Toward a reactive motion generation on exteroceptive feedback for generalized locomotion of humanoid robots

Kevin Giraud–Esclasse

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ FÉDÉRALE TOULOUSE MIDI-PYRÉNÉES

Délivré par :
l'Institut National des Sciences Appliquées de Toulouse (INSA de Toulouse)

Présentée et soutenue le *18/12/2019* **par :**

Kévin GIRAUD--ESCLASSE

TOWARDS A REACTIVE MOTION GENERATION ON EXTEROCEPTIVE FEEDBACK FOR GENERALIZED LOCOMOTION OF HUMANOID ROBOTS

JURY

Christine CHEVALLEREAU	Directeur de Recherche	Rapporteur
Andrea CHERUBINI	Maitre de conférence	Rapporteur
Guillaume ALLIBERT	Maitre de conférence	Membre du jury
Philippe SOUÈRES	Directeur de Recherche	Président du jury
Olivier STASSE	Directeur de Recherche	Directeur de Thèse

École doctorale et spécialité :
EDSYS : Robotique 4200046
Unité de Recherche :
Laboratoire d'analyse et d'architecture des systèmes
Directeur(s) de Thèse :
Olivier STASSE
Rapporteurs :
Christine CHEVALLEREAU et *Andrea CHERUBINI*

Acknowledgments

First of all I would like to thanks Christine Chevallereau and Andrea Cherubini for reviewing this thesis, as well as Guillaume Allibert and Philippe Souères for being members of the jury. I would also like to thank Olivier Stasse who guided me along these years. I also gratefully acknowledge the other Gepetto researchers who have been part of this powerful team : Bertrand Tondu, Bruno Watier, Florent Lamiraux, Jean-Paul Laumond, Michel Taïx and Nicolas Mansard.

I would like to thank all the external collaborators with whom I worked and specifically Ang Zhang, Edouard Brousse, Francesco Ferro, Guillaume Avrin, Hilario Tome, Ixchel G. Ramirez-Alpizar, Luca Marchionni, Rémi Régnier and Rodolphe Gelin.

A very special gratitude goes out to all down at Gepetto team postdoc members, PhD students, engineers and interns I have met there, specially Alexis, Alexis, Andrea, Aurélie, Carlos, Céline, Christian, Diane, Dinesh, Ewen, Fanny, Florenç, Florian, François, Gabriele, Guilhem, Isabelle, Joseph, Justin, Maximilien, Médéric, Mehdi, Melya, Mylène, Naoko, Nassime, Nirmal, Noëlie, Olivier, Paolo, Pierre, Pierre-Alexandre, Raphaël, Rohan, Sabrina, Steve, Thibaud and Thomas. I am especially grateful to my officemates Amit, Bernard, Daeun, Galo, Greg, Jason, Mathieu and Roméo for having endured my puns. The solidarity among the students was for me an essential source of motivation. this team spirit goes further than Gepetto, I would like to give a special mention to colleagues and friends with whom I have spent epic moments : Alexandre, Axel, Claire, Delphine, Ezio, Manon, Martin and Thomas.

Beyond the LAAS, I also thank my friends for their help, particularly Florian, Guillaume, Jessica S., Mohammed and Sanela. I would like to thank my parents and my family for their unwavering support, they built the foundations of my life and assisted me during all these years.

Finally, special thanks for Jessica who has stayed by my side as a precious flagship reminding me of the direction to contemplate.

Contents

This thesis manuscript is constructed in order to reach all audiences, with elements adapted for all levels of knowledge in the robotic field. For that reason, it begins with personal considerations on our perception of robotics with some definitions allowing to grasp the most important concepts. Then Chapters 1 and 2 give essential details for understanding the following Chapters, more related to my contributions.

Robots, more than ever before, seem to stimulate the public imagination. There are two main reasons behind this statement: first, the gradual arrival of robots in our everyday life; and secondly, the mental picture reflected by the genre of Science-Fiction, in literature or in movies, dealing with the marvels and risks of advances in the robotic fields.

This foreword starts with personal points of view on robots, and specifically on the strong bias inherited from literature related to our perception of a robot. Then the definition of a robot is recalled. This helps to distinguish the different field a roboticists should be aware of. Finally, I propose some insight about the different robot structures and their characteristics, which prompted our team to study humanoid robots specifically.

0.1 Where are the robots from ?

The first author to have used the word "robot" in literature is the Czech writer Karel Čapek in his play *R. U. R.* (*Rossum's Universal Robots*) from 1920. Doing so, he gives a new signification of the Slavic word meaning previously *work/duty/labor/chore*. In his play, robots initially created to work as factory workers, rebel and decide to exterminate humankind. For a start of this new word, we find here the theme of the sorcerer's apprentice whose creation turns on him. This theme is not new in literature, it appears for instance in both *Frankenstein; or, The Modern Prometheus* of Mary Shelley, published en 1818, or in Jewish folklore with the tale of the "Golem of Prague" where the golem, created entirely from clay, becomes out of control and must be destroyed.

Fortunately, literature does not only present animated humanoid creations as uncontrollable creatures. We can cite Pinocchio, a wooden puppet which comes to life thanks to its creator Geppetto, and then achieves his dream of becoming a real boy. Here we see clearly the idea of the creature wishing to become human. Another example is *The Sandman*, a short story from Theodor Amadeus Hoffmann, published in 1816: a young man falls in love with a woman, who is in reality an automaton (the term "android" appears only in 1886 in *The future Eve* from Auguste de Villiers de L'Isle-Adam). This work may have inspired Sigmund Freud (in 1919) then Masahiro Mori (in 1970), and lead to the "uncanny valley" concept: when a machine looks almost like a human being but not perfectly, it provokes strange feelings of revulsion in humans. Finally we can cite the myth of Talos, a bronze giant created by Hephaistos who was in charge of the Crete island protection. This conveys the image of an animated creation meant to be a warrior under humans

control to protect an area.

In 1920 when Karel Čapek published his novel, the world was ready for a new word to describe this emerging concept of an autonomous machine dedicated to automating tasks. The term "robot" has then become a part of the more modern literature and movies.

Another writer has broken this legacy of concepts in 1950 with his short stories collection *I, Robot*. Issac Asimov has developed ideas around potential laws of robotics and their flaws in specific situations. This approach seems less pessimistic and technophobe than the previous and predominant concept of the "sorcerer's apprentice", but it is still ambiguous due to the flaws encountered in the robotic laws he defined. A relevant example in more recent movies is *I,robot* from Alex Proyas (title of course correlated with Isaac Asimov work) depicting the "zero law" flaw about humanity protection by an artificial intelligence that is finally stopped by a robot not enforced to follow this law. Other modern blockbusters possess this same suspicious atmosphere about robots and their risks to humanity.

This concludes the recall of a few elements highlighting the bias we could be subject to due to our ambient culture where robots often appear powerful and problematic. In a time when European discussions emerge about a robotic personality or killer robots, I think that, more than ever, we should have clear ideas on what is and is not a robot. Moreover, the following definition will help differentiating between robotics and artificial intelligence, which are often assimilated together.

0.2 Robotics

0.2.1 Robot definition

A robot is a machine.

- It is able to move parts of its structure. That implies it contains actuators like motors.
- It contains computational system organ. It could be for instance a computer or a micro-controller.
- Finally it has a feed back loop from sensors in order to control its motion after computation in the computational system organ.

Roboticists are then involved in many engineering fields as mechanics for the structure, electronics for the power and the sensors, informatics and automation for managing the feedback loops. All the organs of a robot are directly or indirectly interdependent. It constitutes a system that is in charge of realizing a task. If one organ does not fit in the global design of the system, the task will not be achieved.

The computational system of a robot runs decisional processes or algorithms as they were designed by the persons in charge of the feedback loop management, in the framework of different models of the robot, and the environment. These algorithms can have a large range of complexity as it can be a simple subtraction between an

infrared signal and a reference to control the orientation of a line follower robot, or, as in the humanoid robots field, a cascade of optimization problems to find a feasible locomotion trajectory. The artificial intelligence takes place in the algorithms. Even it could be a part of the decisional process of a robot, a dedicated application of an algorithm can stand alone and work in a variety of fields that are not specifically related to robotics.

0.2.2 To humanoid robots

The robot is moving to act on its environment. Two main expectations of a robot are the navigation and the manipulation. Both imply a motion, but these motion generations live in different paradigms. The first will face to non-holonomic constraints (a car cannot go directly on the side way for instance), to path discovery or position and mapping estimation problem. The second, mostly in the case of serialized kinematics chain robots, deals with translating the end effector desired motion into commands to send to each motor of its often redundant serial chain of joints.

Humanoid robots combine these two paradigms with in addition constraints like balance or underactuation. They gather most of the main open problems in robotics. But they also promise a huge variability in the tasks they can fulfill. Because of their anthropomorphic shape, they are meant to evolve in a context structured for humans. When other robotics shapes need adjustments of the environment, humanoid robots integration in human context environment should be non invasive.

Even though we can see on the internet impressive videos showing humanoids robots capable of strong potential maneuvers, the safety and the repeatability issues are not solved yet. The applications of humanoid robots are still marginal. As a use case, the Gepetto team is working on drilling tasks in narrow environments with an aircraft company. This use should decrease the musculoskeletal disorders of the employees working all day long in non-healthy postures. It is a prospective work and no immediate application is considered. Furthermore, this type of robots has an operational space much larger than a dedicated fixed robot since it can move from one place to another using installations made for humans. Since just a robot can be used instead of many dedicated fixed robots, it can be seen as more economical in an industrial point of view.

This example is just a use case, societal impact of such a technology can't be known. Because its future usage is unpredictable, agnosticism may be consider, with the hope that researchers explanations and dissemination will imply more marvels than risks from the citizens choices.

During my PhD studies, I participated in the evaluation of our algorithms repeatability, and in the integration of several tools to increase the locomotion capabilities of humanoid robots. I have worked on the three humanoid platforms of the Gepetto team, on different subjects, but always trying to transfer the results from simulation to the real robots. This makes my thesis more experimental and

focused on integration.

Chapter 1

Scientific introduction

Contents

The expectations for a humanoid robot stand in its capability to navigate into structured and unstructured environments, potentially using several limbs or parts of its body for multiple contacts. It should also fulfill tasks, for instance drilling or screwing with its upper body. These expectations imply to solve a problem where the positions of the robot are computed but also the relevant parts of the environment that may not be known in advance. We will see that these elements generate a huge amount of data. This makes the motion generation for humanoid robots a difficult numerical problem. Moreover it comes with constraints that can be modelized like joint limits or contact with the environment and others that are

more practical like the computation time. The complexity of the problem is related to the robot, the environment and the data we are trying to generate. This chapter sums up the different techniques used to tackle this complexity issue. It highlights the main concepts and ideas that lead to this thesis topic : after explaining why the locomotion problem is complex, it is shown how it can be decoupled to make it more numerically tractable and where my contributions take place.

1.1 Complexity

As previously mentioned, we want to make the humanoid robot navigate in environments and manipulate objects. These tasks are considered as behaviors. A behavior realization can be formalized as follows [Stasse 2013]. Let us consider a robot of n degrees of freedom q, and a vector of external (environment) information v of dimension m. Then we consider a function $f(q, v, t) : \mathbb{R}^{n\times m+1} \rightarrow [0, 1]$ such that it is equal to 0 when the behavior is realized. This amounts to find $q(t)$ in such a formulation :

$$\min f(q(t), v(t)) \tag{1.1}$$

$$b(q((t), v(t), t) \leq 0 \tag{1.2}$$

$$\phi(q((t), v(t), t) = 0 \tag{1.3}$$

with b inequality constraints and ϕ equality constraints. Commonly $\hat{f}$, an approximation of f, is built from a model, while $\hat{q}(t)$ and $\hat{v}(t)$ which are the estimations of $q(t)$ and $v(t)$ are used. In the context of this thesis we consider that $\hat{v}$ is given by an external process (extracting features software using cameras or motion capture system for instance).

This problem is complex and hard to solve. The reasons of this complexity are exposed in this section. It introduces the need to decouple this problem in smaller ones to make it tractable. This decoupling is the framework of my thesis and has to be exposed in order to point out where this thesis takes place.

1.1.1 Space dimension

For generating a locomotion behavior, we must know which variables will be computed and, if possible, their number. The Fig. 1.1 depicts the types of variables that need to be computed over a discretized preview time horizon (required to take into account the dynamics of the motion we want to produce). For clarity we expose here only the state variables (see subsection 1.3.1 for other variable types) which are the main contributor of the curse of dimensionality. We can divide the number of state variables in several parts. The first one is composed of values that describe the robot state: joints and position of the base link in a world frame. The second is related to the environment.

Here we present values given as examples, some of them imply implicit assumptions that are pointed out later (see subsection 1.3). The robot Talos has in our case 32

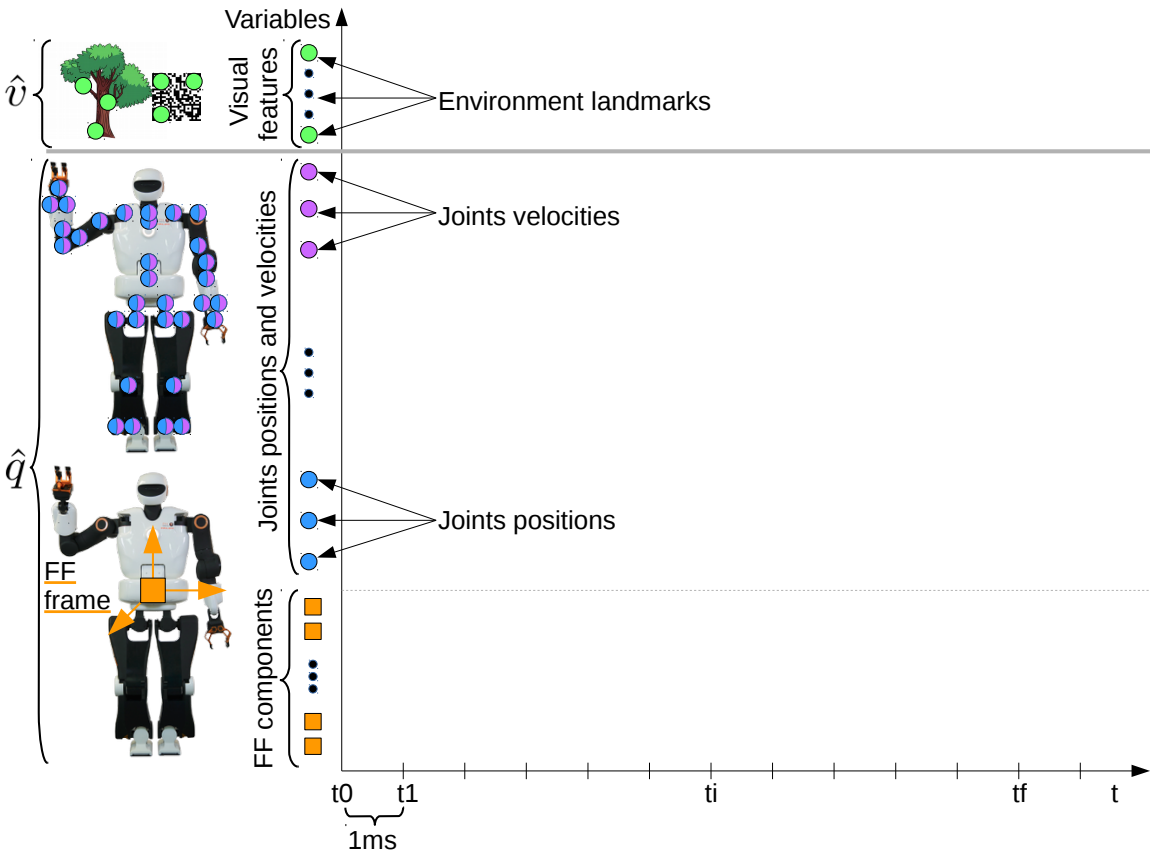


Figure 1.1: State variables in optimization problem.

actuated joints; 6 more degrees of freedom need to be taken into account for the base (often the waist of the robot, called Free Flyer (FF), represented in orange squares on Fig 1.1). The position (blue circles on Fig 1.1) and the velocity of these joints (purple circles on Fig 1.1) usually compose the state of the robot and they must be computed for each time step of the preview horizon. Thus we have at least 77 variables (32 actuated + 6 of the floating base for the positions, the same $32 + 6$ for the velocities, and one more for base position representation in quaternion) to be multiplied by the number of time steps over a time horizon that represents the future of the robot. Usually, we take at least 1.2 seconds for the time horizon that corresponds to 2 steps of the robot on flat ground. To control the robot and have good experimental results, the discretization frequency should be at least $1kHz$ (corresponding to the $1ms$ period between two time steps on Fig 1.1). That represents 1200 time steps over the horizon. This means that the state represents $1200 * 77 = 92400$ variables. The output control variables of this problem are the torque commands, which are sent to the motors, they represent $1200 * 32 = 38400$ variables. Although the constraints on linear dependency of the derivatives of the state variables help breaking down the complexity of the problem, this amount

remains a huge quantity of variables to compute.

These quantities are not fixed since we can decide to solve a complete motion of several seconds or augment the state by letting the solver find the foot prints as in [Herdt 2010a]. The size of the problem to be solved partly explains the difficulty in solving this problem numerically.
The second part, related to the environment, acts differently. On the one hand, the environment model is used to generate the control and the motion of the robot. It acts as constraints (collisions avoidance, contacts) or as tasks (expressed as an error to be reduced in the cost function, see subsection 1.3.7 and 5.2.4) in the problem of locomotion. On the other hand the environment model needs to be estimated. In that case, the visual features (green circles in Fig 1.1) can be used to compose the state variable set to be computed. The dimension of this set may not be known in advance. In my thesis context, this part of the problem is considered as already known. It is considered as an input of the work presented here. We note that the estimation of the environment variables and their use to generate a motion as constraints or tasks are run iteratively: estimated first to update the model that will be used to create the motion. As explained in Chapter 5, we assume that the model of the environment and the number of features are known. Doing this, we reduce the complexity of the problem focusing only on the motion generation ($\hat{v}$ is then removed of 1.1).

1.1.2 Constraints and computation time

In order to be feasible, the state space of the robot needs to evolve in a feasible set. This feasible set is represented by constraints which are the dynamics of the robot, the contacts with the world, the collision avoidance with the environment and itself, and finally the behavior to realize. Unfortunately these constraints are often non linear (configuration space to SE(3), configuration to body-body distance to name a few), and even discrete (making contact or not). Solving such problems at $1kHz$ is extremely challenging and calls for techniques used in optimization to make them tractable. Moreover, not only the problem should be solved in reasonable time interval (minutes or hours) but the main goal is to solve it in real time or on the fly. Solving problem constraints could increase drastically the complexity. Considering collision with the environment or auto-collision is time consuming and often not embedded to run on the fly. Moreover, if random shooting exploration of the feasible set is necessary, the needed computation time is generally not compatible with online running requirements. In [Perrin 2011], paragraph 3.4.2, a sample is treated in 0.4 s which means that for a 6 dimensional space if we want 10 samples per direction, we need 1000000 tests. This means 400000 s which is equivalent to 4 days.
All these points lead roboticists to imagine strategies to speed up calculation like assuming approximations or trying to decouple the problem in simpler sub-problems that could be solved sometimes with really specific methods according to the sub-problem formulation.

1.2 Motion generation decomposition

Since the locomotion problem is complex and long to solve, it is often decoupled in simpler sub-problems. This decomposition is expected to enable a faster resolution. We describe here the different sub-problem blocks as decomposed by [Carpentier 2016]. This is just one way to decompose the problem, some blocks can be replaced, removed or combined. Then, we explain what were the elements leading to the main topic of this thesis. For clarity reasons, pictures from Fig 1.2 are added in the margins in order to recall the output/input between two consecutive blocks/algorithms. A margin picture is then the output of the previous described block and the input of the following one. Each picture is referred in the corresponding block. Moreover, since the dimensionality is one of the main reason for splitting the whole problem in smaller subproblems, figures representing the dimension of the computation of each block is given, when possible, based on Fig 1.1.

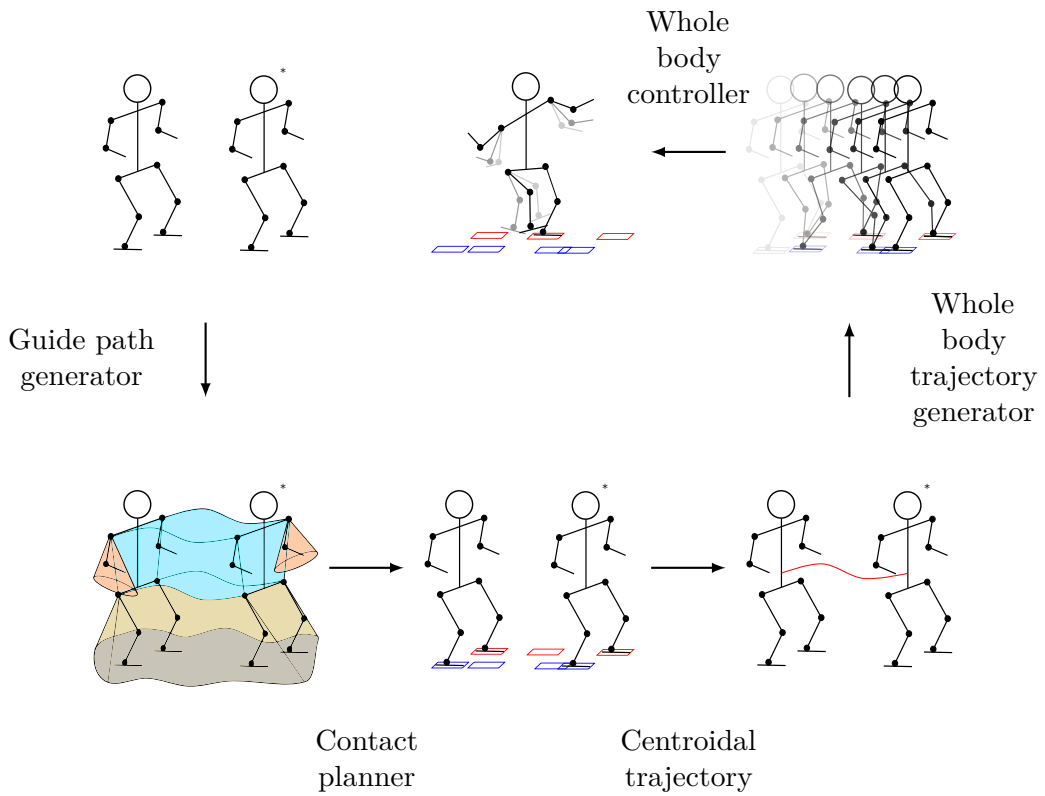
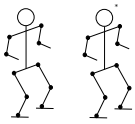


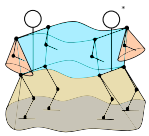
Figure 1.2: Decomposition workflow to solve locomotion problem. Each picture represents internal output between two algorithm blocks

Fig 1.2 represents the different sub-problems of the locomotion problem as it is handled in the Gepetto team [Carpentier 2016]. I briefly introduce these different blocks (guide path generator, contact planner, centroidal trajectory generator, whole body trajectory generator and controller) to describe this thesis context.



1.2.1 Guide path generator

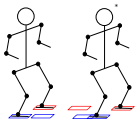
The goal of the first block of Fig. 1.2 is to plan a trajectory so that the robot can go from the starting configuration to the goal one (respectively the left and right robots of the first picture in Fig. 1.2) without any collision of the trunk (blue part of the second picture in Fig 1.2) while maintaining contact with the environment using its limbs (grey and brown for legs, red for the arms in the second picture of Fig. 1.2). It produces a rough guide trajectory for the root of the robot. The method RB-RRT (Reachability Based - Rapidly exploring Random Tree) was first proposed in [Tonneau 2018] and then extended to a kinodynamic version in [Fernbach 2017]. This method plans a trajectory for the center of a simplified model of the robot, using an heuristic based on the reachability space of each limb. This block breaks a part of combinatorial complexity of the next block by limitating the search space.



1.2.2 Contact planner

Once this guide trajectory is found the second block of the framework needs to find a sequence of feasible contacts. The contact generation method presented in [Tonneau 2018] produces a sequence of whole body configurations in contact, so that there is only one contact change between each adjacent configuration. This method generates contact candidates using a set of configuration candidates built offline. It is able to consider each limb separately allowing fast exploration. The reachable space of each limb is intersected with the environment (blue and red foot prints of third picture in Fig. 1.2) while the origin of the robot follows the guide computed previously in order to find configuration candidates close to the contact. Then, the whole body configurations in contact are found by inverse kinematics projection from these candidates.

