Muscle-based Locomotion for Bipedal Creatures. *ACM Trans. Graph.* 32(6): 206:1–206:11. DOI: 10.1145/2508363.2508399.

Geng T, Porr B and Wörgötter F (2005) Fast biped walking with a reflexive controller and real-time policy searching. In: *Advances in Neural Information Processing Systems*, pp. 427–434.

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 : a publication of the IEEE Engineering in Medicine and Biology Society* 18(3): 263–73. DOI:10.1109/TNSRE.2010.2047592.

Geyer H, Seyfarth A and Blickhan R (2003) Positive force feedback in bouncing gaits? *Proceedings. Biological sciences / The Royal Society* 270(1529): 2173–83. DOI:10.1098/rspb.2003.2454.

Hansen N, Müller SD and Koumoutsakos P (2003) Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). *Evolutionary Computation* 11(1): 1–18. DOI:10.1162/106365603321828970.

Hashimoto K, Takezaki Y, Hattori K, Kondo H, Takashima T, Lim Ho and Takanishi A (2010) A study of function of foot's medial longitudinal arch using biped humanoid robot. In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2206–2211. DOI:10.1109/IROS.2010.5650414.

Heremans F, Van der Noot N, Ijspeert AJ and Ronsse R (2016) Bio-inspired balance controller for a humanoid robot. In: *2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, pp. 441–448. DOI:10.1109/BIOROB.2016.7523667.

Hicks JH (1954) The mechanics of the foot. *Journal of Anatomy* 88(Pt 1): 25–30.1.

Hill AV (1938) The Heat of Shortening and the Dynamic Constants of Muscle. *Proceedings of the Royal Society B: Biological Sciences* 126(843): 136–195. DOI:10.1098/rspb.1938.0050.

Hobbelen D, Boer Td and Wisse M (2008) System overview of bipedal robots Flame and TUlip: Tailor-made for Limit Cycle Walking. In: *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2486–2491. DOI:10.1109/IROS.2008.4650728.

Hobbelen DGE and Wisse M (2007) Humanoid Robots, Human-like Machines - Chapter 14: Limit Cycle Walking DOI:10.5772/4808.

Hogan N (1985) Impedance Control: An Approach to Manipulation. *American Control Conference, 1984 IS - SN - VO* - 107: 304–313. DOI:10.1115/1.3140702.

Horak FB and Nashner LM (1986) Central programming of postural movements: adaptation to altered support-surface configurations. *Journal of Neurophysiology* 55(6): 1369–1381.

Hutchinson JR and Gatesy SM (2006) Dinosaur Locomotion: Beyond the bones. *Nature* 440(7082): 292–294. DOI:10.1038/440292a.

Hyon SH, Hale JG and Cheng G (2007) Full-Body Compliant Human;Humanoid Interaction: Balancing in the Presence of Unknown External Forces. *IEEE Transactions on Robotics* 23(5): 884–898. DOI:10.1109/TRO.2007.904896.

Ijspeert AJ (2008) Central pattern generators for locomotion control in animals and robots: A review. *Neural Networks* 21(4): 642–653. DOI:10.1016/j.neunet.2008.03.014.

Ijspeert AJ, Crespi A, Ryczko D and Cabelguen JM (2007) From swimming to walking with a salamander robot driven by a spinal cord model. *Science (New York, N.Y.)* 315(5817): 1416–1420. DOI:10.1126/science.1138353.

Ito K, Ogihara N, Hosoda K, Shimizu M, Kume S, Nagura T, Nakamura T, Imanishi N, Aiso S and Jinzaki M (2014) Direct assessment of foot kinematics during human gait using a dynamic cadaver simulator and a biplane X-ray fluoroscopy. *Journal of Foot and Ankle Research* 7(Suppl 1): A105. DOI:10.1186/1757-1146-7-S1-A105.

Johnson M, Shrewsbury B, Bertrand S, Calvert D, Wu T, Duran D, Stephen D, Mertins N, Carff J, Rifenburgh W, Smith J, Schmidt-Wetekam C, Faconti D, Graber-Tilton A, Eyssette N, Meier T, Kalkov I, Craig T, Payton N, McCrory S, Wiedebach G, Layton B, Neuhaus P and Pratt J (2016) Team IHMC's Lessons Learned from the DARPA Robotics Challenge: Finding Data in the Rubble. *Journal of Field Robotics* DOI:10.1002/rob.21674.

Kaneko K, Kanehiro F, Kajita S, Yokoyama K, Akachi K, Kawasaki T, Ota S and Isozumi T (2002) Design of prototype humanoid robotics platform for HRP. In: *IEEE/RSJ International Conference on Intelligent Robots and System*, volume 3. IEEE, pp. 2431–2436. DOI:10.1109/IRDS.2002.1041632.

Kennedy J and Eberhart R (1995) Particle swarm optimization. In: *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4. IEEE, pp. 1942–1948. DOI:10.1109/ICNN.1995.488968.

Ker RF, Bennett MB, Bibby SR, Kester RC and Alexander RM (1987) The spring in the arch of the human foot. *Nature* 325(7000): 147–149. DOI:10.1038/325147a0.

Kiehn O and Butt SJB (2003) Physiological, anatomical and genetic identification of CPG neurons in the developing mammalian spinal cord. *Progress in Neurobiology* 70(4): 347–361. DOI:10.1016/S0301-0082(03)00091-1.

Kuo AD (2002) The relative roles of feedforward and feedback in the control of rhythmic movements. *Motor Control* 6(2): 129–145.

Kurazume R, Tanaka S, Yamashita M, Hasegawa T and Yoneda K (2005) Straight legged walking of a biped robot. In: *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 337–343. DOI:10.1109/IROS.2005.1545447.

Lacquaniti F, Ivanenko YP and Zago M (2012) Development of human locomotion. *Current Opinion in Neurobiology* 22(5): 822–828. DOI:10.1016/j.conb.2012.03.012.

Li J, Huang Q, Zhang W, Yu Z and Li K (2008) Flexible foot design for a humanoid robot. In: *2008 IEEE International Conference on Automation and Logistics*, pp. 1414–1419. DOI:10.1109/ICAL.2008.4636375.

Li Z, Tsagarakis NG and Caldwell DG (2012) A passivity based admittance control for stabilizing the compliant humanoid COMAN. In: *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, pp. 43–49. DOI:10.1109/HUMANOIDS.2012.6651497.

Li Z, Zhou C, Castano J, Wang X, Negrello F, Tsagarakis NG and Caldwell DG (2015) Fall Prediction of legged robots based on energy state and its implication of balance augmentation: A study on the humanoid. In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5094–5100. DOI:10.1109/ICRA.2015.7139908.

Matsuoka K (1985) Sustained oscillations generated by mutually inhibiting neurons with adaptation. *Biological cybernetics* 52(6): 367–376. DOI:10.1007/BF00449593.

Matsuoka K (1987) Mechanisms of frequency and pattern control in the neural rhythm generators. *Biological Cybernetics* 56(5-6): 345–353. DOI:10.1007/BF00319514.

McCrea DA and Rybak IA (2008) Organization of mammalian locomotor rhythm and pattern generation. *Brain Research Reviews* 57(1): 134–146. DOI:10.1016/j.brainresrev.2007.08.006.

McGeer T (1990) Passive Dynamic Walking. *The International Journal of Robotics Research* 9(2): 62–82. DOI:10.1177/027836499000900206.

Metta G, Sandini G, Vernon D, Natale L and Nori F (2008) The iCub Humanoid Robot: An Open Platform for Research in Embodied Cognition. In: *Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems*, PerMIS '08. New York, NY, USA: ACM, pp. 50–56. DOI:10.1145/1774674.1774683.

Minassian K, Hofstoetter US, Dzeladini F, Guertin PA and Ijspeert A (2017) The Human Central Pattern Generator for Locomotion: Does It Exist and Contribute to Walking? *The Neuroscientist* DOI:10.1177/1073858417699790.