For every adjacent generated configuration in contact, the algorithm has to check if there exists a motion that connects both of these configurations. This is decided by solving a problem which is a convex reformulation of the multi-contact centroidal dynamic trajectory generation problem [Fernbach 2018].



1.2.3 Centroidal trajectory

The centroidal trajectory is generated with the method proposed in [Ponton 2018]. This method takes as input the sequence of contacts and produces a centroidal (Center of Mass -CoM-) trajectory satisfying the centroidal dynamic constraints (linear and angular momentum) for the given contact points and minimising a tailored cost function. This method can generate centroidal trajectory (red path of the fourth picture on Fig. 1.2) for multi-contact scenarii in real-time thanks to a convex relaxation of the problem. That is a sub-part computation of the complete problem as shown in Fig 1.3.

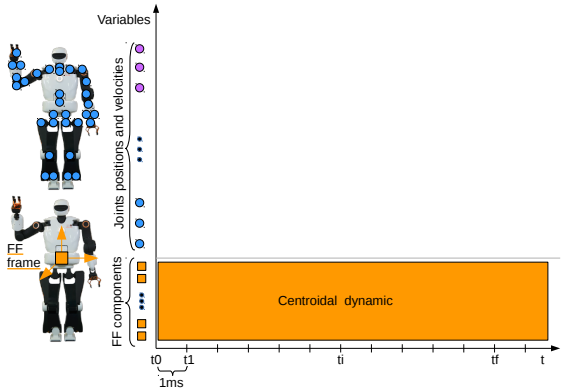
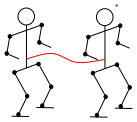


Figure 1.3: A sub part of the full problem is computed : the centroidal dynamic i.e. the trajectory of the center of mass.



1.2.4 Whole body trajectory generator

From the steps placements and CoM trajectory, the whole body trajectory generator is in charge of generating the joints trajectories (depicted with the walking robot of the fifth picture in Fig. 1.2 and in Fig. 1.4). This could be solved by Inverse Kinematics or Operational-Space Inverse Dynamics (IK/OSID) which are often formulated as an optimal control problem. In the Gepetto framework, the previous block, the centroidal trajectory generator 1.2.3, returns also the end effector trajectories as an output. But implementations related to Pattern Generators (PG), as [Naveau 2017], expect to be fed with other input types as CoM velocities. In that case the Whole Body Trajectory Generator (WBTG), here the PG itself, should handle obstacles avoidance based on the model of the environment or exteroceptive sensors loops (see for instance [Stasse 2008]). [Naveau 2017] handles this partly by taking into account 2D flat ground obstacles from the environment model. Taking exteroceptive sensors feedback in the WBTG is an open subject.

Moreover local controller often falls into a local minima. Solving the problem can be done at the WBTG level with a Model Predictive Control (MPC) style as it was done in [Naveau 2017] for walking, or using a trajectory in the task space as in [Toussaint 2007]. Minimizing over a trajectory gives more flexibility but the two above-mentioned papers are limited to a specific task space. Using a Differential Dynamic Program (DDP) is providing a more general framework. However it used to be computationally expensive. The Gepetto group is developing a DDP tailored specifically to perform DDP on a whole body model.

My thesis enters mainly into this block. We evaluated the use of a very efficient DDP algorithm to replace IK/OSID traditionally used in chapter 5. It also deals comparisons of PGs (third and fourth blocks) in chapter 3 and validations of CoM trajectories with WBTG in chapter 4.

Finally, generated trajectories are tested and rejected if collisions are detected.

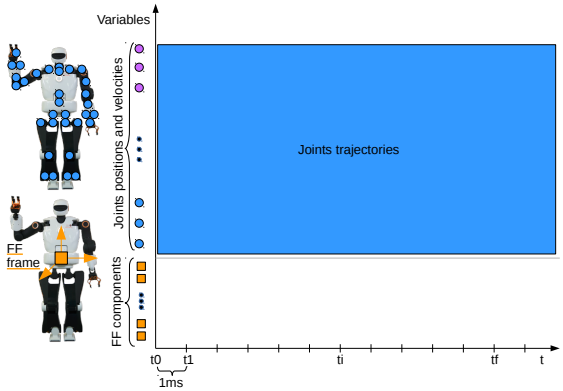


Figure 1.4: A sub part of the full problem is computed : the joints trajectories

1.2.5 Whole body controller

This block is using the same mathematical tools than the whole body trajectory generator, the main difference is that it takes into account sensor feedback and generates the commands to the low level joint controllers. For this reason several implementations are possible OpenSoT, SoT, mc_rtc, or the IHMC whole body controller. Depending on the sensors available (currents, torques, forces, positions), there exist multiple types of controllers that can be used (in position, admittance, impedance, torque ...). After getting the sensors data, it builds the commands so that they are homogeneous with the low level expectations (position or current references). In the case of HRP-2 and more recently of Pyrène, this block (the Stack of Tasks) is capable to handle tasks control since it was designed to generate local motion with high level inputs such as $SE(3)$ tasks. Generally this block computes one Inverse Kinematics for the next time step from the given tasks. It operates an instantaneous control which corresponds to the Fig 1.5.

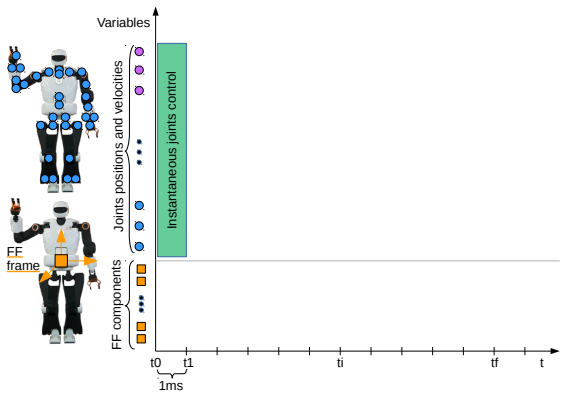
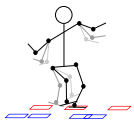


Figure 1.5: Inverse Kinematic on one time step

This block should include a stabilizer (see for instance [Caron 2019]). It is represented by the balancing robot of the sixth picture of Fig 1.2. It tries to correct motions with sensor data (force sensors and inertial measurement unit). The coplanar stepping locomotion is not possible without this correction. For HRP-2, the stabilizer is a black box provided by Kawada Robotics. For Pyrène, PAL Robotics also provided one but during this thesis, I used the ongoing work of the Gepetto team based on [Caron 2019].



1.2.6 Low level controller

The low level controller acts to command the actuator around the reference position. For our robots we cannot manage what this controller does. It is a black box that is assumed to control perfectly the actuators. In practice it is often composed of PIDs feedback controls on current, velocity and position with high gains and sometimes feedforward terms. To be complete, these controllers send commands to a Pulse Width Modulation (PWM) component in charge of powering the motor and making it move accordingly. All the sensor data available are returned to the Wole Body Controller abstract layer.

1.2.7 Conclusion

This decomposition reveals two main approaches. The planning part discovers a path and a contact sequence (no optimality criterion here). This could be seen as global way to handle the problem, but it needs to be validated by methods that are used in the second approach. This last includes the controllers that act on the robot to make it go in basins of attraction ensuring constraints like balance and contact for instance. These ones are local. That means they are not able to discover a whole solution for the locomotion problem, either because of the time horizon they use or because they can only handle a part of the state variables, of the constraints or of the environment. That means they can reach a local minima really far from the objectives. In that case they are unable to get out from that minima and the solution is not satisfying in a human point of view. So they need a global approach method to find a path to follow in order to reach the objective. On purpose I do not place the blocks of Fig. 1.2 in these two approaches because the limit can differ according to the algorithms used to encapsulate these blocks as seen in the next section. More generally the more the controllers will handle, the simpler the planning could be. For instance, [Naveau 2017]'s algorithm described in subsection 1.3.6 computes the foot-prints, letting the planning provide only a Center of Mass velocity as input. Finally, one must notice that no sensor loop has been pointed out in this section. Actually, sensor loops are implemented in the low level controllers and stabilizer. They take respectively currents/encoders and forces/IMU signals as feed-back. A main idea of this thesis is to implement such a feed-back for a higher lever with exteroceptive sensors like cameras or motion-capture system. This leads to contributions of section 3.2.2 and chapter 5.

1.3 Related works

This section describes the evolution of related works, specifically Pattern Generators (PG) and visual servoing. One of my contributions, described in chapter 3, refers and compares certain PG presented in this section. Moreover, visual servoing is presented at the end of this section in order to explain how it is intertwined with PG formulations.

The PG are often a combination of different blocks previously exposed in 1.2. Depending on the implementation, they can contain a Whole Body Trajectory Generator or Controller, a centroidal trajectory generator and sometimes a contact planner.

An important attention is needed on the real time computation requirement (in the sense that the motion should be computed on the robot in a reactive manner). This often implies to know whether a preview window is used to predict the motion and what is its duration. It is also related to the number of variables taken into account to compute the motion.

1.3.1 Optimal control

We have to give here some pieces and vocabulary from optimal control theory to make the following paragraphs clearer. Optimal control allows to find a control input for a given system such that an optimal criterion is achieved, i.e. the cost function is minimized observing the constraints, if any. The optimal control problem is formulated as follows :

$$\text{State variable:} \quad x(t) \tag{1.4}$$

$$\text{Control variable:} \quad u(t) \tag{1.5}$$

$$\text{Cost function:} \quad J = \Phi(x(t_0), t_0, x(t_f), t_f) + \int_{t_0}^{t_f} \mathcal{L}(x(t), u(t), t)dt \tag{1.6}$$

$$\text{State equation:} \quad \dot{x} = a(x(t), u(t), t) \tag{1.7}$$

$$\text{Equality constraints:} \quad \phi(x(t), u(t), t) = 0 \tag{1.8}$$

$$\text{Inequality constraints:} \quad b(x(t), u(t), t) \leq 0 \tag{1.9}$$

Depending on the form and regularity (time invariant, linear with respect to state and control variables) of the state and constraints functions, the problem can be either solved analytically or numerically.

In our context, numerical methods prevail and are often used to solve the formulations expressed in the following subsections dealing with Pattern Generators (PG).

1.3.2 Cart-table model

[Kajita 2003a] uses inverted pendulum model with horizontal plane constraint (the CoM altitude is considered constant). This leads to the following cart-table model

equations :

$$\ddot{y} = \frac{g}{z_c}(y - p_y) \tag{1.10}$$

$$\ddot{x} = \frac{g}{z_c}(x - p_x) \tag{1.11}$$

and the counterpart Zero Moment Point (ZMP) $p = [p_x, p_y, p_z]^T$ equations :

$$p_y = y - \frac{z_c}{g}\ddot{y} \tag{1.12}$$

$$p_x = x - \frac{z_c}{g}\ddot{x} \tag{1.13}$$

where the center of mass (CoM) position is $[x, y, z_c]^T$, g is the gravity acceleration and z_c the horizontal plane coordinate. Equation 1.13 is used to construct an optimal control problem on a preview window of size N_l. Formulation of this problem considers $\begin{bmatrix} x & \dot{x} & \ddot{x} \end{bmatrix}^T$ as the state variable (only x dimension is presented here, y dimension gives exactly the same form from eq. 1.12). In a discrete manner, state equation is formulated as follows:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ p(k) &= Cx(k) \end{aligned} \tag{1.14}$$

Where u is the jerk $\dddot{x}$ considered here as the control variable, p is the position of the ZMP to be controlled by adding a term in the cost function. A and B matrices are derivative operators in discrete time (T is the constant sample duration) given by:

$$A = \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \tag{1.15}$$

$$B = \begin{bmatrix} T^3/6 & T^2/2 & T \end{bmatrix}^T \tag{1.16}$$

Matrix C translates the ZMP equation 1.13:

$$C = [1 \quad 0 \quad -z_c/g] \tag{1.17}$$

We emphasize here that the formulated problem has a state function linear in state and control. Moreover the cost is formulated as a quadratic cost of the errors on ZMP, references and control variables. That means the problem is formulated as a Linear Quadratic Regulator (LQR) without any constraint:

$$\min_{\dddot{X}_k} \frac{1}{2}Q(P_{k+1} - P_{k+1}^{ref})^2 + \frac{1}{2}R\dddot{X}_k^2 \tag{1.18}$$

Where P^{ref} is the reference for the ZMP, here it corresponds to the foot placements given as input, $\dddot{X}_k = \begin{bmatrix} \dddot{x}_k & \cdots & \dddot{x}_{k+N_t-1} \end{bmatrix}$, $P_{k+1} = \begin{bmatrix} p_{k+1} & \cdots & p_{k+N_t} \end{bmatrix}$, Q and R

are weighting matrices. Terms to be minimized constitute the cost function of this problem, it refers to minimizing J in Eq. 1.6 . LQR lies into Optimal Control theory and can be solved by Riccati equations giving an analytical solution of the control variable $u(k)$. This is very convenient for having a high computation speed. Moreover this work gives clues to choose the correct size of the preview window. It compares the gains of the control variables given by optimal controller on preview action term. As shown in [Kajita 2003a], in this case, the 2 first seconds of preview have the more important effect on the control. This magnitude order will remain in the following described works.
[Kajita 2003a]'s authors evoke a refining of the solution by taking into account the multi-body dynamics to compute the ZMP. That can be seen as a Dynamic Filter (DF, also referred in subsection 1.3.6).
We recall that this algorithm takes footprints as input and generates CoM trajectory on time horizon as an output. That corresponds to the "centroidal trajectory block" in Fig 1.2. The angular momentum of the center of mass is set to zero.

In light of the trade-off made between the computation speed and the model precision, one can say that the ZMP equations are reducing too much the robot dynamics. Other ways to control the gait exist as Hybrid Zero Dynamics developed in [Westervelt 2018] or the use of a sequence of Quadratic Constraints Quadratic Program (QCQPs) for solving a 3D locomotion with multiple non-coplanar contacts as in [Perrin 2018]. These approaches will not be detailed further in my thesis.

1.3.3 Inequalities constraints addition

[Wieber 2006] brings a new feature to manage strong perturbations during the walk. Instead of expressing the problem as LQR formulation, inequality constraints (see Eq. 1.9) are added and make the problem a QP (Quadratic Problem). Indeed the QP formulation can handle linear constraints. The constraints are given to bound the ZMP. This means that the ZMP is free to move under the feet into that limits, but the foot prints remain an input of this algorithm. The problem is now formulated as follows :

$$\min_{\dddot{X}_k} \frac{1}{2}\dddot{X}_k^2 \tag{1.19}$$

$$s.t. \quad P_k^{min} \leq P_k \leq P_k^{max} \tag{1.20}$$

This form (with constraints) does not have any analytical solution and can be longer to compute. But putting wisely the bounds P_k^{min} and P_k^{max} under the polygon support of the feet with an appropriate margin improves greatly the robustness against external perturbations such as forces applied on torso when walking.
Both inputs and outputs of these methods are the same as the previous ones but the resolution and implementation are different to allow online computation for the whole horizon and taking into account ZMP boundaries constraints. The feet have imposed placements that can't be changed (reference is fixed). The z coordinate of

the CoM could be changed in this method.

1.3.4 4th order polynomial model

[Morisawa 2007] proposes to express the CoM and ZMP (linked by ZMP equations 1.13) trajectories as polynomials of order N (tested with $N = 4$). Analytical solutions come with *cosh* and *sinh* functions. Walk phases (simple and double support) are calculated separately but this formulation allows to connect them efficiently. The goal of this work was to be able to modify the placement of the next landing foot (new reference given before taking off) online. To reach online behaviour, the solution is calculated using a pseudo-inverse to find the coefficients of the polynomials used to describe the trajectories. Polynomials are set to express the two next steps, shifting the time horizon at each landing so that we always have two steps ahead.
If a too important change for the foot placement occurs, the polynomials of ZMP can have huge fluctuations. The maxima of this trajectory can overpass the bound limits of the support convex hull. If such a point appears, the landing time is used to regulate the ZMP and to make it stay in the bounds. This really slows down the motion and conflicts can appear if the time landing correction is different from x and y axis.
This method is really fast in computation time but doesn't embed the complete dynamic and doesn't allow to modify foot placements as expected (small modifications, slows the motion, creates conflicts).

1.3.5 Foot prints as free variables

The main contribution of [Herdt 2010a] is to let the system find its foot placements and to formulate the problem such that feet positions become free variables. The input becomes the velocity for the CoM. The problem is formulated as follows :

$$\begin{aligned}\min_{\dddot{C}_i^\alpha, F_{i+1}^\alpha} & \frac{\zeta}{2}\left\|\dot{C}_{i+1}^x - \dot{C}_{ref}^x\right\|^2 + \frac{\zeta}{2}\left\|\dot{C}_{i+1}^y - \dot{C}_{ref}^y\right\|^2 \\ & + \frac{\beta}{2}\left\|EC_{i+1}^x - \dot{c}_{ref}^x\right\|^2 + \frac{\beta}{2}\left\|EC_{i+1}^y - \dot{c}_{ref}^y\right\|^2 \\ & + \frac{\gamma}{2}\left\|F_{i+1}^x - Z_{i+1}^x\right\|^2 + \frac{\gamma}{2}\left\|F_{i+1}^y - Z_{i+1}^y\right\|^2 \\ & + \frac{\varepsilon}{2}\left\|\dddot{C}_i^x\right\|^2 + \frac{\varepsilon}{2}\left\|\dddot{C}_i^y\right\|^2 \end{aligned} \tag{1.21}$$

Where C terms are referring to the center of mass. Let us call α either x or y indexes, then $\dot{C}_{i+1}^\alpha = \begin{bmatrix}\dot{c}_{i+1}^\alpha & \cdots & \dot{c}_{i+N_t}^\alpha\end{bmatrix}^T$. The only inputs $\dot{C}_{ref}^\alpha$ are the velocity references for the center of mass, which are constructed on the same scheme as $\dot{C}_{i+1}^\alpha$. The product EC_{i+1}^α is the average velocity built from CoM position and an ad hoc average derivative matrix E. The last line of equation 1.21 regulates Jerk of the center of CoM which is a part of the control variable, it is constructed on the same scheme as $\dot{C}_{i+1}^\alpha$ In the third line lies the freedom of the foot placement. Z_{i+1}^α is the ZMP position (over the horizon) calculated from ZMP equations 1.13 with

CoM as input (which is know by integration of the reference). F^x_{i+1} is the vector over time horizon that contains foot placements. It is composed from selection matrices expressing the time steps when the feet should be on the ground. The QP formulation allows to find the steps together with both ZMP and CoM trajectories. Moreover it also manages the limits the foot is able to reach in position by inequality constraints. As explained in [Stasse 2009], this is possible by creating a polygon of reachability from random shooting of the positions of the foot on the ground, then validated to be kinematically achievable and out of collision.

The main drawback of formulating the problem as a single QP is that it cannot handle non-linear constraints or cost functions. The solution in [Herdt 2010a] was to solve beforehand another optimization problem with the orientations. Another extension is to use a sequence of QPs where the non linear problem is approximated by building several QPs. This solution is thus very expensive, unless we can limit drastically the number of iterations

We can however notice that [Herdt 2010a]'s algorithm was succesfully used in [Dune 2010] that we will explain in subsection 1.3.8.

1.3.6 Non-linear Pattern Generator

Finally, [Naveau 2017] tackles the non-linearities using a Sequential Quadratic Problem (SQP) solver. The goal is to use iteratively QPs assuming quadratic approximations of the non-linearities. So then, they handle the orientation of the feet for balance and achievability. This method also allows to tackle obstacles avoidance. They use previous solution to warm start the next SQP sequence that improves the convergence to be able to run it online. The problem is now written as follows :

$$\begin{aligned}\min_{\ddot{C}^{\alpha}_{k},F^{\alpha}_{k+1}} \frac{\zeta}{2}\left\|\dot{C}^x_{i+1}-\dot{c}^x_{ref}\right\|^2_2+\frac{\zeta}{2}\left\|\dot{C}^y_{i+1}-\dot{c}^y_{ref}\right\|^2_2 \\ +\frac{\gamma}{2}\left\|F^x_{i+1}-Z^x_{i+1}\right\|^2_2+\frac{\gamma}{2}\left\|F^y_{i+1}-Z^y_{i+1}\right\|^2_2 \\ +\frac{\varepsilon}{2}\left\|\ddot{C}^x_i\right\|^2_2+\frac{\varepsilon}{2}\left\|\ddot{C}^y_i\right\|^2_2 \\ +\frac{\gamma}{2}\left\|F^{\theta}_{k+1}-\int Vel^{\theta,ref}_{k+1}\right\|^2_2\end{aligned} \tag{1.22}$$

Where the last term is new in the cost function, it is non-linear due to the orientation of the feet. We have to notice that α index can either be x, y or θ. Non-linearities due to the rotation appear also in the constraints : balance criteria is expressed as a convex hull polygon support for the ZMP, taking into account the orientation of the foot; a similar formulation containing orientations is written for the reachability polygon of the next foot step.

To improve the solution to be sent on the robot, [Naveau 2017] applies a Dynamic Filter on the previous result. It computes a correction given from the error between the cart-table model used in the SQP and the multibody-computed CoM trajectory.

This corresponds to take gesticulation into account, but the multicontact is not. We will see in Chapter 5 how the DDP algorithm (Differential Dynamic Programming) will fulfill this mission.

The Gepetto team has implementations of these algorithms wich allow to generate motions on the robot. From [Kajita 2003a], these algorithms search for the CoM trajectory, so then they act as centroidal trajectory generator: the third block in Fig. 1.2. From [Herdt 2010a] they also act as contact planner since they find the position of the feet on ground. Implementations often embed a Whole Body Trajectory Generator as [Naveau 2017] which uses it to correct the motion with a multibody CoM pass over the horizon. These works are called Pattern Generators. The goal is to take the highest level inputs as possible to unload the planner cost (that is more time consuming). When using these algorithms, the Stack of Tasks algorithm ([Mansard 2009]) plays the role of WBC as represented by the fifth block of Fig. 1.2. Notably, there is not only an evolution in the complexity of the problem resolution by taking non-linearities into account, but also by the input level from steps sequence to CoM velocity references. In the subsection 1.3.10, we will see how the issues emphasized in these paragraphs can be solved at the Whole Body Trajectory Generator level.

1.3.7 Visual servoing

As mentioned in subsection 1.2.7, a high level feed-back loop should improve the humanoid robots motions. It could be based on external sensor like motion-capture system as explained in 3.2.5.2, or on exteroceptive sensor like cameras. This subsection deals with this last option.

Visual information is often treated separately as a special field of robotics called computer vision. Providing a loop on visual data to generate reactive motions is called visual servoing. There exist different ways to treat these data. If 3D information is provided by the sensors (binocular stereoscopy, infrared pattern deformations, time of flight, structure from motion...), the loop can be based on points or more complex visual features. The control is then living in $SE(3)$ space that corresponds to the natural description of the end effector or camera position and tasks for robots ($SE(3)$ is the Special Euclidean group that describes rigid body transformations). It is called Position Based Visual Servoing (PBVS). In the case that only one camera is available or if 3D information is too noisy or too slow, we need to be able to control them with only 2D information living on the image plane of the single camera. That is called Image Based Visual Servoing (IBVS). IBVS and PBVS control approaches are compared for path following in [Cherubini 2011]. In the 1.2 pipeline, we think IBVS should be linked to the Whole Body Trajectory Generator. The features should be kept simple to allow on the fly computation not far from the WBC rate which is 1kHz while PBVS should be kept in SLAM loop to feed the multicontact planner with slower computation time.

In that perspective, we will describe here IBVS main characteristics and the way they are related to locomotion for driving the motion. Using a camera consists in projecting 3D elements of the scene on the photo-sensitive sensor (CCD, CMOS) of the camera. Values got from the sensor are expressed in pixels. The corresponding model is the pinhole model. Knowing the intrinsic parameters of the camera it allows to project 3D points of interest on the image plane of the camera. It also allows to translate the values from pixels to an euclidian metric associated to the image plane. Visual servoing consists in creating a motion in order to place the camera in a reference placement ($SE(3)$) with respect to the feature. To do so, the error corresponding to the projected reference and the actual projection of the feature is reduced. This difference is called e and a relation exists to connect this difference in the image plane and the joint actuation:

$$\dot{e} = L_e v_c \tag{1.23}$$

This equation links the derivative of the error e with the end effector (supporting the camera) velocity v_c in $se(3)$ space. The formulation of L_e is described in [Chaumette 2006] and developped in Chapter 5 eq. 5.23. The main issue with this matrix is its non-linearity on $SE(3)$ coordinates. Then the velocity v_c can be linked with the joint velocity vector $\dot{q}$ of the robot by

$$v_c = J_c \dot{q}$$

where J_c is the jacobian of the end effector. This brings down to the relationship

$$\dot{e} = L_e J_c \dot{q}$$

The main way to solve this equation to find $\dot{q}$ is to assume an exponential decrease of the solution and to inverse the product $L_e J_c$ using the Moore-Penrose pseudo inverse, which can also be seen as a least square resolution.

1.3.8 Pattern Generator driven by IBVS velocity

The previous paragraph presents a way to control joints of a kinematic chain. [Dune 2010] does not need expression of q to drive the locomotion. They use the extracted v_c value from 1.23 (pseudo-inverse explanations are detailed in [Chaumette 2006]) as an instantaneous input reference in the Pattern Generator problem. Thus, v_c should become $\dot{c}_{ref}^{x,y}$ in the first line of eq. 1.21. The problem lies in the sway motion of the walk, which is the sideways motion created by balancing on one foot at a time during the walk. To avoid injecting the sway motion as reference but just average velocity over two steps, they reconstruct and remove the sway motion from the feature expression assuming that CoM and camera are fixed with respect to each other. We should notice that the behaviour was computed thanks to [Herdt 2010a]'s algorithm. This work is then bringing visual based input into whole body pattern generator.

1.3.9 Linearized visual features embedded in Pattern Generator

The contribution of [Garcia 2014] is to avoid the need of canceling sway motion in [Dune 2010] by putting visual information expressed in the image plane as a cost directly in the optimization problem.

$$\begin{bmatrix} u_{l,j} & v_{l,j} \end{bmatrix} = \begin{bmatrix} u(x^c_{l,j}, y^c_{l,j}, z^c_{l,j}) \\ v(x^c_{l,j}, y^c_{l,j}, z^c_{l,j}) \end{bmatrix} = \begin{bmatrix} x^c_{l,j}/z^c_{l,j} \\ y^c_{l,j}/z^c_{l,j} \end{bmatrix} \tag{1.24}$$

To cope with the non-linearity of the projection equations (that is a prior to find L_e matrix in [Chaumette 2006]), [Garcia 2014] uses first order linearisation :

$$\begin{cases} u(x_0 + dx, y_0 + dy, z_0 + dz) \approx \frac{x_0}{z_0} + \frac{dx}{z_0} - \frac{x_0 dz}{z_0^2} \\ v(x_0 + dx, y_0 + dy, z_0 + dz) \approx \frac{y_0}{z_0} + \frac{dy}{z_0} - \frac{y_0 dz}{z_0^2} \end{cases} \tag{1.25}$$

where l is the 3D landmark index, j is the time index, c is the end-effector index (the camera). That means that the solver will solve the trajectory with just the first point direction to follow to hopefully decrease the feature error. To reinforce the algorithm, they decided to run it in a MPC style, i.e. iteratively sending the first point of the trajectory found to the robot, getting the new state of the robot and running the algorithm again. The formulation (simple QP without non-linearities, but constraints taken into account) becomes now :

$$\begin{aligned} \min \frac{\gamma}{2} \left\| F^x_{i+1} - Z^x_{i+1} \right\|^2 + \frac{\gamma}{2} \left\| F^y_{i+1} - Z^y_{i+1} \right\|^2 \\ + \frac{\varepsilon}{2} \left\| \ddot{C}^x_i \right\|^2 + \frac{\varepsilon}{2} \left\| \ddot{C}^y_i \right\|^2 \\ + \frac{\alpha}{2} \sum \left[S^d_l - S^m_{l,i} \right]^T W \left[S^d_l - S^m_{l,i} \right] \end{aligned} \tag{1.26}$$

where the last line embeds the linearized linear feature expression under a quadratic form for the cost function. Index d means desired, m measured.
Again this QP solver is based on [Herdt 2010a] and adds visual information improving the method of [Dune 2010]. Main issue lies in the first order linearization of the visual features to keep QP formulation.

We have to notice that related works as [Allibert 2008] have proposed similar approaches for generating motions on fixed base manipulators by using Non-linear Model Predictive Control (NMPC) including constraints. The main difference with [Garcia 2014] in our context, is that [Garcia 2014] doesn't embed non-linearities.

1.3.10 Thesis topic

This leads to the topic of this thesis. Whole body trajectory generators led to motions with non-linearities taken into account while visual features expressed in image plane included into the optimization problem ([Garcia 2014]) suffer from linearization due to the QP formulation. We decided first to provide a

loop between all the motions provided in the KoroiBot project (chapter 3) by generating motions from an already implemented [Morisawa 2007]'s algorithm. This loop was based on motion capture inputs as explained in section 3.2.5.2. Results were not as repeatable as expected. We decided to create a base of data and setups to evaluate the different algorithms available on the HRP-2 robot. Then, coming back to the visual features and multicontact requirements, we had here to make a choice. Either implementing the visual features in SQP formalism from [Naveau 2017] achievement, or using another type of solver namely Differential Dynamic Programming (DDP) that is raising up in the humanoid robotics field. It can handle the complete dynamic of the system (not only the cart-table model plus a Dynamic Filter (DF)), takes advantage of the problem sparsity, and most importantly allows to compute multicontact motions in a computation time that is promising. Since it can solve a nonlinear problem, it can embed directly the nonlinear equations of the visual features projections (dark blue cell in table 1.1) contrary to [Garcia 2014] which needs to linearize them. The only theoretical issue of this solver is that it does not handle easily the constraints. A DDP algorithm was a work in progress in the Gepetto team at the moment to do the choice between these two options. We selected the second option not only for the reasons exposed about DDP abilities but also because this could be better integrated in the team's work pipeline. Actually this has led to a complete pipeline integration (light blue cell in table 1.1) for generating the motion presented in Chapter 5. Table 1.1 highlights in green the most suitable functionalities over the three algorithms. A major contribution of this thesis was to embed nonlinear visual features projections in the DDP solver (so then turning the dark blue cell in green in table 1.1). Another was to integrate the DDP with the team pipeline (light blue cell in table 1.1).

Algorithms	[Garcia 2014]	[Naveau 2017]	DDP
Footprints	Free	Free	To be given by the workflow
Optimisation	Linear	Nonlinear	Nonlinear
Centroidal dynamic	Linear momentum	Linear momentum	Linear and angular momenta
Multibody dynamic	No	One pass of DF	Yes
Multicontact	No	No	Yes
Visual features	Yes	No	To be integrated
Inequality constraints	No	Yes	No

Table 1.1: Comparison of algorithm functionalities. Green cells are the most suitable functionalities over the three algorithms, blue ones are the new contributions of my thesis

This thesis is organized as follows: Chapter 2 presents the mechanics particularities of the humanoid robots to highlight further discussions and choices that have

been made during this thesis. Then results obtained for evaluating Key Performance Indicators are the shown in Chapter 3. Chapter 4 presents how motions expressed in lower dimension basis can speed-up computation. Finally DDP algorithm used with visual feature expressions is detailed to generate multicontact motions in Chapter 5.

Chapter 2

Mechanical properties

Contents

This chapter points out mechanical issues that should be taken into account when using a humanoid robot. Since my background is mechanics and mechatronics, I would share the importance of the mechanical phenomena and how much they can impact the motion generation design. This chapter is a part of the big picture corresponding to the objective of making a humanoid robot walk, driven by sensor feedback like camera images. Indeed, plenty of mechanical effects affect the motion realized by a robot and make the control much more hard. These effects should be known, modeled or avoided and reduced if possible. They can drastically impact the control strategy, but often only consequences are considered and sensors feed-backs are needed to correct differences between the predicted motions and the real ones. This chapter present firstly the main leading technologies improving the use of

humanoid robots. Then elasticity in actuators, that could appear detrimental to the control, will prove to be potentially convenient to measure torques on Romeo's actuators. Follows a description of the mechanical phenomena observed on Gepetto's HRP-2 robot that lead the team to specify the design of a new robot, named Pyrène, described in the last section.

2.1 Introduction

The DARPA Robotics Challenge (DRC) was motivated by the design of several high-performance humanoid robots such as ATLAS, or LOLA [Lohmeier 2009]. It has led to the design of new powerful robots such as Walkman [Negrello 2016], S-One/JAXON [Kojima 2015], PROXI/DURUS [Pro 2016], [Hereid 2016] and the upgrade of standard platforms such as HRP-2 with HRP-2 Kai [Kaneko 2015] or HUBO with DRC-HUBO+ [Rainbow 2017]. In the same way that the first DARPA Grand Challenge pathed the way for industrial researches on autonomous cars, the *DRC* led to a renewed interest in humanoid robotics for industrial purposes. Now, it is interesting to investigate the outcomes of the technologies developed for this competition and to integrate some of the lessons learned on this occasion. To realize such investigation, the difficulty is to find an available prototype of a humanoid robot that integrates the recent technologies. This section introduces a work in which I was involved as coauthor and mechanical referent for the Gepetto team: [Stasse 2017].

2.1.1 Availability of recent humanoid robot technologies

Humanoid robots have to perform reactive motions to keep balance, which requires actuators that can generate both high torques and high speeds. We clearly quantified those requirements for possible industrial applications through specific examples in [Stasse 2017]. According to [Englsberger 2014], the Atlas robot provides elevated torques with high bandwidth, but is generating loud sounds, needs a lot of power and has high friction/stiction that complexify its control. In addition, discussions with industrial partners pointed out that hydraulic systems have less chance to be certified than electrical motors based ones due to the huge pressure involved, and the difficulty to implement the needed security features. On the other hand, the seminal work of [Urata 2010], leading to the design of the S-one robot from Schaft, shows that it is possible to realize high power motions with electric motors. WALK-MAN [Negrello 2016] and DRC-HUBO+ [Rainbow 2017] are robots using this kind of technology. WALK-MAN has new high performance actuators with impressive capabilities, but it is a laboratory prototype and the maintenance and support are difficult to perform outside IIT (Istituto Italiano di Tecnologia). The DRC-HUBO+ (leg joints) as well as the HRP-2 Kai (pitch joints of the legs) have two motors in parallel, increasing the torque capabilities but not the speed. In addition, the DRC-HUBO+ is using a CAN bus, which is a bottleneck for high-frequency (e.g. $1khz$) access to the low-level controllers. To mitigate this problem, recent robots

use EtherCAT (e.g. HRP-2 Kai and WALK- MAN) as their communication bus. Humanoid robots such as Valkyrie [Radford 2015] or WALK-MAN [Negrello 2016] use Serial Elastic Actuators (SEA). SEA can absorb and partly store the impact energy and release it later. In addition, the spring can be used to measure the joint torque. However, SEA makes the robot control more complex as the actuator dynamics needs to be taken into account [Paine 2015]. Torque controlled robots such as TORO [Englsberger 2014] or DURUS [Hereid 2016] are very promising to deal with unplanned contacts with the environment and humans. Unfortunately TORO is unlikely to be available outside DLR (Deutsches Zentrum für Luft- und Raumfahrt - German Aerospace Center). Recent robots such as REEM-C have encoders both on the motor and at the joint side. Provided that the encoder resolution at the joint side is sufficient, it is possible to use them to model the deflection induced by the harmonic drive. This deflection is proportional to the torque. DURUS is another new humanoid robot, sold by SRI International (from the Stanford Research Institute), with impressive efficiency using springs at the ankles and torque control. The transmission has been designed to have low friction, and the robot has torque sensing in all the joints. However, while the spring set along the vertical axis at the ankles is efficient to store energy and absorb impacts, its implication during manipulation is less clear. Our experience on several applications is that humanoid robots are usually used in acyclic behaviors and mix high-stiffness (for manipulation and support legs) with low-stiffness (for impacts and human-robot interaction). This stiffness regulation can be provided either by mechanical design or by the control. A mechanical compliance can be particularly useful during feet landing to filter the impacts. However, during high-precision manipulation the mechanical compliance needs to be controlled, and most of the current solutions are not able to provide the same stiffness as a rigid system. Therefore the approach for Pyrène [Stasse 2017] was to have a controlled compliance instead of a mechanical one. As torque control is still a research topic, Pyrène allows for both position and torque control. Finally the recent advances on whole body control rely mostly on complex optimization problems solved in an efficient manner [Feng 2016], [Sherikov 2014], [Kuindersma 2016], [Carpentier 2016]. This is possible thanks to the significant embedded computational power available on the robot.

2.1.2 Applications

In [Stasse 2017], we focused mostly on scenarios coming from aircraft manufacturing. Indeed, for tasks like climbing stairs or moving in narrow passages, humanoid robots have an obvious advantage over other mobile platforms. Another point that drove the robot specifications was musculoskeletal stress in human operators. Operations in aircraft manufacturing imply frequent works in narrow spaces while handling heavy tools with stretched arms. This imposes to have a robot able to handle tools weighing up to 6 Kg with stretched arms. The environments in which the robot has to evolve include narrow spaces and stairs, similar to the ones found in the *DRC*. Although robots with wheels have already been investigated in aircraft industry, they

involve the need of elevators and thus increase the cost of deployment. An important factor for the deployment of humanoid robots in real scenarios is repeatability. This goal can be reached when the design of the robot is done in collaboration with an industrial partner mastering the integration knowledge, or with institutes having strong technological centers such as DLR [Englsberger 2014] and IIT [Negrello 2016]. The successful production of the HRP series [Bouyarmane 2017], [Kaneko 2008], and the robots from Boston Dynamics are good examples of robots with industrial quality. For this reason, Pyrène, the robot presented in [Stasse 2017] and introduced in 2.4, was built by the company PAL- Robotics. PAL-Robotics is also the manufacturer of the REEM-B [Kaneko 2011] and REEM-C humanoid robots.

In the following sections, we will describe some of the main state of the art issues concerning mechanical phenomena in humanoid robots. Firstly, we will describe how to use the inherent elasticity of actuators to measure and control the series elastic actuators stiffness. Then, the different mechanical issues observed in the very reliable robot HRP-2 will be exposed. Finally, the new technologies embedded in Talos robot will be presented.

2.2 Romeo

In the Romeo project I was in charge of implementing the Stack Of Tasks (SOT) [Mansard 2009] on the robot. That was achieved with a visual-based whole body control before the beginning of my PhD studies and no publication was realized at that time. At the beginning of my PhD studies, I provided identifications of the actuator test-bench elasticities as well as the corresponding coefficient improvements using the Stack of Tasks to feed the control architecture of the test-bench. This work led to a complete re-identification of the model of Romeo's actuator test-bench and the use of a Differential Dynamic Programming scheme to control the test-bench. A publication of which I am a coauthor has followed this work: [Forget 2017].

2.2.1 Introduction of SEA actuators

As previously mentioned the impact resiliency is critical for a humanoid robot for each contact creation with the environment. The impact should be the smallest as possible to avoid breaking mechanical parts and making the sensor-command loop unstable. Another decisive point is to be able to get the force information from the environment to improve the contact itself. In the case of an actuator, we try to get the torque information really generated after the transmission chain that includes all the barely modeled friction effects; this is called the transparency of the actuator. On many electrically-powered humanoid robots, strain-wave gears (e.g. Harmonic Drive gear) are used for their compactness. However, when back-drivable, strain-wave gears have poor transparency, i.e. the torque exerted at the joint level (output) is poorly correlated to the torque at the motor level (input), and the output torque is difficult to estimate from the motor current. If an accurate joint-torque

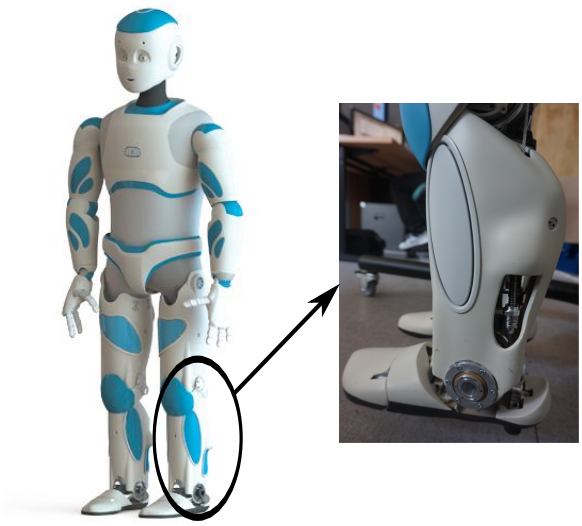


Figure 2.1: The leg of the robot Romeo is designed based on a screw-nut-cable actuator, which is shock-proof and low-friction, but induces flexibility due to the cable.

estimation is needed, a joint torque sensor must be added to the robot design, which increases the total design cost and the actuation flexibility. Moreover, strain-wave gears are sensitive to impacts, which tend to damage the gear. Their maximum torques are also limited, in particular when impacts have to be expected. For the design of full-size humanoid robots (i.e. size similar to Shaft, NASA Valkyrie, PAL-Robotics Talos), strain-wave gears are clearly one of the main limiting factors of the design. On the other hand, most alternative gears are either not compact enough, or have insufficient reduction ratio. The design of such kind of actuators is a widely studied subject. Different technologies are used to address these problems. Electric-based actuation is very desirable because it is simple to implement, hence more reliable for a given integration effort. A first step is to adapt electric motors to humanoid robotics needs as done in [Wensing 2017] for quadruped robots. By increasing the motor diameter, the nominal speed is lowered while the nominal torque is improved, allowing a decrease of the reduction ratio, which, in turn, improves the back-drivability of the system. Another route is to improve the design of the gear box, as done in [Englsberger 2014] where authors chose to improve strain-waves gear technology to build a torque-controlled robot. However, despite the improvement, strain-wave gears imply many drawbacks: insufficient torque limits, sensitivity to impact, lack of efficiency and of transparency. Adding a passive element at the gear output relaxes some of the limitations: it protects the gear from impacts. It also makes a section of the actuation transparent, while an encoder may be used to directly (after calibration) measure the torque applied on the joint side. However, the passive element makes actuation more difficult to control and intrinsically lowers the possible control bandwidth, which is not desirable for achieving fast and

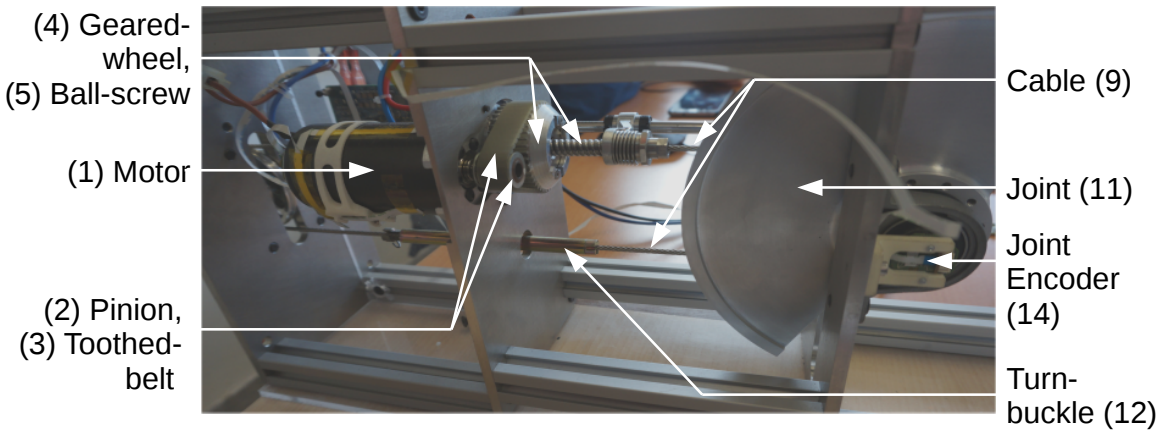
dynamic movements. Variable-stiffness actuators as in [Wolf 2008] make it possible to dynamically stiffen the robot when high dynamics are needed, but have the same limits as rigid electric actuation. On quite a different route, hydraulic technology is promising to conceive robotics actuators allowing good power to weight ratio and shock absorption [Semini 2011, Alfayad 2011], although the implementation of the complete robot becomes more challenging.

In this section, the actuator built by SoftBank Robotics (previously Aldebaran Robotics) from [Garrec 2010]'s original design is presented. This actuator induces a flexibility coming from the cable connecting the screw to the joint output. Adding elasticity into the actuation smooths the contact with the environment, which prevents rebounds and in certain cases sliding effects [Lee 2016]. The flexible element in this particular gear can also be exploited to directly measure the output torque, by equipping it with a sensor able to measure the spring deflection (e.g. angle encoders attached to each side of the elastic element). We show that measures of torques/forces can be obtained for quasi-null additional cost. The flexible element behaves like a series-elastic actuator (SEA) ([Pratt 1995]). It must be taken into account in the actuator control loop to avoid instability. However, the flexibility is an order of magnitude smaller than on typical SEA.

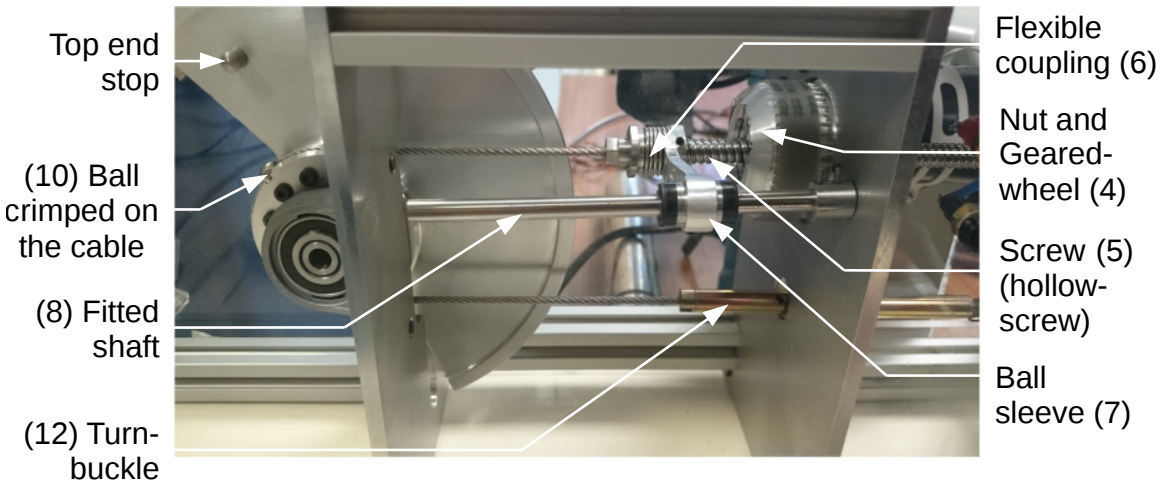
2.2.2 Romeo's legs design and identification

2.2.2.1 Design

In this subsection, the actuator general mechanics of Romeo's leg (Fig. 2.1) are summarized. The actuator is composed of an electrical motor attached to a ball screw guided along a fixed axis but which can freely rotate inside the nut. The output of the screw is connected to two cables which can pull the output joint in the two rotation directions. Our particular actuator is mounted in a test-bench used for identification and control validation. The same actuator equips 10 degrees of freedom of the legs of the medium-size humanoid robot Romeo. Two pictures of the actuator with legends are shown in Fig. 2.2. A schema of the actuator is shown in Fig. 2.3. The motor (referenced as (#1) in Fig. 2.2) is fixed on the base, a pinion (#2) is mounted on its shaft. The pinion leads a toothed belt (#3) to a geared wheel (#4). This part is fixed to the nut of the ball screw (#5). The screw is the main component, allowing the trade-off between a high reduction ratio (of about 100) and a high reversibility. It also increases the compactness of the system. To avoid the screw rotation around its main axis and to enforce its motion to be a translation, an additional part is flexibly coupled (#6)(#7) between the screw and a fixed shaft (#8). Note that this part is not introducing the elasticity we try to manage in [Forget 2017] paper. The cable (#9) is the main part of the system introducing the flexibility we deal with in this work. The forward part of the cable is linked to the joint with a crimped ball (#10) placed in the spherical imprint of the joint (#11). The cable then goes to the turn-buckle (#12). The backward part of the cable is also going to the turn-buckle by the way of a pulley. The turn-buckle is used to fix and



(a)



(b)

Figure 2.2: Right (top) and left (bottom) views of the actuator.