Mordatch I, Wang JM, Todorov E and Koltun V (2013) Animating Human Lower Limbs Using Contact-invariant Optimization. *ACM Trans. Graph.* 32(6): 203:1–203:8. DOI:10.1145/2508363.2508365.

Mosadeghzad M, Medrano-Cerda Ga, Saglia Ja, Tsagarakis NG and Caldwell DG (2012) Comparison of various active impedance control approaches, modeling, implementation, passivity, stability and trade-offs. In: *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM*. IEEE, pp. 342–348. DOI:10.1109/AIM.2012.6265960.

Murray MP, Kory RC, Clarkson BH and Sepic SB (1966) Comparison of free and fast speed walking patterns of normal men. *American journal of physical medicine* 45(1): 8–23.

Ogura Y, Shimomura K, Kondo H, Morishima A, Okubo T, Momoki S, Lim Ho and Takanishi A (2006) Human-like walking with knee stretched, heel-contact and toe-off motion by a humanoid robot. In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3976–3981. DOI:10.1109/IROS.2006.281834.

Patla AE, Adkin A and Ballard T (1999) Online steering: coordination and control of body center of mass, head and body reorientation. *Experimental Brain Research* 129(4): 629–634. DOI:10.1007/s002210050932.

Paul C, Bellotti M, Jezernik S and Curt A (2005) Development of a human neuro-musculoskeletal model for investigation of spinal cord injury. *Biol. Cybern.* 93(3): 153–170. DOI: DOI10.1007/s00422-005-0559-x.

Pratt G and Williamson M (1995) Series elastic actuators. In: *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, volume 1. IEEE Comput. Soc. Press, pp. 399–406. DOI:10.1109/IROS.1995.525827.

Pratt J, Chew CM, 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. DOI:10.1177/02783640122067309.

Rajagopal A, Dembia CL, DeMers MS, Delp DD, Hicks JL and Delp SL (2016) Full-Body Musculoskeletal Model for Muscle-Driven Simulation of Human Gait. *IEEE Transactions on Biomedical Engineering* 63(10): 2068–2079. DOI:10.1109/TBME.2016.2586891.

Razavi H, Bloch AM, Chevallereau C and Grizzle JW (2017) Symmetry in legged locomotion: a new method for designing stable periodic gaits. *Autonomous Robots* 41(5): 1119–1142. DOI:10.1007/s10514-016-9593-x.

Riener R, Rabuffetti M and Frigo C (2002) Stair ascent and descent at different inclinations. *Gait & Posture* 15(1): 32–44.

Robotran (2009) Modeling Multibody Systems with ROBOTRAN. Technical report, Université catholique de Louvain, Louvain-la-Neuve.

Ronsse R, Sternad D and Lefevre P (2009) A computational model for rhythmic and discrete movements in uni- and bimanual coordination. *Neural computation* 21(5): 1335–70. DOI: 10.1162/neco.2008.03-08-720.

Rose-Jacobs R (1983) Development of gait at slow, free, and fast speeds in 3- and 5-year-old children. *Physical Therapy* 63(8): 1251–1259.

Rossignol S, Dubuc R and Gossard JP (2006) Dynamic sensorimotor interactions in locomotion. *Physiological reviews* 86(1): 89–154. DOI:10.1152/physrev.00028.2005.

Saibene F and Minetti AE (2003) Biomechanical and physiological aspects of legged locomotion in humans. *European Journal of Applied Physiology* 88(4-5): 297–316. DOI: 10.1007/s00421-002-0654-9.

Samin JC and Fisette P (2003) *Symbolic Modeling of Multibody Systems*. Number 112 in Solid Mechanics and Its Applications. Springer.

Sardain P and Bessonnet G (2004a) Forces acting on a biped robot. Center of pressure-zero moment point. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 34(5): 630–637. DOI:10.1109/TSMCA.2004.832811.

Sardain P and Bessonnet G (2004b) Zero Moment Point - Measurements From a Human Walker Wearing Robot Feet as Shoes. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 34(5): 638–648. DOI:10.1109/TSMCA.2004.832833.

Schaal S (2007) The New Robotics-towards human-centered machines. *HFSP journal* 1(2): 115–26. DOI:10.2976/1.2748612.

Schepelmann A, Taylor MD and Geyer H (2012) Development of a Testbed for Robotic Neuromuscular Controllers. In: *Robotics: Science and Systems*. MIT Press.

Seo JT and Yi BJ (2009) Modeling and analysis of a biomimetic foot mechanism. In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1472–1477. DOI: 10.1109/IROS.2009.5354153.

Shik ML, Severin FV and Orlovskiĭ GN (1966) Control of walking and running by means of electric stimulation of the midbrain. *Biofizika* 11(4): 659–666.

Shimoda S, Yoshihara Y and Kimura H (2013) Adaptability of Tacit Learning in Bipedal Locomotion. *IEEE Transactions on Autonomous Mental Development* 5(2): 152–161. DOI: 10.1109/TAMD.2013.2248007.

Smith EP and Rose KA (1995) Model goodness-of-fit analysis using regression and related techniques. *Ecological Modelling* 77(1): 49–64. DOI:10.1016/0304-3800(93)E0074-D.

Smith RL (1998) *Intelligent motion control with an artificial cerebellum*. Thesis, ResearchSpace@Auckland.

Smola AJ and Schölkopf B (2004) A tutorial on support vector regression. *Statistics and Computing* 14(3): 199–222. DOI:10.1023/B:STCO.0000035301.49549.88.

Song S and Geyer H (2011) The energetic cost of adaptive feet in walking. In: *2011 IEEE International Conference on Robotics and Biomimetics*, pp. 1597–1602. DOI:10.1109/ROBIO.2011.6181517.

Song S and Geyer H (2012) Regulating speed and generating large speed transitions in a neuromuscular human walking model. In: *2012 IEEE International Conference on Robotics and Automation*. IEEE, pp. 511–516. DOI:10.1109/ICRA.2012.6225307.

Song S and Geyer H (2013) Generalization of a muscle-reflex control model to 3d walking. *Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference* 2013: 7463–6. DOI:10.1109/EMBC.2013.6611284.

Song S and Geyer H (2015a) A neural circuitry that emphasizes spinal feedback generates diverse behaviours of human locomotion. *The Journal of physiology* DOI:10.1113/JP270228.

Song S and Geyer H (2015b) Regulating speed in a neuromuscular human running model. In: *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 217–222. DOI:10.1109/HUMANOIDS.2015.7363554.

Spong MW, Hutchinson S and Vidyasagar M (2005) *Robot Modeling and Control*. Wiley. Google-Books-ID: wGapQAAACAAJ.

Stephens B (2007) Humanoid push recovery. In: *2007 7th IEEE-RAS International Conference on Humanoid Robots*, pp. 589–595. DOI:10.1109/ICHR.2007.4813931.

Stokes VP, Andersson C and Forssberg H (1989) Rotational and translational movement features of the pelvis and thorax during adult human locomotion. *Journal of Biomechanics* 22(1): 43–50.

Taga G (1994) Emergence of bipedal locomotion through entrainment among the neuro-musculo-skeletal system and the environment. *Physica D: Nonlinear Phenomena* 75(1-3): 190–208. DOI:10.1016/0167-2789(94)90283-6.

Taga G, Yamaguchi Y and Shimizu H (1991) Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment. *Biological Cybernetics* 65(3): 147–159. DOI:10.1007/BF00198086.

Ting LH, Chvatal SA, Safavynia SA and Lucas McKay J (2012) Review and perspective: neuromechanical considerations for predicting muscle activation patterns for movement. *International Journal for Numerical Methods in Biomedical Engineering* 28(10): 1003–1014. DOI:10.1002/cnm.2485.

Todorov E (2009) Efficient computation of optimal actions. *Proceedings of the National Academy of Sciences* 106(28): 11478–11483. DOI:10.1073/pnas.0710743106.

Todorov E and Jordan MI (2002) Optimal feedback control as a theory of motor coordination. *Nature Neuroscience* 5(11): 1226–1235. DOI:10.1038/nn963.