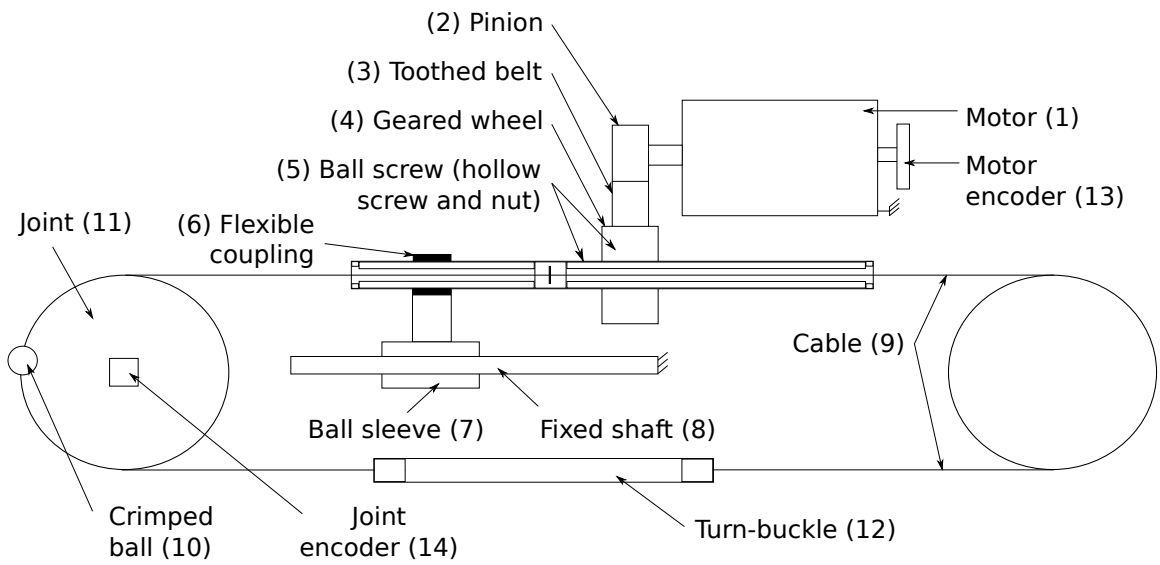


Figure 2.3: Schema of the actuator mounted on the test-bench.

preload the cable. To keep the test-bench simple to use, a rope is attached to the joint (#11) in order to apply some load. This setup limits the output load to only one direction of the joint. This has no negative consequence for our experimental protocol in comparison with the real robot. To measure the angle positions, two absolute magnetic encoders are mounted on each side of the gear. One is fixed behind the motor on its main shaft (#13), the other is placed on the joint (#14), after the transmission chain. This layout measures the ratio and the deformation of the transmission and makes the model parameters theoretically observable. Even though the flexible coupling should increase the transparency of the system, it seems to create constraints on the screw by preventing it from oscillating freely. Moreover, contrary to [Garrec 2010], the space around the cable (attached to the middle of the hollow screw with a crimped sleeve) is not sufficient, generating constraints in the mechanism. These constraints create efforts and introduce non-linear frictions depending on the actuator angular position.

2.2.2.2 Identification

First of all, off-line estimation of the test-bench have been conducted: all measurements (joint position, motor position, motor current, motor supply voltage ...) are collected while controlling the actuator with a simple controller (PID with low gains). The estimation of the model parameters is done using MATLAB. Thanks to the transparency of the actuator, the model is easily identified. The prediction in simulation properly fits the real trajectory. Although the parameters are better estimated than on other types of transmission, the identification is not perfect. From the captured data, we identified that this comes from several defects in the implementation of the actuator (ball-screw being too constrained by the flexible

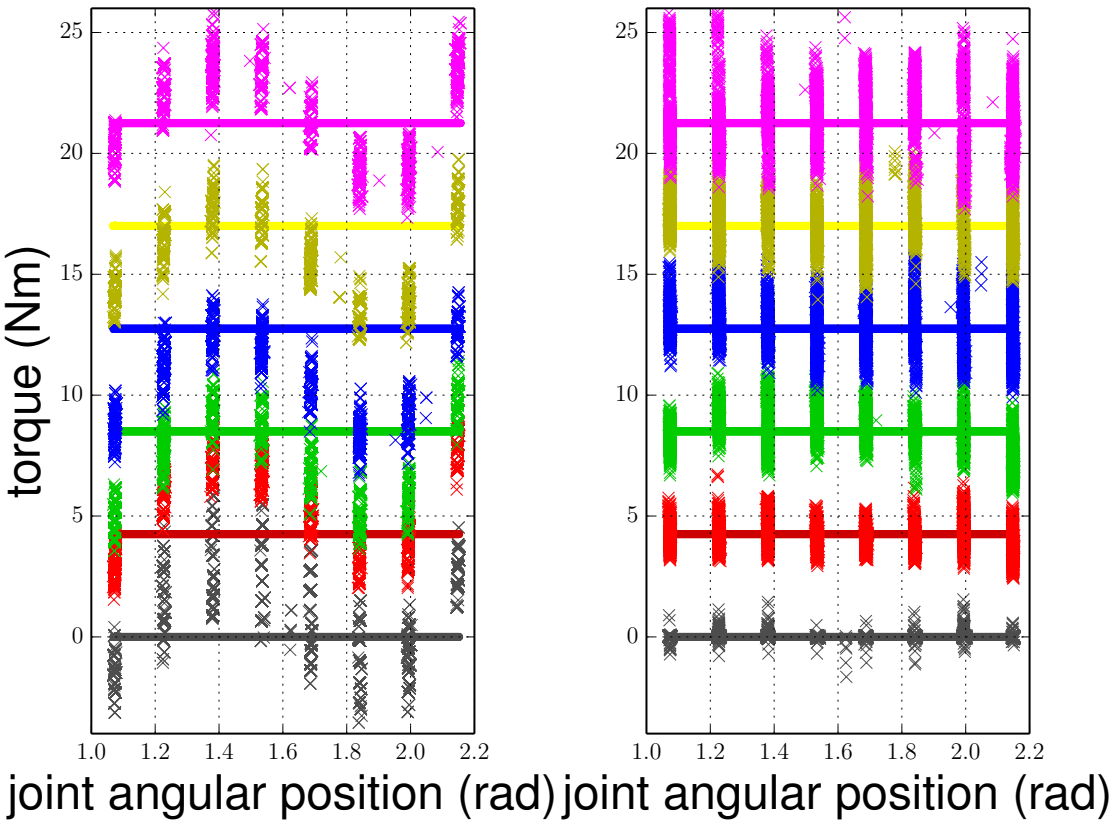


Figure 2.4: Output joint torque estimation: (left) using only the two joint encoders measuring the spring deflection; (right) using the current measures and the full actuator model. The estimation from the encoders is biased by the friction in the hardware. The current measure leads to a quite good torque estimation although noisy. Both measures are satisfactory given the absence of a direct torque sensor, and are complementary. Each color represents a different output load (masses attached to the actuator output).

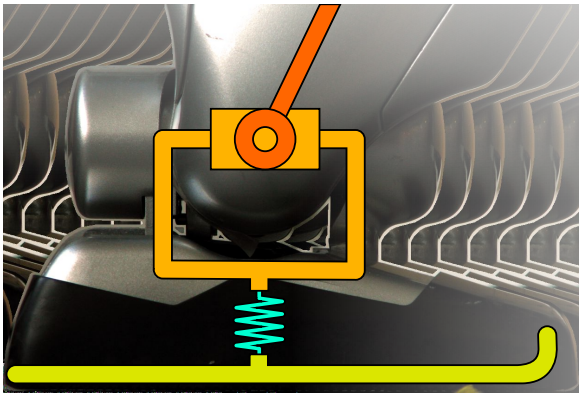


Figure 2.5: Rubber bush between the sole and the ankle of the robot HRP-2. Image coming from [Benallegue 2015].

coupling, cable not being free enough at the mounting point with the ball-screw, elasticity being different in the two directions due to the unequal lengths of the two cables). It would be possible to model these effects, hence to obtain a better prediction. However, it would also make the controller more complex and more costly. We rather believe that it would be easier to correct this effect by a more careful physical implementation of the actuator.

Secondly, on-line estimation have been conducted: the actuator is not equipped with a direct torque sensor. However, two indirect measurements are available. We have two encoders on each side of the flexibility, and can then use the model to estimate the output torque. Thanks to the actuator transparency, we can also use the measured motor current to estimate the output torque. To validate both measurements, we took measurements points for different joint positions in a static state with different known masses attached to the actuator output, this is our ground truth. The mass being static, the output torque is known and can be compared to the estimation using either the encoders or the current sensor. The result is displayed in Fig. 2.4. Both estimations are accurate. They are also complementary: the estimation from the encoders is biased by friction; the estimation from current is more noisy. In conclusion, the transparency of the actuator leads to accurate model estimation, both for (off-line) calibration and (on-line) torque estimation.

2.3 HRP-2

This section presents the main mechanical issues the team had to face with the robot HRP-2 to make it move correctly.

2.3.1 Flexibility in the ankle

As described in [Benallegue 2015], the ankle of the HRP-2 robot (Fig 2.5) contains a bush to absorb impacts in order to protect the 6D force sensor in the ankle. As dynamics calculations assume the rigidity of all the parts of the robot, this flexible

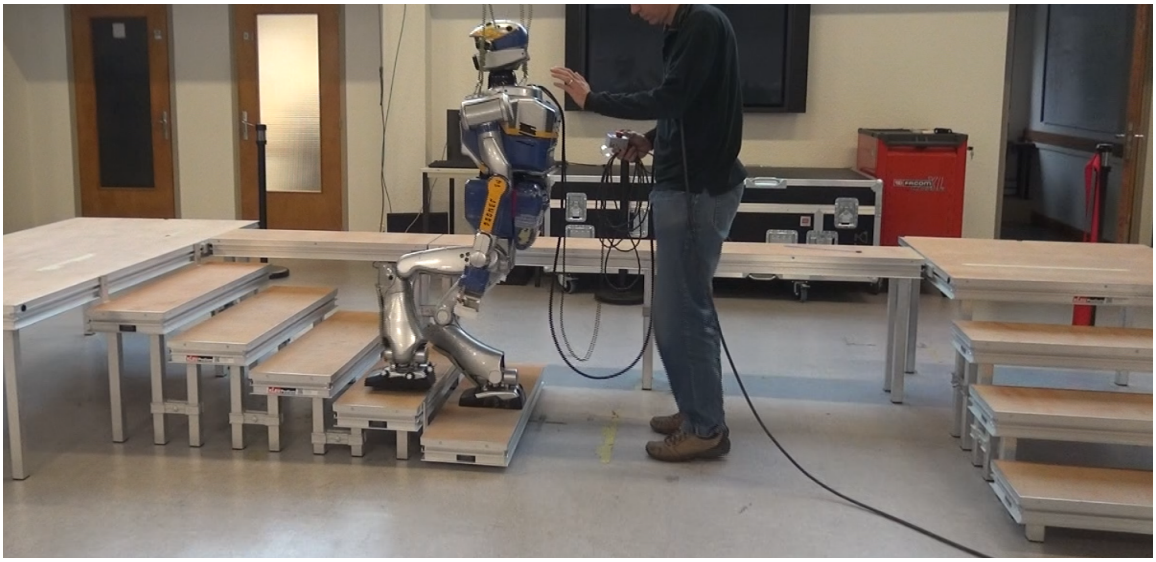


Figure 2.6: HRP-2 robot climbing 10cm stairs

element implies errors in the model that have to be taken into account. For instance, the encoder-based estimation of the base position is biased by the deformation of this bush. Then, the dynamical information given by the torque-force sensor in the ankle needs to be enriched with the deformation model of the bush. These elements imply the use of a special stabilizer module to be able to make the robot walk. In our case the stabilizer module is given by the company which provided the robot: Kawada Robotics. This is considered as a black box in our case. The advantage of this block is to extend the stability region of the robot motions. The drawback consists in a modification of the motion of the robot in an unknown manner, changing either the dynamics or the placement of the feet when the robot walks. This could explain partly the incoming issues of the Chapter 3.

Generally speaking, flexible parts in the kinematic chain represent an impediment for controlling the robot precisely. That often implies estimation of the compliance and specific blocks to be controlled or taken into account. We will see in section 2.4 that a similar phenomenon occurs on Pyrène.

2.3.2 Kinematics of the legs

The kinematics of humanoid robots bring inconvenience in the realization of motions that appear simple for humans. The obvious reason is that the robot body is much simpler than the human's one. It generally contains around 30 degrees of freedom whereas the human body is composed of more than 200 bones and 600 muscles with complex joints. HRP-2 robot is $154cm$ high and has 6 degrees of freedom in each leg. Kinematicaly speaking that means that for a defined pose of the foot in space ($SE(3)$ placement), there exists only one configuration of the kinematic chain from the pelvis (referred as the base body or the free-flyer) that reach this position. This

has two effects. Firstly, it can be used for speeding up calculations of joint positions when the center of mass and feet trajectories are known. Inverse geometry can be used instead of the Jacobian pseudo inversion or optimization problem. Secondly, for climbing stairs (10 or $15cm$) (see Fig 2.6), the trajectories of the feet to avoid the steps make the joint trajectories really close to the joints limits of the ankle. Contrary to the human body, the feet of the robot are not articulated with a toe joint and the robot has to place the foot completely flat on the surface when it is moving. Feet contact is not allowed to be on the edge of the sole to improve the capabilities of the robot to go upstairs. This explains partly why the joint trajectories are so close to their limits. In the stairs case, the motion is not calculated online, the center of mass height trajectory can be modified by some user parameters as well as feet trajectories (see section 1.3). These parameters are bounded because of the joint limits. These parameters are often tuned by users' knowledge in an iterative manner.

2.3.3 Power to achieve motions

HRP-2 robot was not designed for achieving high dynamic motions. We reached the dynamical limits when trying to climb stairs of $15cm$, which is trivial for a human. The bottleneck in this case was not the mechanics itself but the use of the motors compared to the capacities of the power supply. The sum of the motor and electronic boards current loads in that demonstration is more than the battery alone can feed (peaks up to $40A$ were observed). That leads to a failure of the computers of the robot because of a correlated voltage drop. A solution was to plug the robot to the main power supply. In that situation, the interaction between the power asked by the motion (velocity and torque references) and the capacities of the battery is a limit to tune the motion we want to reach. This phenomenon is clearly a mechatronics problem. It is improved in the new generation robot Pyrène where the battery has an output of $74VDC$, with a capacity of $15Ah$ and peaks up to $150A$ without voltage drop.

2.3.4 Repeatability in the walking motions

As previously mentioned, unexpected behaviours can be created by multiple factors coming from mechanics, electronics or software parts. In the case of HRP-2, we noticed a repeatability issue for locomotion. Firstly, we observed differences between reference distances and achieved ones using the pattern generator of [Morisawa 2007]. This pattern generator needs to be fed the step placements during the walk and also ensures that these placements are normally reached. Trials were done given different length step placements for the feet. None of them were reached. The errors laid between 5 and $12mm$. We then changed the pattern generator used. We tried [Kajita 2003a] with the Kawada's interface, i.e. the one used for the maintenance tests of the robot. The same errors were observed. The explanation could lie in the mechanics and the calibration protocol. Each time the robot is turned on, a

calibration phase is needed. The actuators push each limb against its joint stops to check if encoders data fit their nominal values. If not, that may imply that the transmission has moved and slipped. The solution is to overwrite encoder values with the new one. That often happens after dynamic tests. For the legs, this method can bring up little errors caused by parts wear. A deeper calibration is then needed to make the model correspond to the mechanics. That deeper calibration was made by Kawada but errors were still observed.
We were not able to improve the precision of the placement of the feet for one step walking motions. That could be due to parts wear that makes the model wrong enough to lose the expected precision of foot placement. The motion is stabilized by the Kawada stabilizer that could modify the motion and change the foot placement. This problem was the main issue for aligning motion in front of obstacles in the Koroibot project (see Chapter 3). That especially makes it nearly impossible to have a good angle before starting the locomotion motions and makes the robot fall before the end of the motion.

2.4 Pyrène - Talos

Following the lack of power and kinematics limitations of the HRP-2 robot, the Gepetto team decided to buy a new robot which would fulfill their expectations. The team made the specifications of the robot that PAL Robotics then manufactured: the Talos series. The Gepetto team bought the first one of this series and gave it the name of Pyrène. In this manuscript, referring to this robot by either Pyrène or Talos name is equivalent. This section details some of technologies contained in this robot. They are related to the specifications provided by the Gepetto team. It refers to a contribution as coauthor in [Stasse 2017].

2.4.1 Kinematics constraints

The kinematics of Pyrène is depicted in Fig 2.8.

1. *Range and overall structure*: Pyrène has almost the same joint range as the average human. Its range of motion is wider than ATLAS, except for the ankle inversion. Conversely it has a slightly smaller range than WALK-MAN for almost all joints. It has 32 DoFs as most recent humanoid robots, like HRP-2 Kai, WALK-MAN, and DRC-HUBO+. In terms of weight, with $95kg$ Pyrène is heavier than DRC-HUBO+ and HRP-2 Kai, but lighter than WALK-MAN and ATLAS.

2. *Shoulders*: In contrast to HRP-2 and HRP-2 Kai, Pyrène has been designed to have a maximum manipulability in its front in order to perform drilling and screwing motions. For this reason, the first axis of the shoulder, instead of being along the pitch axis, is along the yaw axis. In this way, when both shoulders are folded in the front the robot has a width of $550mm$ instead of $775mm$. It becomes then more narrow than HRP-2 Kai. In addition,

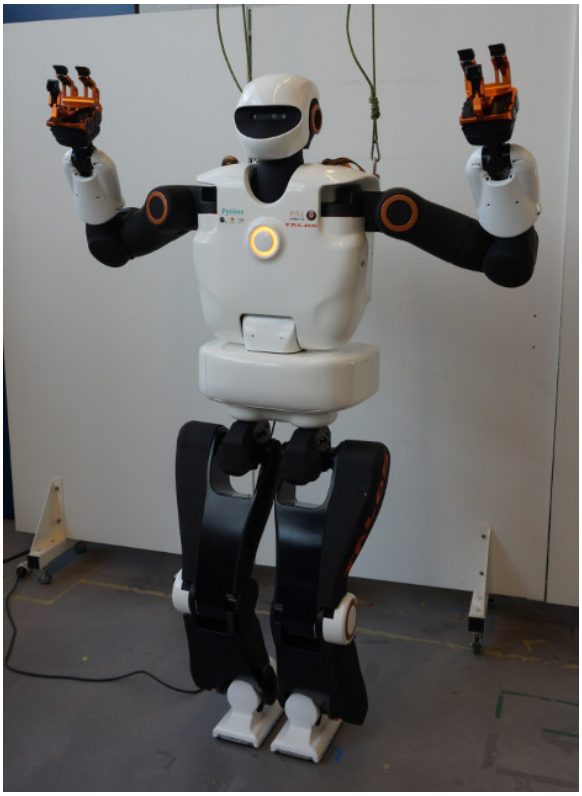


Figure 2.7: Pyrène: the first robot of the TALOS series built by PAL-Robotics

HRP-2 has only 6 DoFs in its arms, thus no redundancy to control the 6D pose of the end-effector. While this simplifies the computation of the inverse kinematics, it severely limits the manipulability if additional constraints need to be handled. The generalization of numerical methods to deal with the problem of redundancy is now less interesting.

3. *Hip and knee*: From the kinematic viewpoint the cantilever structure of HRP-2 is interesting to alternatively put one foot in front of the other when going through narrow spaces. However, this structure is not ideal for legs with higher-powered motors because it puts more stress on the mechanical structure and increases the width of the robot. For these reasons, the two legs, although close to each other, are designed not to collide while spanning a wide range of motion.

2.4.2 Batteries

Pyrène is equipped with Li-CNM (Cobalt-Nickel- Manganese) batteries, which are able to deliver 74V DC with a capacity of 15 Ah. Compared to HRP-2 Kai the capabilities of the batteries are 50% higher. Finally, if powerful motions are needed, the batteries are also able to deliver peaks of 150 A. These capabilities are driven by the work of Urata, which led to the design of the S-One robot [Urata 2010].

2.4.3 Actuators

For each joint, a brushless motor is linked to a Harmonic Drive (strain wave gear), which is itself connected to a torque sensor. The maximum peak motor torque given by the data sheet is usually higher than the maximum peak torque for the Harmonic Drive (HD). In addition the peak motor and HD torques can be exceeded during a short time interval. Finally, the torque sensor is connected to a link. Two high-precision encoders (19 bits) measure the motor and joint positions. Finally, this actuator allows for both position and torque control.

2.4.4 Sensors

1. *IMU and force sensors*: Like many of its peers, Pyrène is equipped with an IMU placed at the level of its waist. This allows measurements of the trunk orientation with respect to the gravity field and the robot global acceleration. In addition, Pyrène is equipped with 6-axis force/torque sensors at the hands and the feet. The main difference with respect to HRP- 2 is the capacity for these sensors to sustain up to 6 times the weight of the robot, whereas it is only 2 times on HRP-2.

2. *Torque sensors*: Pyrène is equipped with torque sensors in almost all the joints, except the wrists and the two head DoFs. They directly measure the torque applied on the load side. We believe that the redundancy of the sensors, as well as their fast update thanks to the EtherCAT bus, are a key ingredient to achieve torque control on humanoid robots.

3. *Vision*: Pyrène is equipped with an ORBBEC Astra Pro RGB-D camera. The camera providing the RGB part is a CMOS camera using a rolling shutter system, which can be problematic with dynamic scenes. The head has been designed to be modified, and we plan to add a stereoscopic system, or at least a CCD camera, together with an IMU for later investigation.

2.4.5 Computational power

The robot is supplied with two computers, each equipped with dual i7 CPU at 2.8 GHz. Each CPU has two cores and is hyper-threaded, which gives a total of 8 cores per computer. However, since the real-operating system used is RT-PREEMPT, only 4 cores are available on the control computer. Eight cores are anyway available on the computer for vision and high-level computations. The motherboards are in a specific box called the logic box, which can be easily removed. This ensures proper cooling of the CPUs and facilitates their upgrade, a key factor for Pyrène extended lifespan.

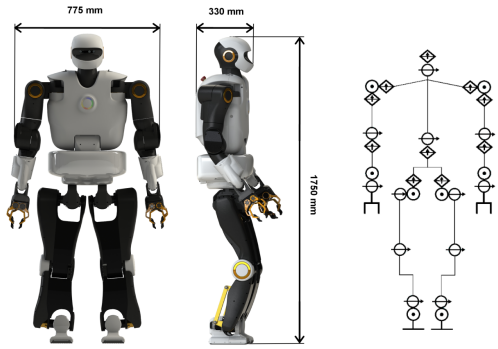


Figure 2.8: Pyrène's size and kinematics

2.4.6 Software

PAL-Robotics has a history of providing its robots (REEM-C and TIAGO) with ROS deeply integrated. The operating system used in Pyrène is an Ubuntu 16.04 LTS. The robot low-level system is developed using ROS control. The robot can be simulated with Gazebo, and the multimedia system is using the ROS stacks providing navigation and map-building.

2.4.7 Flexibility in the hip

Unexpected flexibilities were observed in the hip of the robot, in an axis not supported by a torque sensor. This flexibility is then not directly observable. No deeper investigation has yet been done to explore that problem. To correct the flexibility effects, a stabilizer is needed. Currently the Gepetto team is working on an implementation of [Caron 2019]'s algorithm. First results allow to reach $10cm$-length steps (single support $0.9s$, double support $0.115s$).

2.5 Conclusion

We have seen in this section that mechanics and mechatronics of humanoid robots still have open problems. From performance requirements to safety necessities, trade offs have to be made among actuation technologies (electric or hydraulic for instance), flexibility in the contact and actuation, sensors to enable detection and flexible control, even in communication buses and algorithms to fasten up computation on highly dynamic phenomena. Successions of tested technologies in Gepetto led to the design of a new robot specified to achieve highly dynamic tasks that correspond to trivial behaviours for a human. As the architecture of this kind of robot is complex, each unexpected phenomenon between the calculation and the response of actuators or environment implies potentially huge modifications to bring in each level of the robot, in the non-linear online constrained optimization algorithms for instance. Mechanics and low-level control takes a really important

place in controlling humanoid robots. In our case, some dead-ends were reached because of badly handled blocks. This will be shown in Chapter 3.

Chapter 3

Benchmarking

Contents

In the previous chapter, we have seen that repeatability of some motions is disappointing. In Chapter 2 we have raised some calibration and mechanical wear issues. We must have a procedure to evaluate the motion of our robot, not only to quantify wear over time, but also to have comparison indicators between the robots and the algorithms. We should be able to compare works of different research groups,

with different robots, on common indicators. The work presented in this chapter is referring to previous results obtained during the KoroiBot project [Torricelli 2015]. More specifically it provides a comparison procedure. We have followed this method to generate data about the robot HRP-2 and the algorithms available at that point. This chapter is mainly based on a journal paper produced during this PhD as coauthor : [Stasse 2018]. My personal contribution consisted in realizing the in situ tests with the collaborators and the robot. I was also in charge of analyzing the data collected during the tests to generate the results.