Truong TVA, Flavigne D, Pettrée J, Mombaur K and Laumond JP (2010) Reactive synthesizing of human locomotion combining nonholonomic and holonomic behaviors. In: *2010 3rd IEEE RAS EMBS International Conference on Biomedical Robotics and Biomechatronics*, pp. 632–637. DOI:10.1109/BIOROB.2010.5626817.

Tsagarakis N, Laffranchi M, Vanderborght B and Caldwell D (2009) A compact soft actuator unit for small scale human friendly robots. In: *2009 IEEE International Conference on Robotics and Automation*, pp. 4356–4362. DOI:10.1109/ROBOT.2009.5152496.

Tsagarakis NG, Li Z, Saglia J and Caldwell DG (2011) The design of the lower body of the compliant humanoid robot "cCub". In: *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, pp. 2035–2040. DOI:10.1109/ICRA.2011.5980130.

Tsagarakis NG, Morfey S, Medrano Cerda G, Li Z and Caldwell DG (2013) COMpliant huMANoid COMAN: Optimal joint stiffness tuning for modal frequency control. In: *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 673–678. DOI:10.1109/ICRA.2013.6630645.

Van der Noot N and Barrea A (2014) Zero-Moment Point on a bipedal robot under bio-inspired walking control. In: *MELECON 2014 - 2014 17th IEEE Mediterranean Electrotechnical Conference*. IEEE, pp. 85–90. DOI:10.1109/MELCON.2014.6820512.

Van der Noot N, Colasanto L, Barrea A, van den Kieboom J, Ronsse R and Ijspeert AJ (2015a) Experimental validation of a bio-inspired controller for dynamic walking with a humanoid

robot. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 393–400. DOI:10.1109/IROS.2015.7353403.

Van der Noot N, Dzeladini F, Ijspeert AJ and Ronsse R (2014) Simplification of the Hill Muscle Model Computation for Real-Time Walking Controllers with Large Time Steps. In: *Dynamic Walking Conference*. ETH Zurich.

Van der Noot N, Ijspeert AJ and Ronsse R (2015b) Biped gait controller for large speed variations, combining reflexes and a central pattern generator in a neuromuscular model. In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. Seattle, WA, 26-30 May 2015, pp. 6267–6274. DOI:10.1109/ICRA.2015.7140079.

van Jaarsveld HW, Grootenboer HJ, de Vries J and Koopman HF (1990) Stiffness and hysteresis properties of some prosthetic feet. *Prosthetics and Orthotics International* 14(3): 117–124. DOI:10.3109/03093649009080337.

Vukobratovic M and Borovac B (2004) Zero-Moment Point - Thirty five years of its life. *International Journal of Humanoid Robotics* 01(01): 157–173. DOI:10.1142/S0219843604000083.

Wahde M and Pettersson J (2002) A brief review of bipedal robotics research. In: *In Proceedings of the 8th Mechatronics Forum International Conference*, pp. 480–488.

Wang JM, Hamner SR, Delp SL and Koltun V (2012) Optimizing locomotion controllers using biologically-based actuators and objectives. *ACM Trans. Graph* : 25.

Weile DS and Michielssen E (1997) Genetic algorithm optimization applied to electromagnetics: a review. *IEEE Transactions on Antennas and Propagation* 45(3): 343–353. DOI: 10.1109/8.558650.

Winter DA (1995) Anatomy, biomechanics and control of balance during standing and walking. *Waterloo, Canada: Waterloo Biomechanics* .

Wren TAL, Do KP, Rethlefsen SA and Healy B (2006) Cross-correlation as a method for comparing dynamic electromyography signals during gait. *Journal of Biomechanics* 39(14): 2714–2718. DOI:10.1016/j.jbiomech.2005.09.006.

Yamaguchi GT, Sawa AGU, Moran DW, Fessler MJ and Winters JM (1990) A survey of human musculotendon actuator parameters. In: *Multiple Muscle Systems: Biomechanics and Movement Organization*. Springer-Verlag, pp. 717–778.