3.1 Introduction

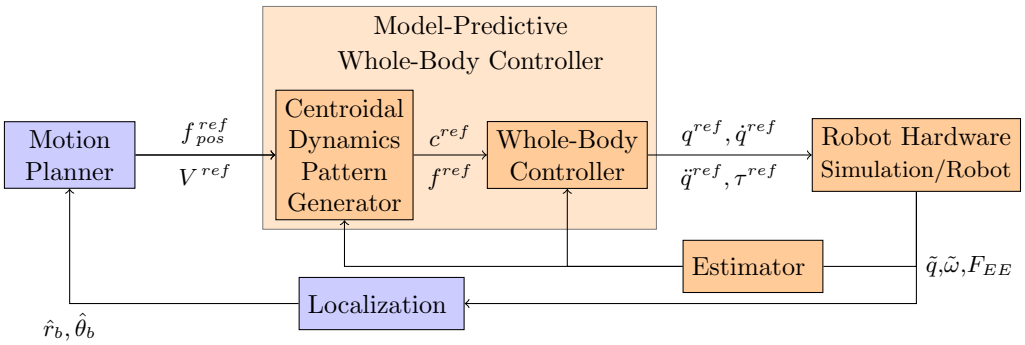


Figure 3.1: A conventional architecture used to generate humanoid robot motions. In this work the modules in the orange boxes are the ones that are benchmarked, whereas those in blue are not benchmarked

From the seminal work of [Chestnutt 2010] to the recent methods proposed in the frame of the Darpa Robotics Challenge (DRC) [Tsagarakis 2017, Lim 2017, Radford 2015, Johnson 2017, Marion 2017, DeDonato 2017], the motion generation of humanoid robots lies in a control architecture that roughly follows the general framework depicted in Fig. 3.1. We can notice that this framework is different from the one presented in section 1.2. The reason is that section 1.2 is based on Gepetto team workflow. This section has no focus on a particular team and summarizes a general framework adopted in the different references. To bridge the gap, let us give the correspondences between this section and section 1.2. The two first blocks of Fig. 1.2 (Guide path generator and Contact planner) are merged in the **Motion Planner**. The third block of Fig. 1.2 (Centroidal trajectory generator) corresponds to the **Centroidal Dynamics Pattern Generator**. Finally, the Whole body trajectory generator is merged into the Whole body controller of Fig. 1.2 in such a way that only the **Whole-Body Controller** remains.

Based on an internal representation of the environment and on the localization of the robot ($\hat{r}_b$ and $\hat{\theta}_b$ being respectively the base position and orientation), the **Motion Planner** (**MP**) plans a sequence of reference end-effector contact positions

(f^{ref}), or a reference center of mass linear velocity combined with a reference waist angular velocity (V^{ref}). These references are then provided to a **Model-Predictive Whole-Body Controller** (**MPWBC**) which generates a motor command for each joint (joint torques (τ^{ref}), positions (q^{ref}), velocities ($\dot{q}^{ref}$) and accelerations ($\ddot{q}^{ref}$)). This block is critical in terms of safety as it maintains the dynamic feasibility of the control and the balance of the robot. The **Model-Predictive Whole-Body Controller** can be expressed as a unique optimal control problem but at the cost of efficiency in terms of computation time or solution quality. This is why this controller is usually organized in two stages. First, trajectories for the robot center of mass c^{ref} and positions of contacts with the environment f^{ref} are found using a **Centroidal Dynamics Pattern Generator** (**CDPG**). Then, a **Whole-Body Controller** (**WBC**) computes an instantaneous controller enabling to track these trajectories. More details about the **CDPG** can be found in the next paragraph. The whole body reference is in turn sent to the **Robot Hardware**, which can be either the simulator or the real robot. The feedback terms are based upon the measurements of the different sensors. The encoders evaluate the joint position ($\tilde{q}$). The inertial measurement unit (IMU) measures the angular velocity ($\tilde{\omega}_{IMU}$) and the linear acceleration ($\tilde{a}_{IMU}$) of the robot torso, which give information about the orientation of the robot with respect to the gravity field. Finally the interaction with the environment is provided by the force sensors classically located at the end-effectors ($F_{EE} \in \{F_{RF}, F_{LF}, F_{RH}, F_{LH}\}$, where the subscripts have the following meaning: (EE): end-effector, (RF): right foot, (LF): left foot, (RH): right hand, (LH): left hand). All these information are treated in an **Estimator** to extract the needed values for the different algorithms. Finally the **Localization** block is used to locate as precisely as possible the robot in its 3D environment. Various implementations of this architecture have been proposed with various levels of success from the highly impressive Boston Dynamics System, to robots widely available such as Nao. An open question is the robustness and the repeatability of such a control system as well as its performance. In this work, our main contribution is to propose a benchmarking of the HRP-2 robot in various setups and provide performance indicators in scenarios which are possibly interesting for industrial applications. We hope this study will provide a quantitative comparison and will serve as a baseline for the elaboration of new algorithms. In addition we believe that this work is one of the first attempts to apply the detailed performance indicators provided by [Torricelli 2015] to a human size humanoid robot. The chapter is structured as follows: firstly, the section 3.2 presents the related work on control and benchmarking for humanoid robots. To continue, section 3.3 lists the materials and different methods used to perform the benchmarking. In turn section 3.4 shows the experimental results using the indicators from section 3.3. Finally, the conclusion in section 3.5 summaries the contributions and results of the study.

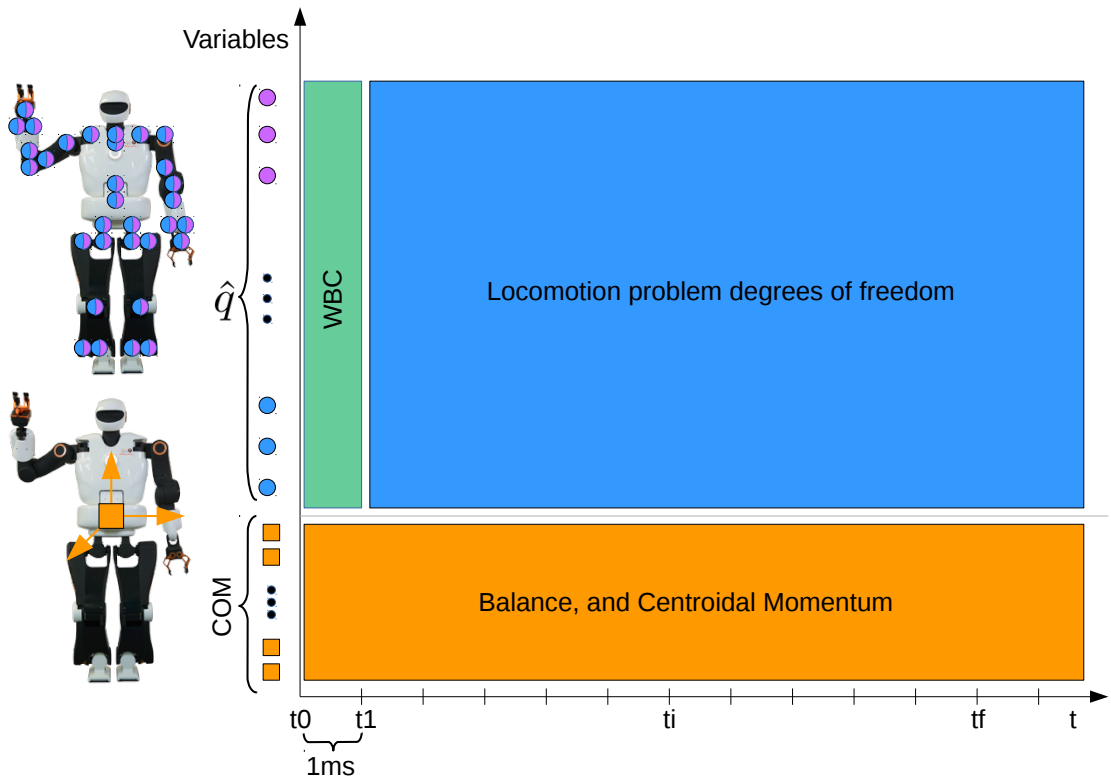


Figure 3.2: Representation of the dimension of the locomotion problem. The abscissa represents the duration of the predicted horizon S and the ordinate the number of robot DoF

3.2 Related work

In this paragraph we present the work that has been done relative to the control and the benchmarking of the HRP-2 humanoid robot.

3.2.1 Motion generation for humanoid robots

The different benchmarks included in this work are relative to the **MPWBC** sketched in Fig. 3.1. This related work is presented in this first subsection. Several techniques are used to mathematically formulate this problem. For instance hybrid-dynamics formulations as proposed by [Grizzle 2010] or [Westervelt 2007] are efficient but difficult to generalize. The approaches used in this work are based on mathematical optimization which is broadly used in the humanoid robotics community. More precisely, the locomotion problem can be described as an **Optimal Control Problem** (OCP). The robot generalized configuration (q^{ref}) and velocity ($\dot{q}^{ref}$) usually compose the state ($\mathbf{x} \in \mathbb{R}^n$). The future contact points can

be precomputed by a **Motion Planner** or included in the state of the problem. The control of this system $\mathbf{u} \in \mathbb{R}^m$, can be the robot generalized acceleration ($\ddot{q}^{ref}$), the contact wrench (ϕ_k with $k \in \{0, \ldots, \text{Number of Contact}\}$), or the motor torques (τ^{ref}). We denote by $\underline{\mathbf{x}}$ and $\underline{\mathbf{u}}$ the state and control trajectories. Being more precise than in eq. 1.3, the following optimal control problem (OCP) represents a generic form of the locomotion problem (which can be for instance a direct multiple shooting problem):

$$\min_{\underline{\mathbf{x}},\, \underline{\mathbf{u}}} \sum_{s=1}^{S} \int_{t_s}^{t_s+\Delta t_s} \ell_s(\mathbf{x}, \mathbf{u})\, \mathbf{dt} \tag{3.1a}$$

$$s.t. \quad \forall t \quad \dot{\mathbf{x}} = dyn(\mathbf{x}, \mathbf{u}) \tag{3.1b}$$

$$\forall t \quad \phi \in \mathcal{K} \tag{3.1c}$$

$$\forall t \quad \mathbf{x} \in \mathcal{B}_x \subset \mathbb{R}^n \tag{3.1d}$$

$$\forall t \quad \mathbf{u} \in \mathcal{B}_u \subset \mathbb{R}^m \tag{3.1e}$$

$$\mathbf{x}(0) = \mathbf{x}_0 \tag{3.1f}$$

$$\mathbf{x}(T) \in \mathcal{X}_* \subset \mathbb{R}^n \tag{3.1g}$$

where $t_{s+1} = t_s + \Delta t_s$ is the starting time of the phase s (with $t_0 = 0$ and $t_S = T$). In the direct multiple shooting problem a phase s corresponds to an interval where the system is simulated using constraint (3.1b) which makes sure that the motion is dynamically consistent. Phases are connected through the constraints (3.1d) and (3.1e) which impose bounds on the state and the control. These are lying respectively in admissible set of states $\mathcal{B}_x$ and in admissible set of controls $\mathcal{B}_u$. Constraint (3.1c) enforces balance with respect to the contact model. Breaking and adding contacts is usually done at phase junctions because it changes the structure of the dynamics. Constraint (3.1f) imposes the trajectory to start from a given state (estimated by the sensor of the real robot). Constraint (3.1g) imposes the terminal state to be in the viable terminal states set $\mathcal{X}_*$ [Wieber 2008]. The cost (3.1a) is decoupled $\ell_s(\mathbf{x}, \mathbf{u}) = \ell_x(\mathbf{x}) + \ell_u(\mathbf{u})$ and its parameters may vary depending on the phase. ℓ_x is generally used to regularize and to smooth the state trajectory while ℓ_u tends to minimize the forces. The resulting control is stable as soon as ℓ_x comprehends the L_2 norm of the first order derivative of the robot center of mass (CoM), [Wieber 2015]. Problem (3.1) is difficult to solve in its generic form. And specifically (3.1b) is a challenging constraint. Most of the time the shape of the problem varies from one solver to another one only by the formulation of this constraint. The difficulty is due to two main factors: 1) There is a large number of degrees of freedom (DoF). In practice we need to compute 36 DoF for the robot on a preview window with 320 iterations ($1.6s$) to take into account the system inertia. 2) The dynamics of the system is nonlinear. Fig. 3.2 depicts the structure of the problem. To be able to solve the whole problem, represented by the full rectangle (blue plus orange and green) in Fig. 3.2 researchers often use nonlinear optimization. In this work we evaluated a resolution of the **MPWBC** based on the formulation

given by Eq. 3.1. In this approach described in [Koch 2014], the authors computed a dynamical step-over motion with the HRP-2 robot, but this process can take several hours of computation. So simplifications are necessary, for example [Tassa 2014], [Koenemann 2015] use simplifications on the contact model. This method is very efficient but not suitable for complex contacts during walking. Seminal works ([Orin 2013],[Kajita 2003b]) show that (3.1b) can be divided into two parts, the non-convex centroidal dynamics (orange horizontal rectangle in Fig. 3.2) ([Orin 2013]) that includes few DoF, and the convex joint dynamics (vertical rectangle in Fig. 3.2). [Kuindersma 2014] and [Sherikov 2016] chose to deal the two aforementioned parts of Fig. 3.2 at once. They optimize for the centroidal momentum on a preview horizon and the next whole body control. [Qiu 2011], [Rotella 2015], [Perrin 2015] decouple the two separated aforementioned rectangles in Fig. 3.2. They solve first for the centroidal momentum and then for the whole body control. In general the centroidal momentum remains difficult to handle due to its non-convexity. Finally [Kajita 2003a], [Herdt 2010a], [Sherikov 2014] linearize the centroidal momentum. This provides a convex formulation of the locomotion problem. In [Deits 2014], the problem is formulated as a mixed-integer program (i.e. having both continuous and discrete variables) in case of flat contact. In [Mordatch 2012], the same problem is handled using a dedicated solver relying on a continuation heuristic, and used to animate the motion of virtual avatars.

3.2.2 Benchmarking

Different methods exist to benchmark robot control architectures. In [Del Pobil 2006] the authors argue that robotic challenges offer an efficient way to do so. For example, the results of the DARPA Robotics Challenge published in the Journal of Field Robotics special issues [Iagnemma 2015] and [Spenko 2017], show the different control architectures in a given context. Each behavior successfully accomplished grants point to the team and the best team wins the challenge. This benchmarking was however costly as the robots had no system to support them in case of fall. In addition, as it is mostly application driven, the challenge provides an overall evaluation of the system integration but not of the independent sub-parts.

For the specific case of motion generation, it has been recently proposed by [Brandao 2017] to use a scenario called "Disaster Scenario Dataset". It allows benchmarking posture generation (solved by the **WBC**) and trajectory generation (**MPWBC**) using optimization. A set of problems is proposed by means of foot step locations (F_{RF}, F_{LF}). Using this approach, it is possible to compare algorithms realizing the two functionalities (**WBC** and **MPWBC**). The evaluation is realized in simulation using the Atlas robot and the ODE dynamic simulator. This first step is necessary but one step further is required to benchmark a real humanoid platform. For this work we used a more systematic decomposition of the humanoid bipedal locomotion [Torricelli 2015]. Further description can be found in paragraph 3.3.7. This work focuses on evaluating the **MPWBC** and **WBC** on the **Robot Hardware**. The **Estimator** used in this context is important but it is reflected

in the stabilization process. The **Motion Planning** is not evaluated here as the planned motion is always the same or solved at the **MPWBC** level. The **Localization** is provided by a motion capture system.

3.2.3 A motivating example: the Koroibot project

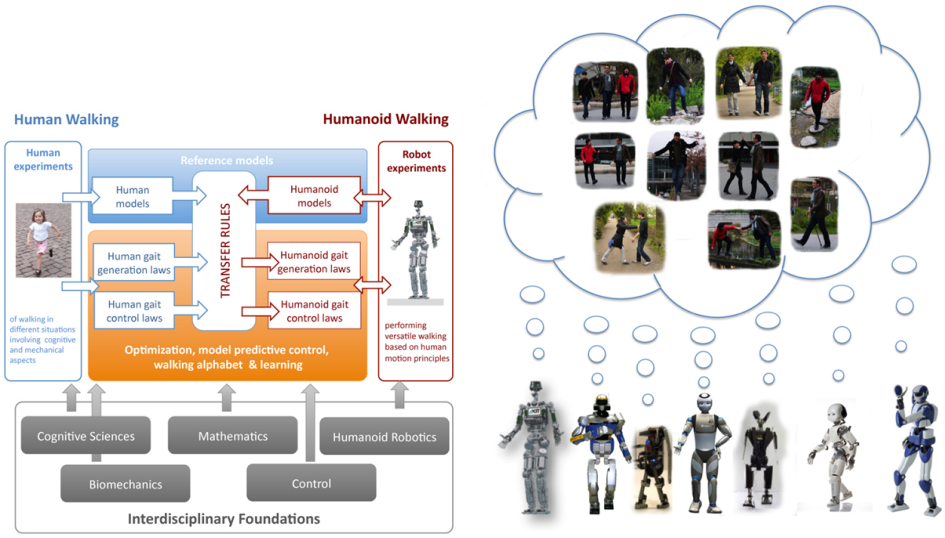


Figure 3.3: (left) Graphical representation of the scientific approach of the Koroibot project - (right) View of the humanoid robot used in the Koroibot project dreaming of human walking capabilities

The work presented here takes its root in the context of the European project Koroibot (`http://www.koroibot.eu/`). The goal of the Koroibot project was to enhance the ability of humanoid robots to walk in a dynamic and versatile way, and to bring them closer to human capabilities. The Koroibot project partners had to study human motions and to use this knowledge to control humanoid robots via optimal control methods. Human motions were recorded with motion capture systems and stored in an open source data base which can be found at `https://koroibot-motion-database.humanoids.kit.edu/`. With these data several possibilities were exploited:

- Criteria that humans are assumed to minimize using Inverse Optimal Control.
- Transfer from human behaviors to robots given by walking alphabets and learning methods [Mandery 2016].
- Human behaviors safely integrated in robots by means of optimal controllers.
- Design principles derived for new humanoid robots. [Mukovskiy 2017, Clever 2017]

To evaluate the progress of the algorithms at the beginning and at the end of the project, a set of challenges focusing specifically on walking were designed (see

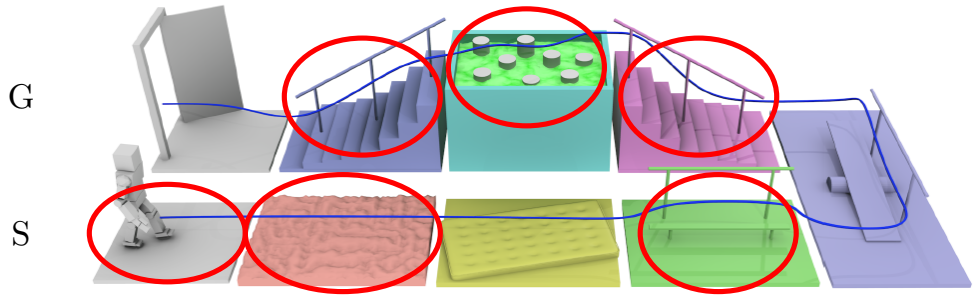


Figure 3.4: Challenges of the Koroibot project. In red the challenges chosen by Gepetto team at LAAS-CNRS.

Fig. 3.4). Fig. 3.3-(right) shows all the robots hosted by the Koroibot partners. Each team owning a robot had to perform some of these challenges considering the current and the potential state of their robots and controllers.

3.2.4 The Key Performance Indicators (KPI)

In this context and in collaboration with the H2R project, a detailed set of key performance indicators (KPI) have been proposed [Torricelli 2015]. These KPI try to capture all the bipedal locomotion patterns. Specific sub-functions of the global motor behaviors were analyzed (see Fig. 3.5-(right)). The results are expressed as two different sub-function sets. First, the sub-functions associated with the body posture task without locomotion. Second, the same sub-functions but including the robot body transport. The initial condition may vary depending on the experiment to perform. This is the idea of the intertrial variability. The sub-functions are also classified by taking into account the changes in the environment or not. Each of these functions can be evaluated for different robots using the criteria depicted in Fig. 3.5-(left). The performances are classified into two sub categories, quantitative performances and human likeness. In addition, information in the last two columns indicates whether the criteria is applicable on a standing task or on a locomotion task. Again, all the teams owning a robot had to perform an evaluation of these KPI, considering the current and potential state of their robots and controllers.

3.2.5 The work done in the Koroibot context

In the Koroibot context the Gepetto team evaluated the KPI one the robot HRP-2 (second robot from the left in (Fig. 3.3-(right)). Among the challenges presented in Fig. 3.4, we considered the following ones:

- walking on a flat ground,

		Abilities		Benchmarks		
		Name	Description	Benchmark	Applicability Posture	Applicability Transport
Performance	Stability	Intratrial Stability	Ability to Maintain Equilibrium Within a Single Trial	Time Until Falling	X	
				Cycles Until Falling	X	X
		Intertrial Stability	Ability to Maintain the Equilibrium Across Different Trials	Success Rate Across *N* Different Trials	X	X
		Gross Body Equilibrium	Ability to Maintain Equilibrium Over the Base of Support	Energy Stability Margin (ESM)	X	
				Maximum Accepted Disturbance Amplitude	X	X
				Maximum Accepted Disturbance Frequency	X	X
	Efficiency	Global Energy Consumption	Ability to Transport Body with Low Energetic Costs	Specific Energetic cost of Transport C_{et}		X
				Specific Mechanical Cost of Transport C_{mt}		X
		Passivity	Ability to Minimize Joint Torques During Walking	Passive Gait Measure (PGM)		X
		Reaction Time	Ability to Promptly React to Disturbance or External Command	Time from Input and Initiation of Motor Action	X	X
Human Likeness	Kinematics	Gross Body Motion	Motion of the Whole Body Expressed by Global Variables	CoM Trajectory (Correlation, Dynamic Time Warping)	X	X
				Gait Harmony		X
				Body Sway (Frequency Response Function)	X	
				Natural Looking Motion	X	X
		Individual Joint Motion	Motion of the Single Joints or Limbs Taken Separately	Joint Trajectory (Correlation, Dynamic Time Warping)	X	X
				Knee, Ankle Forefoot Rocker		X
		Interlimb Coordination	Ability to Coordinate Between Different Body Parts	Symmetry (Ratio Index)	X	X
				Trunk/Arm Motion	X	X
		Intralimb Coordination	Ability to Move Multiple Joints of the Same Limb Coordinately	Kinematic Synergies	X	X
	Dynamics	Gross Body Kinetics	Forces Exerted Between the Whole Body and the Environment	Ground Reaction Forces (Correlation, Dynamic Time Warping)	X	X
		Single-Joint Kinetics	Force Exerted Among Limbs	Joint Torques (Correlation, Dynamic Time Warping)	X	X
		Dynamic Similarity	Ability of Having Leg Pattern Dynamically Similar to Most Legged Animals	Froude Number (Dimensionless Gait Velocity)		X
		Dynamicity	Ability to Use Falling State for Body Progression	Dynamic Gait Measure (DGM)		X
		External Compliance	Ability to Respond Resiliently to External Disturbances	Impulse Response Function (IRF)	X	X
		Internal Compliance	Ability to Store and Release Energy	Active/Net Joint Torque	X	X

			Function	
			Body Posture	Body Transport
Environment	Stationary	Intertrial Variability: No	Static Horizontal Surface	Horizontal Ground at Constant Speed
			Static Inclined Surface	Sloped Ground
				Stairs
		Intertrial Variability: Yes	Different Static Surfaces	Variable Slopes
				Irregular Terrain
				Slippery Surface
	In Motion	Intertrial Variability: No	Continuous Surface Tilts	Treadmill at Constant Speed
			Continuous Surface Translations	Soft Terrain with Constant Compliance
			Bearing Constant Weight	Bearing Constant Weight
			Pushes	Pushes
		Intertrial Variability: Yes	Sudden or Pseudorandom Surface Tilts	Treadmill at Variable Speed (Including Start-Stop)
			Sudden Surface Translations	Seesaw
			Body Sway Referenced Platform (BSRP)	Soft Ground with Variable Compliance

Figure 3.5: (left) performances indicators, (right) motor skills considered in the benchmarking scheme. This scheme is limited to bipedal locomotion skills. The concept of intertrial variability represents modifications of the environment between trials. (dashed) motor skills evaluated in [Naveau 2016] (not dashed) motor skills evaluated in this work.

- walking on an uneven ground,
- walking on a beam without handrail,
- climbing a stair case with/without handrail,
- walking on stepping stones,
- going down a stair case without handrail,

They are depicted by red circles in Fig. 3.4. In addition to these challenges we added the perturbation rejection. Considering the selected challenges we picked the following KPI:

- horizontal ground at constant speed,
- stairs,
- bearing constant weight (the robot's own weight)

while considering the following motor-skills:

- success rate across N different trials,
- mechanical energy,
- mechanical plus electrical energy,

All these choices are shown in Fig. 3.5 by red ellipses in the table. The mathematical details and results are presented below in paragraph 3.3.7.

In order to test all the motions in practical situations and point out more possible indicators and side effects that can be benchmarked , tests have been provided to make the robot the whole Koroibot plateform that Gepetto team bought.

3.2.5.1 Gepetto's platform

The Gepetto team bought a platform composed of stairs (10 and 15cm), a hand rail next to 15cm stairs and a 3m beam. As mentioned, several algorithms have been used to overpass these obstacles. The platform is depicted in Fig. 3.6

One objective was to close the KoroiBot loop by playing all the algorithms produced during the project without any human interaction. The description of the platform is more relevant by knowing the direction (clockwise or counter clockwise) that the robot has to follow. Indeed the algorithms are not the same depending on that direction on one obstacle. We did not used the same algorithms either the robot is going up or downstairs: going upstairs on 15cm stairs is realized with handling the handrail, so then we can't use the Kawada's stabilizer contrary to the motion going down that does not use the handrail. Depending of the generated motions and their repeatability, it was decided to turn clockwise as a first trial. The starting point was the end of the 15cm stairs on the right of Fig. 3.6, on flat ground. Then, the robot should have reached the 10cm stairs on flat ground, going upstairs (six 10cm stairs), turning to the beam, crossing it, turning again to 15cm stairs and going downstairs (four 15cm stairs), then repeat the complete circuit.

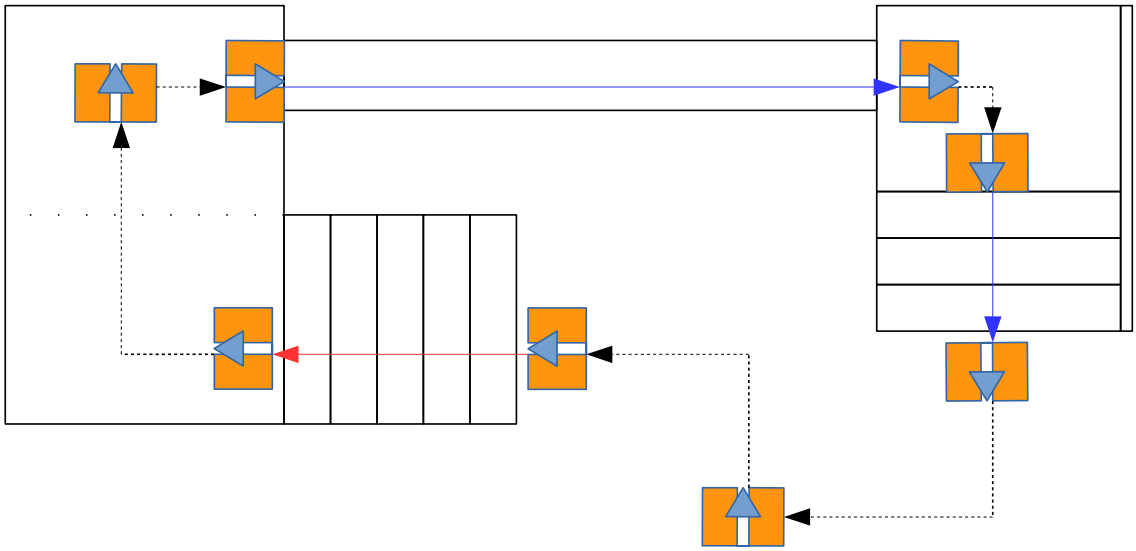


Figure 3.6: Upper view of the platform bought for the KoroiBot project by Gepetto. Orange rectangles with big light blue arrows are the position of the robot to be corrected before the next offline-computed sequence of steps. Dashed black arrows represent the motions generated for connecting two obstacles. Red and blue arrows are prior motions generated to cross the KoroiBot challenges.

Since the obstacle motions were already generated, the missing part was to generate the walk between each of them. Even if the model of the platform is known and most of theses connecting motions have been calculated offline, the robot position should be corrected before a sequence of steps (crossing an obstacle or going to the next obstacle). So then, a exteroceptive sensor feedback is needed; we chose the motion capture sensor to close this feedback loop.

3.2.5.2 Motion capture loop

The motions provided during the KoroiBot challenge by the Gepetto team are computed offline. That means the initial position of the robot is crucial to be able to cross the obstacles. Specifically, a good position of the feet is necessary to execute correctly the motion. Algorithms used to cross the obstacles do not have a feedback loop to correct them in case the robot is not well placed on the ground beforehand. Launching a motion for a demonstration needs sometimes two or three trials for the operator to put correctly the robot on the ground. Positions of the feet is the key point to have a good result. To close the KoroiBot loop, we need to place the robot well enough in front of the obstacles (stairs and beam). As a first approach, we decided to use the motion capture (MotionSys) of the Bauzil room (Gepetto team experimental room). The motion capture is composed of active cameras that send infra red in the scene that bounces of reflective balls (called reflectors) placed in the scene. Since there are multiple cameras and points of view, the positions of the

reflectors are reconstructed.

3.2.5.3 Motion capture

During these experiments, our motion capture system was composed of ten cameras. Since some cameras have been out of order because of the wear effects, spare cameras from different providers have been introduced in the set of our cameras; three types of camera composed this set. The system needs a calibration before being used: the cameras has to be placed in order to see the largest section of the operational working volume, taking into account that at least three cameras should be able to see a reflector to detect it correctly. Then the focus of the cameras has to be adjusted for the working distance. Finally, two patterns must be used to calibrate an origin frame and to know how the cameras are placed with respect to that frame. This process is commonly realized by computing the maximum likelihood estimation of these positions. The volume parts that are the least covered by the cameras can be affected by detection or precision issues. The different camera types are also source of imprecision.

Information given from motion capture is position and orientation of a pattern composed of several reflectors. In our case, this pattern was made with four reflectors placed on the left foot of HRP-2 as shown in Fig 3.7.

Figure 3.7: Reflectors on the left of HRP-2 form the pattern to be detected by motion capture

Motion capture information is sent from a dedicated computer through ROS middleware. To get the reference positions, the operator places the robot in front of the obstacle correctly and the position is saved. During the experiment, the motion capture system measures the robot position and compares them to the reference position in order to generate the corrective motion.

3.2.5.4 Feedback loop

The goal is to connect two Koroibot challenges (final position of going downstairs 15cm to the 10cm stairs to climb up for example). I generated the main part of the trajectory offline with the model of the platform (the stairs) with a Pattern Generator available in the team (1.3.4). When this offline computed motion is played on the robot, errors of placement remains and the robot is not placed well enough to fulfill the next challenge. Here comes the feedback loop with the motion capture. The goal is to make the robot move to the position allowing it to achieve the next challenge (climbing stairs or crossing the beam)

The correction is generated in two stages. The first one provides foot steps that correspond to the error correction. I made an ad hoc foot step planner that splits total length of the correction into minimum equal steps length with respect to reachability limits. Angular correction was taken into account and executed on the last steps of the motion. This step planner is a heuristic developed for a low price considering that other colleagues from the Gepetto team were in progress to provide a much more efficient planner: [Tonneau 2018].

The second stage mentioned is the whole body trajectory generator. Two possibilities were available : [Naveau 2017]'s algorithm detailed in subsection 1.3.6 or [Morisawa 2007]'s algorithm developed in subsection 1.3.4. The first one takes CoM velocity as reference that can be changed online. This makes it interesting for error compensation as we were trying to achieve online behaviors. The main problem lies in the position of the feet. This algorithm chooses its own foot positions. This means that the robot would have never been correctly positioned for executing the next precomputed motions. [Morisawa 2007]'s algorithm does need the foot placements. But as a counterpart, it is not online and our implementation needs a specific posture at the beginning and at the end of the motion. We opted for this one, which corresponds to our constraints.

To go over this loop, the motion capture gets the position of the left foot of HRP-2 from a dedicated computer (virtual machine) that sends information via ROS. Another computer compares this position to the reference and generates foot steps for the correction. These foot steps are given to the [Morisawa 2007]'s Gepetto implementation that generates the joint trajectories. These trajectories are then sent to the robot to be played. For playing these trajectories, references go through Stack of Tasks (evoked in Chapter 1) to add the stabilizer during the run.

3.2.6 Repeatability issue

The motions needed between the different challenges were all generated. Each of them has been tested to start the next challenge with success. The main issue happened for crossing the beam. Even though the robot succeeded to cross the beam several times, the repeatability was not sufficient. I created limits on position and orientation to check if the robot would be able to go through the beam or not. If

not, the correction procedure was made again until the position was correct enough (in the position and orientation limits). In the case of crossing the beam, several repetitions (between three and seven) were needed because the corrected orientation of the robot was not acceptable to let it cross the beam. This challenge is the most demanding in orientation placement. Because this number of attempts was not satisfying, the placement precision of the robot feet were checked on one step motions. As mentioned in subsection 2.3.4, depending on the calibration of the robot, an error from $5mm$ to $12mm$ was observed on [Kajita 2003a] and [Morisawa 2007]'s algorithms. These errors occurred even after the Kawada's calibration. This seems to come from the mechanics wear or the stabilizer effects. So then, we decided to stop investigations in that way. No publication was provided for this work.

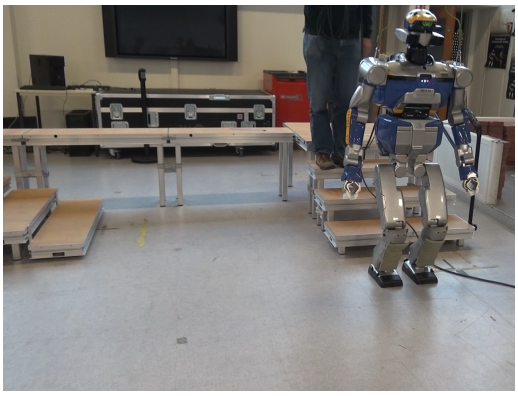
3.2.6.1 Motions

At the point we stopped, all the motions were provided. From the operator point of view, the motions were not really good. The feet were touching the ground too early, provocating bumps when reaching the ground, there was drift of several centimeters on few steps. The stabilizer acted as expected preventing the robot to fall. For offline motions drift was so important and balance so uncertain that I cut some parts in pieces with corrections between. It is the case for each feet placement in orange on Fig. 3.6 that connects two black dashed lines (on flat ground between 15 and $10cm$; between $10cm$ stairs and beam).
Fig. 3.8 presents pictures of the realized motions.

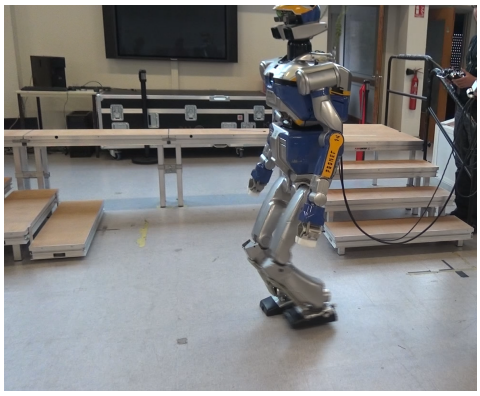
3.2.6.2 Perspectives

For the tests, repeatability of the robot was a problem, either coming from calibration or controller effects. The calibration of the motion capture was exhausting and we suspect non homogeneous precision over all the KoroiBot platform. Correcting online would have meant to rely on [Naveau 2017]'s algorithm and provide a sensor feedback in it. In addition intersection of constraints (foot reachabillity on stairs would lay on two stairs for one time step) leading to mixed integer problems formulation would have been necessary. The motion capture has been changed with more cameras, that allows a better coverage of the working volume.

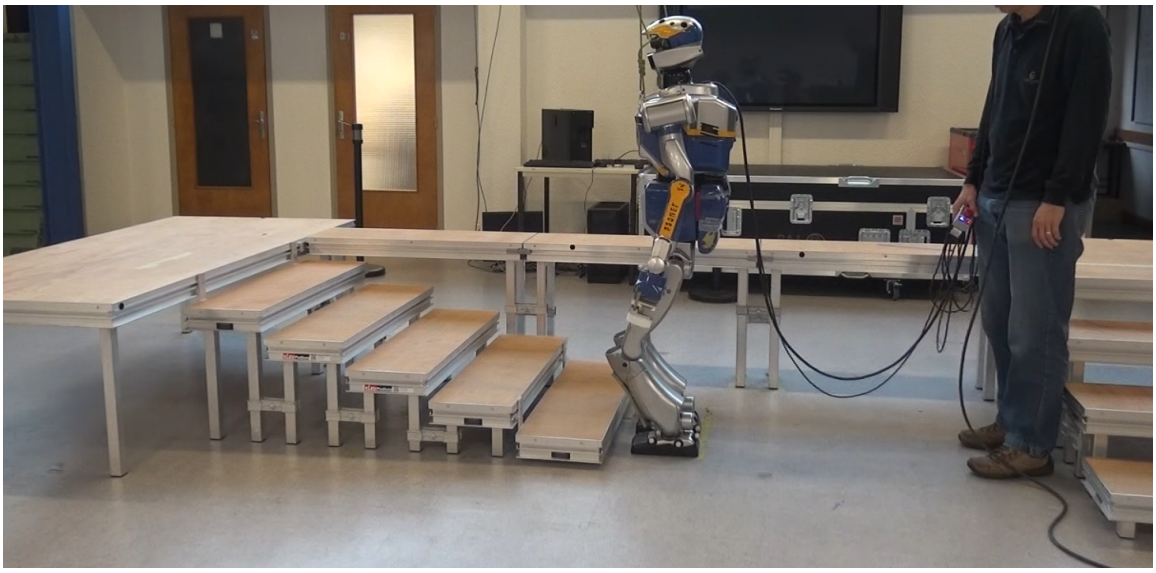
More academically, the most significant element to improve is the feedback reaction during the motions crossing the obstacles. In that case, we could use a more reliable pattern and whole body generator as [Naveau 2017]. This reinforces our prospective to use visual information at whole body joint trajectory generator to react to the environment or to the drift of the robot. Otherwise, as described in [Tonneau 2019], the decoupling approach seen in Chapter 1 (splitting the locomotion problem in smaller sub-problem) is now converging towards satisfying results.



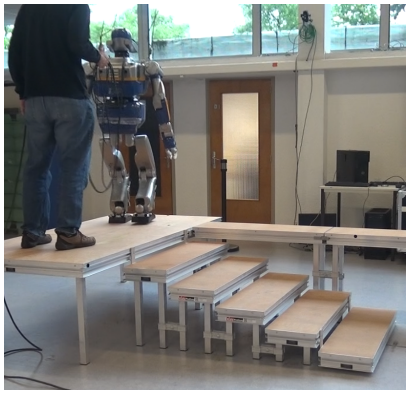
(a)



(b)



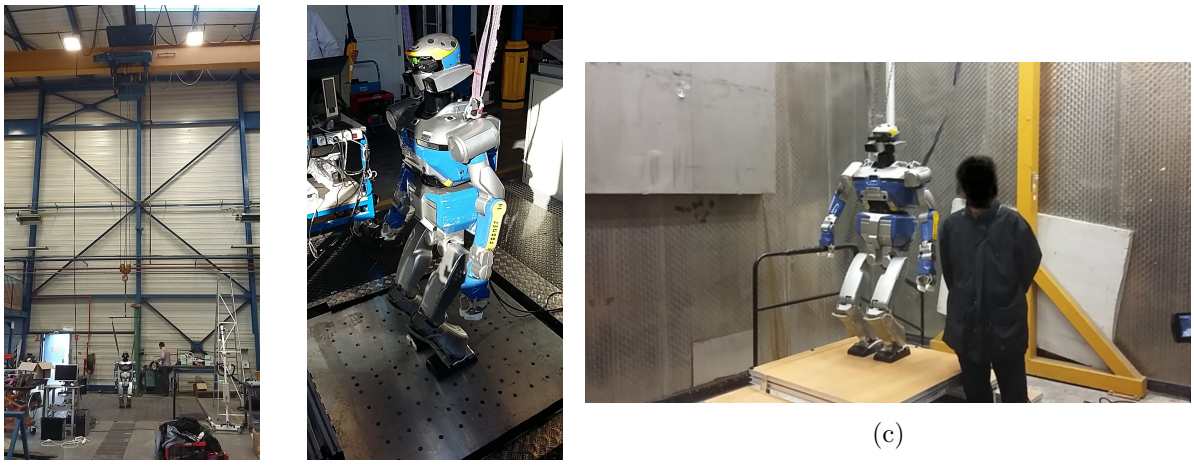
(c)



(d)

(e)

Figure 3.8: HRP-2 reaching different challenges of the KoroiBot using motion capture loop.



(a) (b) (c)

Figure 3.9: Pictures of the experimental setup at LNE (a) the robot hang up to walk on a slope (b) the translational plate (c) the temperature-controlled chamber (end of the robot climbing 15 cm at $10°C$)

3.3 Materials and Methods

In order to comprehend such a phenomena, described above, and to compare different abilities of the humanoid robots, the experimental setups used to compute each of the performance indicators given in 3.3.7 are described in this section. The motor skills given in Fig. 3.5 and their implementation are also presented. In addition, the algorithms used to perform the different tests are depicted in paragraph 3.3.8.

3.3.1 Different temperatures

The LNE (Laboratoire Nationale de Métrologie et d'essais) is equipped with temperature-varying rooms which allowed us to measure some of the performance indicators at various temperatures ranging from $5°C$ to $45°C$. In this way, we evaluated the robustness and limits of our robot with respect to the performance indicators in different environmental conditions. It appeared that the robot behavior deteriorates at low temperatures. At $5°C$ it is not possible to perform the calibration procedure as the robot could not move. At $10°C$ the friction is sufficiently low such that the robot could move. Another phenomenon occurs above $40°C$ after few motions due to internal temperature build up: thermal protection prevents the robot from moving if the temperature is too high. In this room, apart from these extreme cases, the motions and indicators measurements have been performed as expected on a flat ground or on the staircase testbed of the Koroibot project. This staircase is made of 4 15 cm high stairs and a top platform. The dimension of one stair case is $1\ m \times 0.25\ m \times 0.05\ m$.

3.3.2 Tilted surfaces

In the context of the body skills in motion, we considered tilting surfaces. This was tested with the stabilizer commercially available with HRP-2. The setup is a platform which can be tilted upward and downward on one side with a hydraulic actuator. The surface was tilted continuously until the robot fell off. On the other hand, we tested walking algorithms with different angles (pointing up or down) until the robot fell down. Tests were realized with the robot pointing down, pointing up and across the slope. In Fig. 3.5 this test corresponds to Body Posture - Continuous Surface Tilts.

3.3.3 Horizontal translations

We used a mobile plate controlled in the horizontal plane to perform continuous oscillating surface translations at various frequencies and various amplitudes. The platform was moved by a hydraulic actuator. The aim was to find the frequency and the amplitude that the controlled robot is able to sustain with the available stabilizers. In Fig. 3.5 this test corresponds to Body Posture - Continuous Surface translations.

3.3.4 Bearing

In order to test the robot capability to bear weights, we loaded it with additional masses (bags of 5 kgs to 15 kgs) in such way that its balance is maintained. This approach is a bit limited as there are several ways to bear a weight. Indeed it can be done with a backpack, in collaboration with someone, or by holding the object against its chest. Each approach comes with its own specific constraint. In order to avoid such constraints, we decided to take the simplest choice and hang soft weights on the front and the back of the robot chest. In Fig. 3.5 this test corresponds to Body Transport - Bearing Constant Weight.

3.3.5 Pushes

This paragraph presents the pushes experiments. We tried to find the sufficient force to make the robot fall down. This was achieved by using a stick on top of which was fixed a force sensor displaying the maximum force measured during an experiment. The sensor used was a HBM 1000 N of type u3 together with a HBM Scout 55 amplifier. The experience was realized while the robot was standing and walking. The force was applied in the sagittal and frontal planes until making HRP-2 fall. The force was applied from behind the waist of the robot. This part of HRP-2 was made specifically soft to support impacts. The walking part is the most difficult in terms of repeatability as the robot might be in different foot supports and therefore more or less stable depending on the configuration. In Fig. 3.5 this test corresponds to Body Posture - Pushes and Body Transport - Pushes.

3.3.6 Data

A CAD model of the staircase used is available on the github repository where all the log of the experiments are also present: https://github.com/laas/koroibot_KPI. All the computations performed on the logs and implementing the key performance indicators are available here: https://github.com/laas/EnergyComputation.

3.3.7 Key Performance indicators (KPI)

In this section the performance indicators used to evaluate the humanoid robot HRP-2 are described. They are mostly based on the work proposed in [Torricelli 2015]. In the KoroiBot project we used key performance indicators (KPI) to analyze the

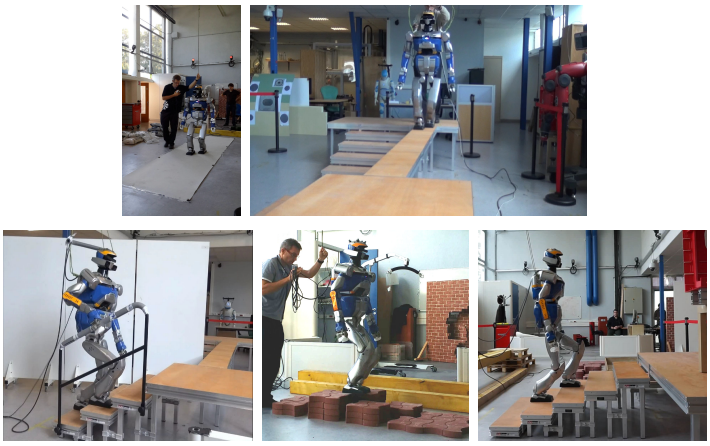


Figure 3.10: Sample of the experimental setup of the KoroiBot project in LAAS-CNRS

behavior of the robot at the beginning and at the end of the project. These results lead us toward the improvements to be made. In 2013 the algorithm mostly used and implemented on HRP-2 in LAAS-CNRS where the walking pattern generators described in [Morisawa 2007] and in [Herdt 2010a]. The performance indicators chosen were:

- The execution time $T_M = t_{end} - t_{begin}$, where t_{begin} is the time at which the sum of the norm of the motor axis velocities reaches $6\,rad\,s^{-1}$ for the first time in the log and t_{end} is when the sum of the norm of the motor axis velocities passes below $0.5\,rad\,s^{-1}$.
- The walked distance, being the distance between the final base position and the initial one. The base pose is reconstructed using odometry with the joint positions only. The drift of this odometry is $8cm$ over $3.6m$ during a straight walk.
- The success rate, being the number of time a specific task could be performed without falling, over the total number of trials of the task.

- The maximum tracking error from the planned trajectory,

$$TrackingError(t) = \int_{t}^{t+0.1} |q^{ref} - \tilde{q}|dt/0.1$$
$$MaxTrackingError = \max_{t}(TrackingError(t))$$

with $TrackingError$ being the average normed difference between the desired joint trajectory (q^{ref}) and the joint pose measured from the encoder ($\tilde{q}$) during $0.1s$ starting at time t. And $MaxTrackingError$ being the maximum value of the $TrackingError$ function.

- The mechanical energy consumed normalized over the walking distance D and the execution time T_M.

$$E_{mechanical} = \int_{t_{begin}}^{t_{end}} |\tau\omega|dt/(T_M\ D)$$

with $E_{mechanical}$ being the integral over time of the mechanical power, τ being the torques applied at the robot joints and ω being the velocity of the robot joints.

- The electrical energy dissipated by the motor resistance normalized over the walking distance D and the execution time T_M,

$$E_{motor\,resistance} = \int_{t_{begin}}^{t_{end}} R\ i^2 dt/(T_M\ D) = \int_{t_{begin}}^{t_{end}} R\ k_c^2\ \tau^2 dt/(T_M\ D)$$

with $E_{motor\,resistance}$ being the integral over time of the electric power dissipated, R being the motor resistances, k_c being the electric motor torque constant and τ being again the torques applied at the robot joints.

- The total energy consumed during the walking distance D and the execution time T_M,

$$E_{total} = E_{mechanical} + E_{motor\,resistance} + E_{electronics}$$

with E_{total} being the sum of the energy consumed by the system normalized over the walking distance D and the execution time T_M, and $E_{electronics}$ being the energy consumed by the on-board electronic cards. $E_{electronics}$ is neglected in this study so:

$$E_{total} = E_{mechanical} + E_{motor\,resistance}$$

- The mechanical cost of transport and the total cost of transport,

$$E_{mechanical\,cost\,transport} = \int_{t_{begin}}^{t_{end}} |\tau\omega| dt/(m\ g\ D)$$

$$E_{total\,cost\,transport} = \left(\int_{t_{begin}}^{t_{end}} |\tau\omega| dt + \int_{t_{begin}}^{t_{end}} R\ k_c^2\ \tau^2 dt \right) /(m\ g\ D)$$

  with $E_{mechanical\,cost\,transport}$ and $E_{total\,cost\,transport}$ being respectively the mechanical and total cost of transport, m being the total mass of the robot, and $g = 9.81ms^{-2}$ the gravity constant.