Yang H, Shuai M, Qiu Z, Wei H and Zheng Q (2008) A novel design of flexible foot system for humanoid robot. In: *2008 IEEE Conference on Robotics, Automation and Mechatronics*, pp. 824–828. DOI:10.1109/RAMECH.2008.4690883.

Yin K, Loken K and van de Panne M (2007) SIMBICON: simple biped locomotion control. *ACM Trans. Graph.* 26(3). DOI:10.1145/1276377.1276509.

Zelik KE and Kuo AD (2010) Human walking isn't all hard work: evidence of soft tissue contributions to energy dissipation and return. *The Journal of Experimental Biology* 213(Pt 24): 4257–4264. DOI:10.1242/jeb.044297.

Zobova AA, Habra T, Van der Noot N, Dallali H, Tsagarakis NG, Fisette P and Ronsse R (2015) Multi-physics modelling of a compliant humanoid robot. In: *Multibody System Dynamics*. Barcelona, pp. 1–20.

Zobova AA, Habra T, Van der Noot N, Dallali H, Tsagarakis NG, Fisette P and Ronsse R (2017) Multi-physics modelling of a compliant humanoid robot. *Multibody System Dynamics* 39(1-2): 95–114. DOI:10.1007/s11044-016-9545-4.

Curriculum Vitae

Nicolas Van der Noot

Electromechanical Engineer
Ph.D. candidate

Born on October 11, 1990
in Brussels

Belgian nationality
Driving license B

Contact
nico.vandernoot@gmail.com

Education

Since 2013 **Ph.D. candidate - F.R.S.-FNRS Aspirant**
Joint Ph.D. thesis between UCL | Université catholique de Louvain and EPFL | École Polytechnique Fédérale de Lausanne.
Two years in Belgium (UCL) and two years in Switzerland (EPFL).

2011-2013 **Master of Science in Engineering**
Electromechanical orientation (mechatronics)
UCL | Université catholique de Louvain.
1st Year - Highest honors, 2nd Year - Highest honors

2008-2011 **Bachelor of Science in Engineering**
Specialization in electricity and mechanics
UCL | Université catholique de Louvain.
1st Year - Highest honors, 2nd Year - Highest honors, 3rd Year - Highest honors

Experience

2013-2017 **Development of the Robotran simulation software**

In parallel to my Ph.D. research activities, I integrated the development team of the Robotran software, a multi-body simulation environment developed within UCL. My main contributions include the following developments: a framework based on *CMake* to configure and run the software on multiple OS, real-time interactions with the simulator, dynamic plotting tools using *SDL* (in real-time) and a 3D visualization of the multi-body system using *OpenGL*.

2013-2017 **Student projects supervision**

On top of the research carried out during my thesis, I supervised the project parts and practical sessions of five different courses (both at UCL and EPFL). I also organized the project of a BEST (Board of European Students of Technology) course. Most of these projects were related to mobile robots. Finally, I supervised four master theses and one semester project.

2012-2013 **Erasmus exchange student in Lausanne**

EPFL | École Polytechnique Fédérale de Lausanne (Switzerland), courses and master thesis (5 months during fall semester).

2011 & 2013 **Tutor in physics**

Tutoring in Physics for UCL students in Bac 1: I was in charge of the exercise sessions of 24 students.

2012 **Eurobot cup**

Participation in the 15^{th} edition of Eurobot, an international amateur robotics contest in a team of 5 students (2^{nd} of Belgium and participation in the European final).

Awards

2016 Second place for the **Best Conference Paper Award** at the 6th IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics, for the paper *Bio-inspired balance controller for a humanoid robot* (second author).

2014 Third place at the **2014 IEEE Region 8 Student Paper Contest**, for the paper *Zero-Moment Point on a Bipedal Robot under Bio-Inspired Walking Control* (first author).

2013 **Best master thesis** in the fields of the Institute of Electrical and Electronics Engineers awarded by the UCLouvain IEEE Student Branch.

2013 Grand Prix - **Prix Pierre Decoux 2013** for the best master thesis awarded by AILouvain (Alumni Ingénieurs Louvain).

2008 **Top of the promotion** (96%) at the admission exam to the Bachelor in Engineering.

Languages

French	Mother language
English	Fluent (spoken & written)
Dutch	Intermediate (spoken & written)

Computer skills

Languages	C/C++, Python, Matlab, Java, html, PHP, CSS, JavaScript, SQL, Verilog, Latex
Applications	Matlab, Altera Quartus II, MG ModelSim

Personal interests

I enjoy playing badminton and tennis, walking, running and photography.

List of publications

Journal paper

Van der Noot N, Ijspeert AJ and Ronsse R (conditionally accepted) **Bio-inspired controller achieving forward speed modulation with a 3D bipedal walker**. International Journal of Robotics Research.

Zobova AA, Habra T, **Van der Noot N**, Dallali H, Tsagarakis NG, Fisette P and Ronsse R (2017) **Multi-physics modelling of a compliant humanoid robot**. Multibody System Dynamics, 39 (1-2), pp. 95-114. DOI: 10.1007/s11044-016-9545-4.

Conference papers

Heremans F, **Van der Noot N**, Ijspeert AJ and Ronsse R (2016) **Bio-inspired balance controller for a humanoid robot**. In: 2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob), Singapore, 26-29 June 2016, pp. 441-448. DOI: 10.1109/BIOROB.2016.7523667.

Colasanto L, **Van der Noot N** and Ijspeert AJ (2015) **Bio-inspired walking for humanoid robots using feet with human-like compliance and neuromuscular control**. In: 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids), Seoul, 3-5 Nov. 2015, pp. 26-32. DOI: 10.1109/HUMANOIDS.2015.7363518.

Van der Noot N, Colasanto L, Barrea A, van den Kieboom J, Ronsse R and Ijspeert AJ (2015) **Experimental validation of a bio-inspired controller for dynamic walking with a humanoid robot**. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Sept. 28 2015-Oct. 2 2015, pp. 393-400. DOI: 10.1109/IROS.2015.7353403.

Zobova AA, Habra T, **Van der Noot N**, Dallali H, Tsagarakis NG, Fisette P and Ronsse R (2015) **Multi-physics modelling of a compliant humanoid robot**. In: ECCOMAS Thematic Conference Multibody Dynamics 2015, Barcelona, 29 June-02 July 2015.

Van der Noot N, Ijspeert AJ and Ronsse R (2015) **Biped gait controller for large speed variations, combining reflexes and a central pattern generator in a neuromuscular model**. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, 26-30 May 2015, pp. 6267-6274. DOI: 10.1109/ICRA.2015.7140079.

Van der Noot N and Barrea A (2014) **Zero-Moment Point on a bipedal robot under bio-inspired walking control**. In: MELECON 2014 - 17th IEEE Mediterranean Electrotechnical Conference, Beirut, 13-16 April 2014, pp. 85-90. DOI: 10.1109/MELCON.2014.6820512.

Poster presentations

Van der Noot N, Ijspeert AJ and Ronsse R (2016) **Neuro-Muscular Controller Based on Reflexes and a Central Pattern Generator to Achieve Gait Modulation**. In: KoroiBot Final Workshop, Heidelberg, 13-14 September 2016.

Van der Noot N, Ijspeert AJ and Ronsse R (2016) **Humanoid Robot Control Recruiting Muscles, Reflexes and a Central Pattern Generator**. In: IEEE-EMB Benelux Chapter and the 14th National Day on Biomedical Engineering, Brussels, 4 March 2016.

Van der Noot N, Colasanto L, Ronsse R and Ijspeert AJ (2015) **Porting Reflex-Based Muscles Control to Real Humanoid Robots**. In: 2015 IEEE International Conference on Robotics and Automation (ICRA) - Workshop on Dynamic Locomotion and Balancing, Seattle, WA, 26 May 2015.

Van der Noot N, Dzeladini F, Ijspeert AJ and Ronsse R (2014) **Simplification of the Hill Muscle Model Computation for Real-Time Walking Controllers with Large Time Steps**. In: Dynamic Walking, Zurich, 10-13 June 2014.