- The Froude number,

$$F_r = \frac{v}{\sqrt{gl}}$$

$$v = \frac{D}{T_M}$$

  where v is the robot center of mass mean velocity along the horizontal plane and l is the leg length. This number represents the ratio between the kinetic energy and the potential energy. It can also be interpreted as an indicator on the stepping frequency.

The trajectories were generated off line and repeatedly played on the robot to analyze their robustness. Views of the experimental setups are given in Fig.3.10.

3.3.8 Motion generation for humanoid robot locomotion

This section explains the links between the motion generation architecture depicted in Fig.3.1 and the Key Performance Indicators given in the paragraph 3.3.7. The set of functions entitled body posture, depicted in Fig.3.1-(right), represents the behavior which is provided by what is called a whole-body controller. It consists of two parts:

- an estimator, which provides the orientation of the robot with respect to the gravity field and the positions of the end-effectors in contact with the environment.
- a whole-body controller which guarantees that the robot balance is maintained with respect to c^{ref}, f^{ref} and possibly a q^{ref}.

In this work we have evaluated independently only one whole body motion controller. It is the stabilizer provided by Kawada Inc. We give detailed performances evaluation of this controller in the experimental part of this work. It was described in various work such as [Kajita 2007] and [Kajita 2001].

The set of functions entitled body transport, depicted in Fig.3.1-(right) in this work, are four **CDPG** and one **MPWBC**. The four **CDPG** evaluated in this work are the following ones: [Carpentier 2016], a multi-contact centroidal dynamic

pattern generator used to climb stairs with given contact positions, [Kajita 2003a], the original walking pattern generator implemented by Shuuji Kajita with given foot steps, [Morisawa 2007], an analytical walking pattern generator allowing immediate foot step modifications, [Naveau 2017], a real time nonlinear pattern generator able to decide autonomously foot-steps positions. In each case the goal of the **CDPG** is to generate a center of mass trajectory and the foot-steps trajectories. For [Kajita 2003a], [Naveau 2017], and [Morisawa 2007] a dynamical filter is used to correct the center of mass trajectory to improve the dynamical consistency of the motion. In each case, a whole body motion generator (not to be confused with a whole body motion controller) is used without feedback to generate the reference position q^{ref}, and the desired z^{ref} which are then sent to the stabilizer. For [Naveau 2017] and [Morisawa 2007] we used the stack of tasks described in [Mansard 2009] as a Generalized Inverse Kinematics scheme. In [Carpentier 2016] a Generalized Inverse Dynamics was used to generate the reference value for q^{ref} and c^{ref}. The **MPWBC** provides the controls directly. The one used is from [Koch 2014] using the Muscod-II [Diehl 2001] nonlinear solver.

3.4 Results

In this paragraph we present the numerical results obtained from the computation of the KPI explained in detail in paragraph 3.3.7 for each set of experiments. As a reminder the list of the KPI is recalled:

- walked distance,
- success rate,
- max tracking error,
- duration of the experiment,
- mechanical joint energy,
- actuators energy,
- cost of transport,
- mechanical cost of transport,
- Froude number.

A video displaying a mosaic of all the experiments is available at the following URL: https://www.youtube.com/watch?v=djWGsb44JmY&feature=youtu.be.

3.4.1 Climbing stairs

3.4.1.1 Stairs of 10 cm

In this experiment, the humanoid robot HRP-2 is climbing stairs of 10 cm height without any handrail. The difficulty of this task is that the robot has to perform quite large steps and vertical motion. For this reason, the robot is climbing one

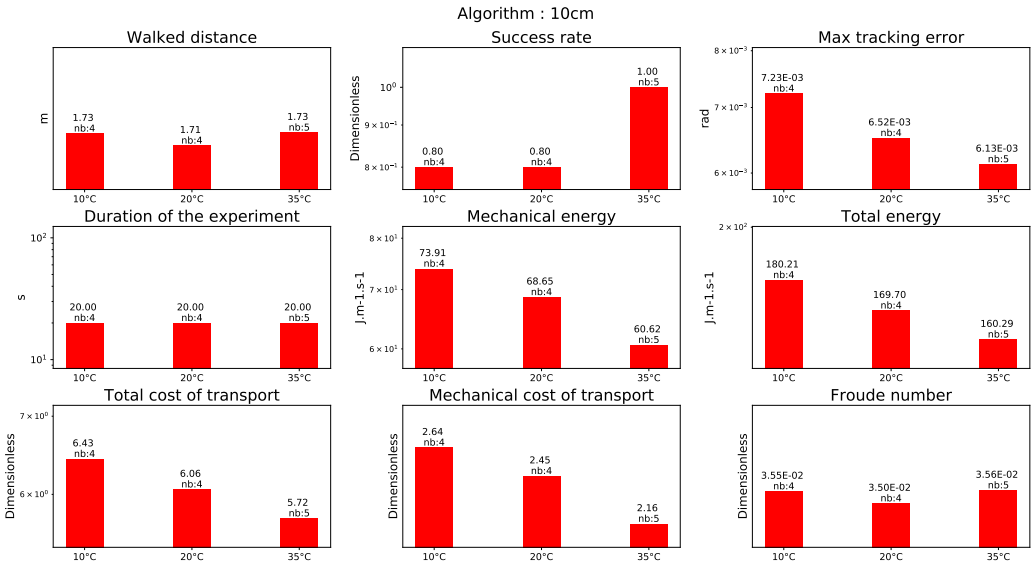


Figure 3.11: Climbing 10 cm stairs without handrail

stair at a time, which means that the robot puts successively one foot on the next stair and the other one on the same stair. This avoids a too large joint velocity that the robot could not track. [Morisawa 2007] **CDPG** was evaluated at the beginning of the project although the variation of height violates the assumption of the cart-table model. But thanks to the dynamical filter the motion generated was dynamically consistent so that the stabilizer could cope with the situation. Because this experiment was not performed at the LNE (it was done three years before) it was not possible to control carefully the room temperature but the test was performed at $20°C$. The KPI results can be seen in Fig. 3.14-(tool upstairs). The other test was performed at the end of the project using the **CDPG** [Carpentier 2016]. This time the **CDPG** took into account the center of mass height variation but not the whole body motion. The stabilizer should theoretically have less trouble to compensate for the simplifications made. For [Carpentier 2016] three different temperatures were tested: $10°C$, $20°C$ and $35°C$. The numerical results are depicted in Fig. 3.11. Interestingly, the temperature level has a direct impact in terms of mechanical cost as it diminishes with the increase in temperature. It is reflected in the tracking error. This intertrial variation does not come from the change of reference trajectory as it is strictly the same for every trial. There is a level of adaptation due to the stabilizer, but each temperature has been tested at least 4 times. A possible explanation is the fact that the grease in the harmonic drives generates less friction at higher temperature. As the cost of transport is dimensionless it allows the two motions to be compared regardless of their duration. It is then interesting to see that the cost of transport in Fig. 3.14-(tool upstairs) and in Fig. 3.11-($10°C$) are very similar. And that, at the same temperature, the total cost of transport for [Carpentier 2016]

CDPG is 9.6% better (from 6.71 to 6.06). One explanation is that the motion from [Carpentier 2016] **CDPG** being more dynamically consistent, the stabilizer consumes less energy to compensate for the model simplifications.

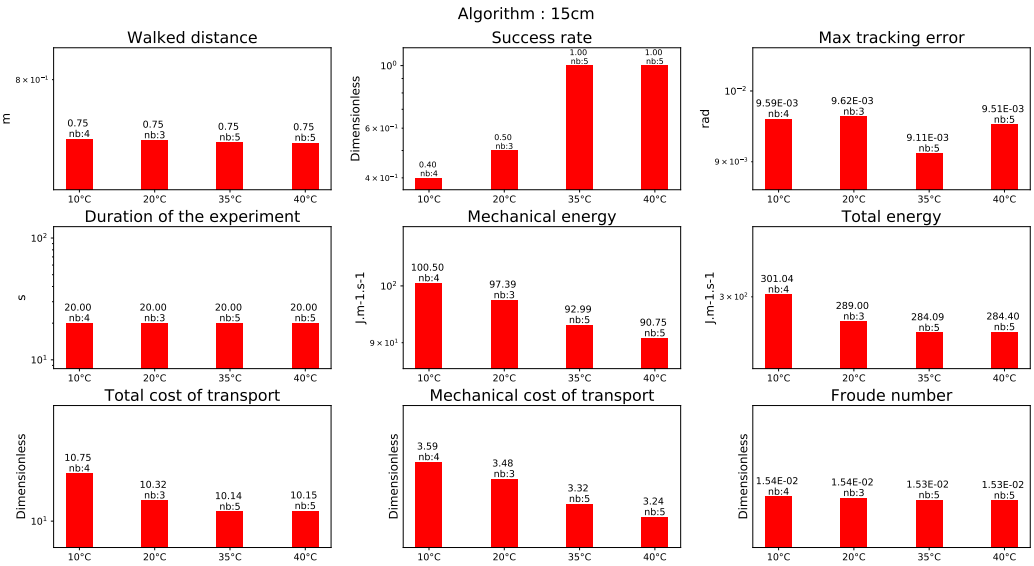


Figure 3.12: Climbing 15 *cm* stairs with a handrail

3.4.1.2 Stairs of 15 *cm*

In this experiment, the humanoid robot HRP-2 is climbing stairs of 15 *cm* height using a handrail. In addition the robot is not using any stabilization algorithm, because there are non-coplanar contacts. In this setup the [Morisawa 2007] **CDPG** has to be used without handrail because of the model simplifications. Trials have therefore been done using a **WBC** (described in [Mansard 2009]) without the handrail. The results show that the current demanded by the motors went up to 45 *A*. And because the HRP-2 batteries cannot provide more than 32 *A*, all trials failed. This is the reason why the results are not shown in this study. Nevertheless, tests using the handrail could be performed with [Carpentier 2016] **CDPG**. The corresponding results are depicted in Fig. 3.12. It confirms that the energy is decreasing with the increase of temperature without the stabilizer. Note that the energy spent by the robot is clearly higher than for the experience on the 10 *cm* stairs, i.e. a 36% increase of the energy for walking.

3.4.1.3 Stepping Stones

In this experience, the humanoid robot HRP-2 had to walk up and down on stairs made of red interlocking paving stones. Between each stair there is a height difference of ±5 *cm*. The **CDPG** described in [Morisawa 2007] was used. this test is slightly

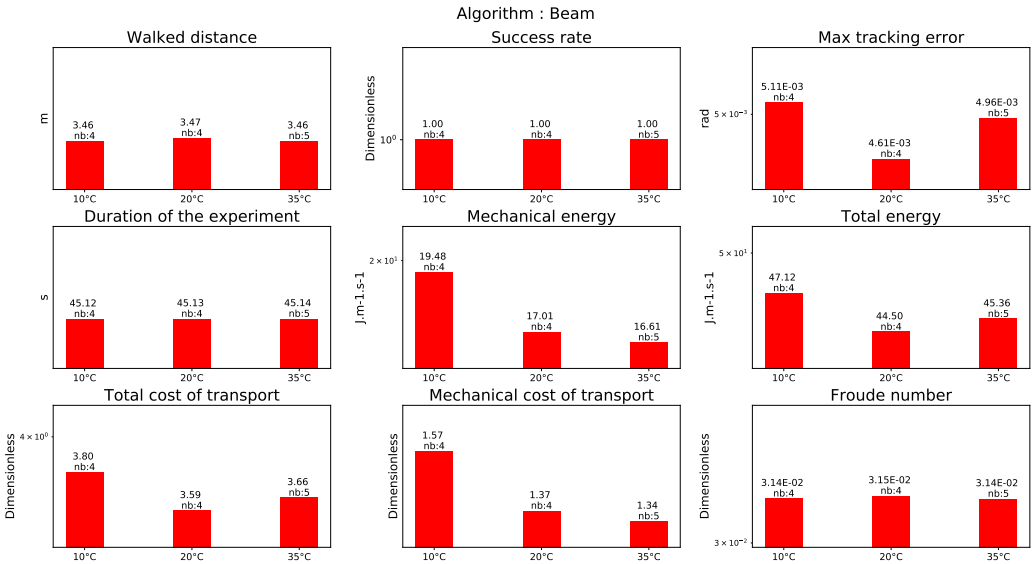


Figure 3.13: Walking on a beam

different from the previous experiments because the robot cannot put his two feet on a same level surface (contrary to a stair step). To cope with this, the generated trajectories had to always change the height of the next support foot. As the paving stones were always slightly moving due to the robot weight, the balance was difficult to obtain in a reliable way. As indicated in the graph depicted in Fig.3.14, despite a success rate of 1, the tracking error reaches a level ($8e^{-03}$ rad). This tracking error is greater than the one obtained during the 10 cm climbing experiment at $10°C$ but lower than the one obtained during the 15 cm climbing experiment at $35°C$ (which is the lowest for this temperature and the **CDPG**). A possible explanation of why the energy consumption is greater than during the 10 cm climbing stairs might be the instability of the stones and the fact that in this experiment the robot climbs the stairs in a human fashion, i.e not one stair at a time.

3.4.2 Walking on a beam

This experiment was realized using the **CDPG** [Morisawa 2007]. In this experiment the humanoid robot HRP-2 is walking on a beam. Initially, the experiment success rate on a real beam was around 20 %. This rate was improved to achieve a 90 % success rate, thanks to a new implementation of the dynamical filter presented in [Kajita 2003a]. It reduced the drift which is important as the beam length is 3 m long. This could probably be improved by a proper vision feed-back. However, in these experiments, the robot walked on a normal ground as if it was on a beam. The reason is the absence of a beam in the temperature-controlled room. Even though the foot step location is discarded, the balance problem is exactly the same. Here, the success rate is 1. The corresponding result is depicted in Fig. 3.13.

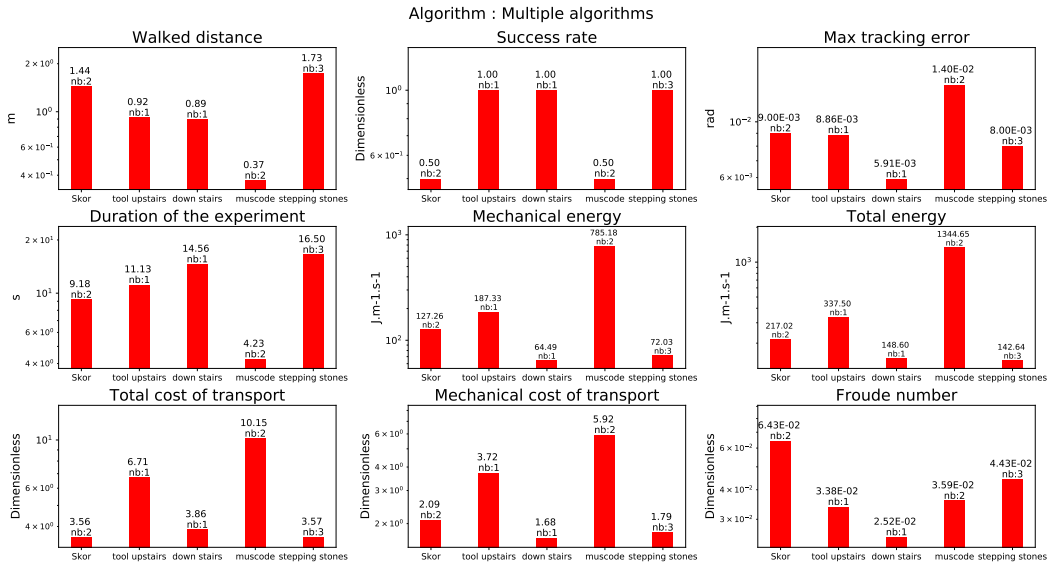


Figure 3.14: Multiple algorithms: going up with a tool on a wooden pallet 10 cm (tool upstairs), going down on a wooden pallet 10 cm(down stairs), going over an obstacle solving an **OCP** approach (Muscod), stepping on a interlocking paving stones (stepping stones)

To perform the motion on a limited bandwidth (beam), the robot has to execute faster motions with its legs in order to place its foot ahead the previous one. It is emphasized by the increase of the cost of transport compared to normal straight walking (see Fig.3.16). Though the robot's legs are moving faster, the step frequency is lowered compared to a normal walking in order to keep the joint velocities in the feasible domain. This is reflected by the fact that the Froude number is around 35% less than during a straight walking (see Fig.3.16).

3.4.3 Straight walking on flat ground

3.4.3.1 Temperatures

In the temperature-controlled room the humanoid robot HRP-2 is performing a $2\,m$ straight walk following the implementation of [Kajita 2003a]. The corresponding result is depicted in Fig. 3.16. Note that the energy with respect to the temperature is following the same trend as for the experiments on the stairs and on the beam. We also tested the algorithm [Naveau 2017] at $10°C$. The total cost of transport is higher than the algorithm [Kajita 2003a] at the same temperature but lower than the one used for walking over the beam. It is however strongly less than the total cost of transport for climbing stairs at $10°C$. The fact that the energy cost is higher for [Naveau 2017] than for [Kajita 2003a] at the same temperature is that [Naveau 2017] provides a higher range of motion but the generated motions are

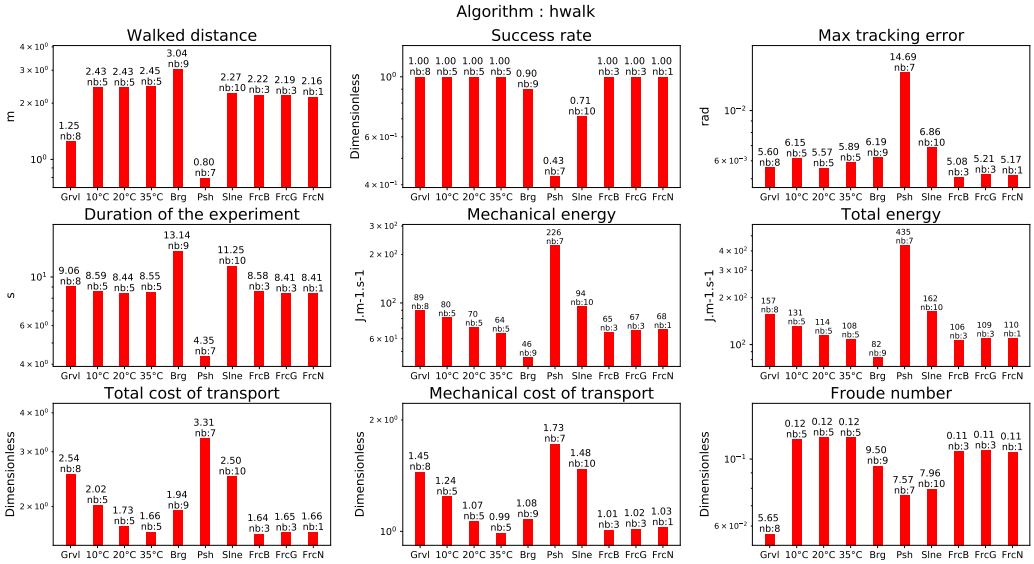


Figure 3.15: Straight walk with Kajita's walking pattern generator [Kajita 2003a]

closer to the limit of the system, so the stabilizer spends more energy to compensate for this.

3.4.3.2 Bearing weights

We made the humanoid robot HRP-2 walk while bearing weights at ambient temperature between 15° and 19°. The two algorithms [Kajita 2003a] and [Naveau 2017] were tested. The robot was able to walk while carrying up to 14 kg with the two algorithms. Note that, as expected, the effort to compensate for the additional weight reflects the cost of transport.

3.4.3.3 Pushes

We performed pushes in the lateral direction and in the frontal direction while the robot was walking along a straight line. The two algorithms [Kajita 2003a] and [Naveau 2017] were again tested. In our case, the tested algorithm was not able to modify its foot-steps according to the pushes contrary to the impressive work by [Takumi 2017]. For this specific set of experiments with push from the back, the robot was able to sustain forces from 31 N to 47 N. Pushes applied in the lateral plane were varying between 23 N and 40 N. For [Kajita 2003a], the cost of transport has a value of 3.31 similar to the one obtained when walking on the beam. It is lower than the cost of transport for climbing stairs. The cost of transport for [Naveau 2017] is of 4.08. For both algorithms pushes are among the most consuming behaviors. It is due to the stabilizer action to compensate for the perturbation.

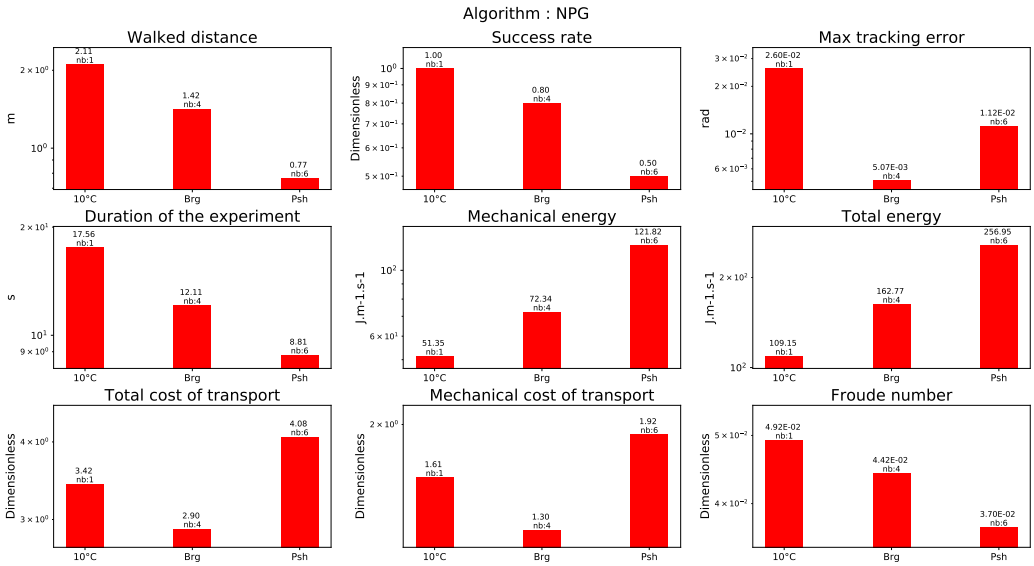


Figure 3.16: Straight walk with the walking pattern generator described in [Naveau 2017]

3.4.3.4 Slopes

The robot walked on a straight line while being on a slope of various inclinations ($[1° - 3.0°]$) -and with two possible directions (upward or downward). The two algorithms [Kajita 2003a] and [Morisawa 2007] were tested. For [Kajita 2003a] the cost of transport is higher than for standard straight walking but far less than during the pushes. For [Morisawa 2007] the cost of transport is higher than when performing the pushes with [Kajita 2003a] approach and is at the same level than the beam test. It can be explained by the fact that when the experiment has been realized the dynamical filter was not used. Therefore the stabilizer had to compensate for the discrepancy between the motion dynamics and the reference given by the center of pressure. An algorithm able to estimate the ground slope and adapt the walking pattern to it would probably increase the efficiency of this motion.

3.4.3.5 Frictions

The robot walked on carpets with different textures including different friction coefficients. In this case, we did not see any consequences with the **CDPG** [Kajita 2003a]. This is probably due to the particular coating of HRP-2 soles used, they might have avoided foot slippage, which is one way to affect the friction coefficient. A possible extension of this work would be to use a more slippery ground. But a proper way to handle such case is to implement a slip observer such as it was done [Kaneko 2005].

3.4.3.6 Uneven terrain

The robot walked over gravels of calibrated size. We tested several diameters with the **CDPG** [Kajita 2003a]. The robot was able to walk on gravels of size up to 8 mm. Beyond this size, the robot was falling. Note that in Fig.3.16 the cost of transport is slightly more expensive than for classical straight walking at nominal temperature, but not much than walking at $10°C$. It is far less expensive than climbing a slope or counteracting pushes. As expected it has no impact on the frequency of the footstep as can be reflected by the Froude Number.

3.4.3.7 Walking over an obstacle

We have computed the same performance indicators to achieve the task described in [Koch 2014] in the frame of the Koroibot project. This strategy is quite different from the others as it implements a **MPWBC** under the formulation of an Optimal Control Problem given by Eq.3.1. The solution of this problem was computed by the Muscod-II [Diehl 2001] solver. As the solver is trying to maximize a solution which is not on a reduced space (the centroidal dynamics for the previous algorithms), but on the whole robot, the solution found is close to the limits of the robot in terms of joint position, velocity, acceleration and torques. This is reflected in the cost of transport which is very high, 10.15, almost as high as for climbing the stairs of $15cm$ (see Fig.3.14-(Muscod)).

3.4.4 Stabilizer

The stabilizer described in [Kajita 2007] and [Kajita 2001] was extremely resilient during all the tests. A horizontal testbed platform was used to generate oscillations along the sagittal plane and the perpendicular plane at 1 Hz and 2 Hz at various amplitude $[10, 20, 30, 40, 48]$ in mm. Along the sagittal plane at 40 mm and 48 mm for both frequencies the feet of the robot were raising up. In the perpendicular plane at 40 mm and 48 mm for both frequencies the overall robot rotated of about 15° and 20°. It was also tried to increase the frequency for a given amplitude of 10 mm. In the sagittal plane, the robot was able to reach 7 Hz without falling. In the perpendicular plane at 7 Hz the robot was making violent oscillations (without falling) reaching mechanical resonance. The trial was subsequently stopped. The results are depicted in Fig.3.17. We can clearly see that for the oscillation in the perpendicular plane the increase of total energy is following an exponential curve, compared to the same experience in the sagittal plane. This clearly shows that the resonance frequency of the system was reached as it can be seen in the video available at the following location https://www.youtube.com/watch?v=djWGsb44JmY&feature=youtu.be.

3.5 Discussion

Human performance in locomotion tasks is still unmatched by humanoid robots. Because of the lack of assessment methods shared and accepted by the entire

robotics community, it is even difficult to estimate the level of maturity of existing technologies. A response to these evaluation needs should induce significant advances in robotics-research. Such an influence of evaluation on the progression of technology performance has been observed in the past, in particular for computer vision and NLP tasks [Martin 2004].

The definition of evaluation protocols including testing scenarios, testing environments and KPIs or metrics is crucial for the definition of common standards for:

- certifying humanoid robots (i.e. to guarantee the conformity of the product to fixed quality and performance requirements);
- allowing the user to make an informed choice when selecting a specific robot among existing technologies;
- establishing a shared reference on which developers and buyers of these technologies can agree in order to define specifications.

This study contributes to the definition of these performance evaluation standards by proposing reproducible experiments and evaluating repeatable performance measurements. These evaluation methods are intended to be passed on to the robotics research community and to standardization committees. In addition we proposed one of the first thorough evaluation of such performance indicators on a human size humanoid robot.

3.5.1 Summary and major outcomes

In this chapter we presented a benchmarking for the control architecture described in Fig.3.1 that was implemented on the HRP-2 robot owned by LAAS-CNRS. The performance indicators used in this work are mostly based on [Torricelli 2015]. Based on this work we computed the following set of KPI:

- walked distance,
- success rate,
- maximum tracking error,
- duration of the experiment,
- mechanical joint energy,
- actuators energy,
- cost of transport,
- mechanical cost of transport,
- Froude number.

These KPI represent either the particular characteristics of the experiments or the performances of the control architecture used. The list of algorithms executed on the HRP-2 robot were:

- a flat ground **CDPG** from [Kajita 2003a],
- an analytical flat ground **CDPG** from [Morisawa 2007],
- a nonlinear flat ground **CDPG** from [Naveau 2017],
- a multi-contact **CDPG** from [Carpentier 2016],
- a **MPWBC** from [Koch 2014],
- a **WBC** which is the stabilizer from [Kajita 2007] and [Kajita 2001]
- a **WBC** that computes the joint position from the end-effector plus center of mass trajectories from [Mansard 2009]
- a **WBC** that computes the joint acceleration from the end-effector plus center of mass trajectories used in [Carpentier 2016]

The list of environmental conditions where the tests could successfully be performed is:

- a temperature controlled room which provided from $10°C$ to $35°C$,
- a sloped ground of various inclinations ($[1° - 3.0°]$),
- a controlled mobile platform that simulates a translating ground,
- a set of calibrated weight from 5 kgs to 15 kgs,
- a stick equipped with a force sensor at its tip to apply to measure perturbation on the robot,
- different floors with different frictions.

The list of motion performed in the environmental conditions :

- climbing up $10\,cm$ high stairs without handrail,
- climbing up $15\,cm$ high stairs with handrail,
- walking over stepping stones,
- walking on a beam,
- walking on a flat ground,
- walking on a slope,
- walking over obstacles.

From all these results and experiments few major results come out. First the temperature plays a role on the energy consumed during a motion. We observed that the colder the room is, the more mechanical and electrical energy is consumed. We also noticed that the more the motion is at the limit of stability the more the stabilizer has to inject energy into the system to compensate for potential drift. This creates a noticeable increase in energy consumption, e.g. when the robot walks on a beam, steps over obstacle, walks on stepping stones. However the most expensive motion is climbing stairs which is clearly a challenge for future potential applications

in which stairs are involved. Finally, in terms of cost of transport, the algorithm proposed by [Carpentier 2016] seems to be the most efficient and the most versatile. Its main disadvantage during this campaign was the lack of on-line implementation compared to [Morisawa 2007] and [Naveau 2017].

3.5.2 Limits

The main limit in the approach proposed here is the difficulty to make the experiments to be more statistically significant. In its current form at least 3 people are needed to perform one experiment, which makes them error-prone and time consuming. Given the wide range of motions that a humanoid robot is able to perform, wear testing needs humanoid robots to be able to fall down and stand up again and restart their behavior. This is a current hot topic in humanoid robotics. The Atlas humanoid robot built by Boston Dynamics has recently demonstrated its capabilities to fall down without breaking and stand up. HRP-2 is an electric-based humanoid robot which is mechanically fragile due to its harmonic drive. Although several works [Samy 2015, Fujiwara 2006] have developed new approaches toward making such robot more resilient to falling, it is still difficult to implement them in practice due to the cost of failure. In the meantime, benchmarking will help to understand the repeatability and the robustness of the various algorithms implemented on humanoid robots. For very unstructured environments more tests will probably be needed, and a way to classify the environments will be necessary (using gravels, stairs, size of stairs, different shapes of stairs, or database of environments, forests). But so far such environments can be handled only by a small number of humanoids and the approach proposed in this work is feasible for a large set of current humanoid robots.

3.5.3 Future work

We could not properly compute the KPI when trying to vary the friction of the ground. A future work is then to implement a proper slip observer like the one in [Kaneko 2005]. Based on this we should build a stabilizer that could be used in multi-contact motions to compensate for external perturbations and modelling assumption. Furthermore, the LAAS-CNRS has acquired a new humanoid robot Talos [Stasse 2017]. The future work consists in implementing all the algorithms presented in this work and perform the benchmarking on this new robot.

Supplemental Data

As a reminder, a CAD model of the staircase used is available on the github repository where all the log of the experiments are also present: https://github.com/laas/koroibot_KPI. All the computation performed on the logs and implementing the key performance indicators are available here: https://github.com/laas/EnergyComputation.

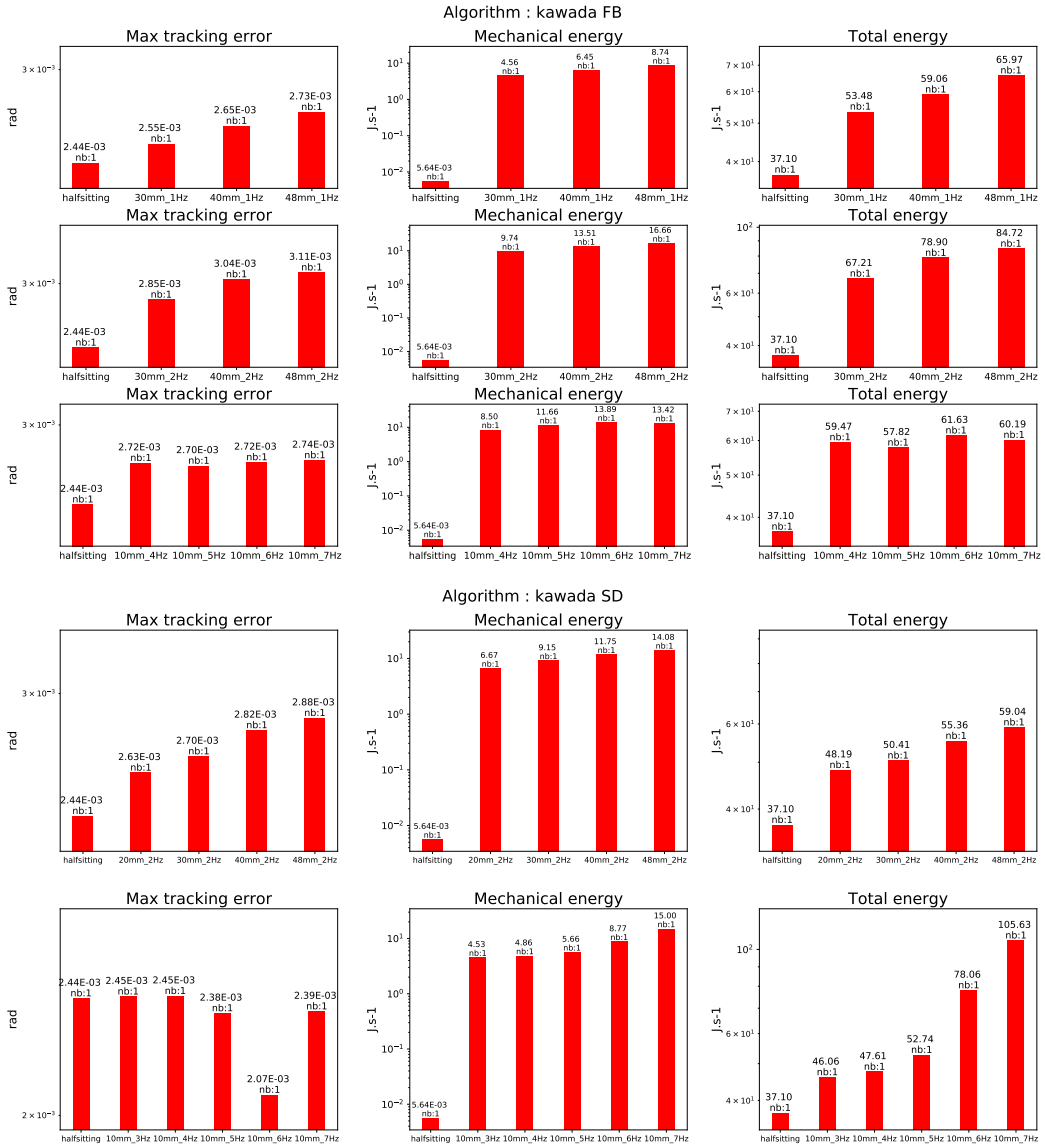


Figure 3.17: Evaluation of the stabilization algorithm described in [Kajita 2007] and [Kajita 2001]. The upper figure shows the results along the sagittal plane, whereas the lower figure depicts the results along the perpendicular plane.

Chapter 4

Approximation of the control signal by pulse basis functions

Contents

From the previous chapters, we can notice that computation time is a key point in controlling humanoid robots. Indeed, the motion is more repeatable and robust if the algorithms have time to correct the disturbances effects. Obviously, the quality of the model is also an important factor on the repeatability and the robustness. The non-linearities are often neglected or linearized for keeping a reasonable computation speed. This trade-off implies losing physically meaningful phenomena in the model. In this chapter we deal with increasing computation velocity for solving a similar optimal control problem as in [Herdt 2010a] (referred in paragraph 1.3.5). The used method is based on expressing control in lower order basis. My contribution for this part lies in the whole body joint trajectory generation with the inputs (foot placements and COM trajectory) given by our collaborators from the engineering school university of Osaka. This chapter is based on a journal paper resulting from this work [Zhang 2019]. For clarity reasons, in the introduction of this chapter the explanations about related works are recalled. They are partly redundant with Chapter 1 but both specific formulations and notations are consistent with the main development of this chapter.

4.1 Introduction

Humanoid robots are promising candidates for a broad variety of applications, for example, disaster relief, industrial labor and domestic assistance [Grey 2016]. Advanced motion controllers are now closer to realtime feasibility, but they require powerful multi-core CPUs. Specially, the bipedal nature of humanoids requires footstep planning which is computationally expensive and in realtime cases, the sensor information should be managed in the meantime. Therefore, to improve humanoids' locomotion, it is important to have a fast online walking pattern generator (WPG). In realtime walking motion generation, Linear Inverted Pendulum Model (LIPM) has been used in many studies [Kajita 2003a]-[Bohórquez 2017]. By using LIPM, an extension of linear quadratic regulator control by preview control is used for a closed loop approach to address the problem of bipedal walking pattern generation [Kajita 2003a]. [Harada 2006] generated a realtime biped gait using an analytical solution. However, these methods calculate the walking pattern one walking step in advance. [Morisawa 2009] proposed an analytical solution based WPG whose can maintain balance against disturbance. [Nishiwaki 2012] proposed a preview control based realtime WPG which calculation cycle is about $10 - 40ms$. A method combining Gauss Pseudospectral with preview control based WPG is tested on HRP-2 [Takasugi 2017]. [Englsberger 2011] proposed the control of the unstable dynamics of capture point (CP) and extended the CP to a 3D divergent component of motion (DCM) [Englsberger 2015]. However, it is difficult to add constraints for the center of pressure (CoP) in their design. The utilization of MPC allows the incorporation of CoP constraints in the controller design [Krause 2012]. [Romualdi 2018] compared the walking performance of DCM based on MPC with that of DCM based on instantaneous controller [Englsberger 2011]. The CoP trajectory generated by MPC is smoother while the instantaneous controller achieved a faster walking velocity. There are a lot of attempts in WPG including several constraint conditions such as CoP. For realizing such purposes, MPC-based WPG was proposed and has been proven to be successful [Herdt 2010b], [Herdt 2010a]. It has also been extended to have abilities such as obstacle avoidance [Naveau 2017] and disturbance rejection [Shafiee-Ashtiani 2017]. Reducing the calculation time in realtime WPG is important, since there are computational loads from analysis of sensor data. Few attempts have been tried on reducing the calculation time of MPC based realtime walking pattern generator. In order to have a faster online WPG, we go further and propose reformulating the control signals of the MPC using basis functions, as illustrated in Figure 4.1.

Decreasing the number of control inputs is a solution to improve efficiency in the MPC [van Donkelaar 1999], [Muehlebach 2016]. In the conventional discrete time MPC [Herdt 2010b], [Herdt 2010a], the control signal can be considered as a combination of pulse basis functions. To cover the whole time horizon(N_p), the number of basis functions required is N_p. In this work, we propose applying Haar and Laguerre basis functions to approximate the control signal. Since Haar and Laguerre functions carry more information than pulse functions, fewer basis functions are

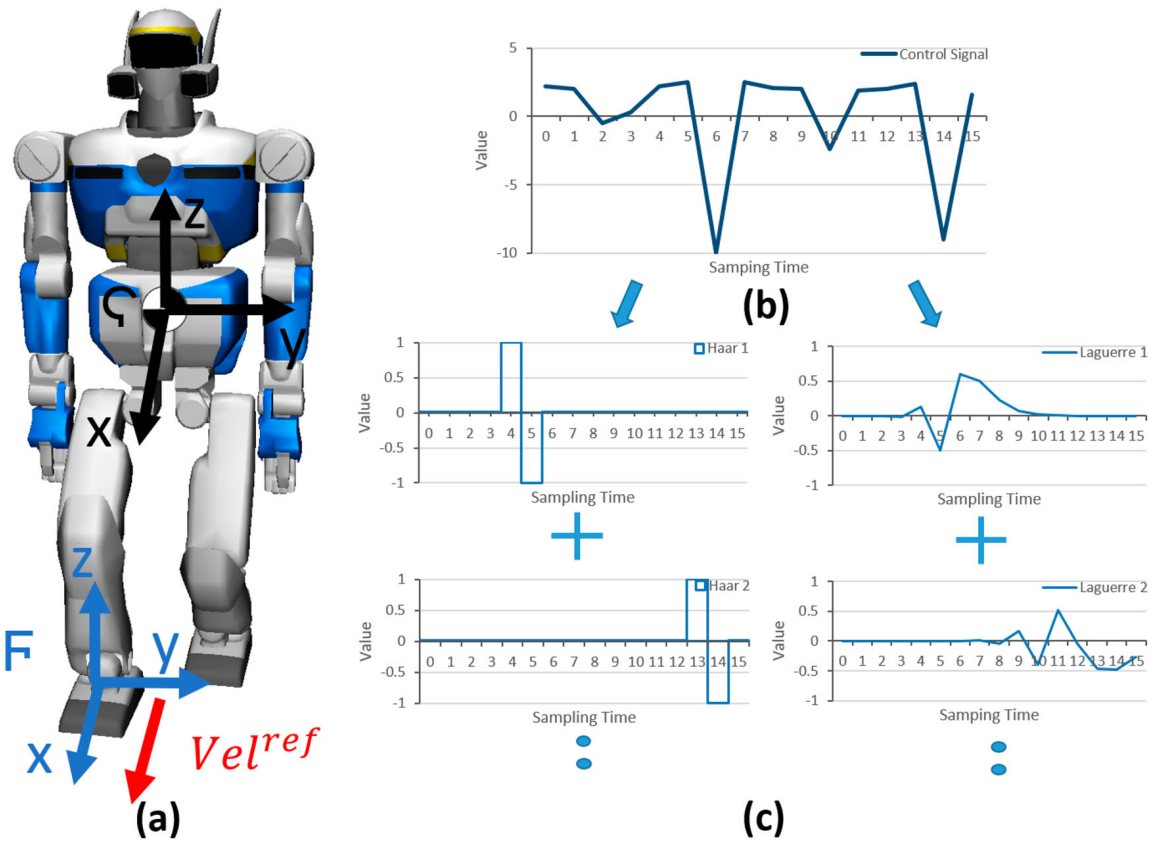


Figure 4.1: In (a), the figure shows two frames ς, F , representing CoM and the supporting foot respectively. (b) shows the control input in MPC-based WPG and two extreme values. In order to improve computational efficiency, we use Haar (left part in (c)) or Laguerre (right part in (c)) basis functions to reformulate the control signal.

needed for representing the input, leading to a reduction in the number of control inputs. Furthermore, Haar and Laguerre functions specially suit the property of inputs in WPG. The control signal in Figure 4.1(b) might be difficult to approximate due to its sharp changes. However, Haar functions can naturally represent the curves while Laguerre functions are embedded with a tuning parameter which modifies them to fit the desired signal. Basis functions are not new and have been already treated in several works [Wang 2009]-[Chen 2018], where the control signal is represented by a set of basis functions, for example, Laguerre, Kauz and Haar. It provides a more compact problem description with fewer control variables, which leads to computational benefits. Haar functions are used to build a transformation matrix which changes the signal from time domain into wavelet domain. This transformation decomposes signals into several different frequency components without destroying their time domain structures [Lee 1995]. Laguerre functions have been applied to the identification of linear time invariant systems for a long time [Wahlberg 1996]. In system identification, the purpose of implementing Laguerre functions is to get low

order models and to incorporate a priori information to the system's time constants. The main contribution of this work is a basis function approach to the MPC-based WPG. The method is a parametric description of the MPC's control inputs via linear combinations of basis functions. Moreover, instead of restricting to special types of basis functions, we provide a general parametrization approach which allows users to select their own basis function. To fit the property of bipedal walking, Haar and Laguerre functions are introduced. Examples of two MPC designed with these different basis functions are shown and compared. This work is organized as follows: Section 2 introduces the dynamic walking motion for both the bipedal's body and feet. Section 3 discusses the MPC scheme. Cost function and constraints are discussed in this part. Then, the properties of walking motion are analyzed in Section 4. The Laguerre and Haar basis functions and its use are proposed in Section 5. Finally the conclusion and future work are presented.

4.2 Dynamic walking motion

4.2.1 Motion model of center of mass (CoM)

In Figure 4.1, a frame ς is attached to the position of the CoM and to the orientation of the robot's trunk. The orientation θ is around vertical axis Z in the frame. In order to generate a smooth motion of the CoM, we consider that its trajectory is differentiable three times. The jerks are assumed to be piece-wise linear on the time intervals $T = 0.1s$. We define the superscript C which implies the states on the CoM. $C \in c_x, c_y$ indicates the axes of the horizontal plane in the frame ς, the future states $\hat{x}^C_{k+1} = \begin{bmatrix} x^C_{k+1} & \dot{x}^C_{k+1} & \ddot{x}^C_{k+1} \end{bmatrix}^T$ (including the CoM's position, velocity and acceleration) can be generated by

$$\hat{x}^C_{k+1} = A\hat{x}^C_k + B\dddot{x}^C_k \tag{4.1}$$

where

$$A = \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix}, B = \begin{bmatrix} T^3/6 \\ T^2/2 \\ T \end{bmatrix}$$

Equation 4.1 is the motion model of the CoM. The real motion of the CoM will be set to follow this motion model. To define the CoM over the prediction horizon N_p, the vector X^C_{k+1} is used to express the positions of the CoM over the whole horizon, $X^C_{k+1} = \begin{bmatrix} x^C_{k+1} & \cdots & x^C_{k+N_p} \end{bmatrix}^T$, similar definitions are used to describe $\dot{X}^C_{k+1}$,$\ddot{X}^C_{k+1}$ and $\dddot{X}^C_{k+1}$. Prediction horizon (N_p) indicates the length of state variables X^C_k. By defining the state $\hat{X}^C_{k+1} = \begin{bmatrix} X^C_{k+1} & \dot{X}^C_{k+1} & \ddot{X}^C_{k+1} \end{bmatrix}^T$ and the control input $U^C_k = \dddot{X}^C_k$ of the following state equation, we obtain the prediction model of the CoM,

$$\hat{X}^C_{k+1} = A_c\hat{X}^C_k + B_cU^C_k \tag{4.2}$$

where matrices A_c , B_c can be obtained from the recursive application of Equation 4.1. To distinguish the prediction horizon N_p , the length of control input U_k^C is defined as control horizon N_c . This is because the control horizon can be reduced by basis functions while the prediction horizon is always kept with same value. The orientation of the CoM, X^{C_θ} in frame ς can be calculated with a similar prediction form of Equation 4.2 (see [Herdt 2010a] for more details).

Assuming that the centroidal dynamic [Orin 2013] which is the dynamic of a humanoid robot projected at its CoM is linear (the angular momentum produced by the rotations of the robot is supposed to be zero and the CoM evolves on a horizontal plane [Naveau 2017]) when all the contacts with the environment are coplanar, the CoP should strictly lie in the support polygon in order to meet the balance criteria. The positions of CoP, y^C , can be approximated by LIPM. The CoP is predicted by a linear function of the CoM

$$y_{k+1}^C = C\hat{x}_{k+1}^C \tag{4.3}$$

where $C = \begin{bmatrix} 1 & 0 & -h/g \end{bmatrix}$, h and g are the height of the CoM and the norm of the gravity vector, respectively. The vector of the CoP over the prediction horizon is defined as $Y_{k+1}^C = \begin{bmatrix} y_{k+1}^C & \cdots & y_{k+N_p}^C \end{bmatrix}^T$, thus we have

$$Y_{k+1}^C = C_c\hat{X}_{k+1}^C \tag{4.4}$$

where matrix C_c is a collection of c in Equation 4.3.

4.3 Properties of control signal

Unlike conventional MPC [Herdt 2010b, Herdt 2010a], we use basis functions to approximate control input U_k^C in Equation 4.2. The physical meaning of control input are jerks of the CoM, $\dddot{X}_k^C$ which can be separated into two directions $\dddot{X}_k^{c_x}$, $\dddot{X}_k^{c_y}$ on X, Y axis. Before reformulation of the jerk, let us have a look at velocity and acceleration of the CoM.

4.3.1 Linear inverted pendulum model

The WPG is based on a 3D LIPM. In Figure 4.2(b), the walking motion of the LIPM is separated into single support (SS) phase and double support (DS) period. With $N_p(= 16)$ time intervals ($0.1s$), the time horizon of prediction is $1.6s$. The full duration of one step is $0.8s$, including single support ($0.7s$) and double support ($0.1s$). Two steps are predicted in a prediction horizon. Figure 4.3 shows the velocity and acceleration of the robot in X direction during a horizon. Small fluctuations happen in SS at $0 - 0.2s$ and $0.8 - 1.0s$, these are related to the dynamic of the LIPM. During the DS period ($t = 0.6s$ and $1.4t = 0.6$ and $1.4s$), the acceleration jumps from maximum to minimum and the CoP moves from the backward support foot to the forward one, see Figure 4.2(b).

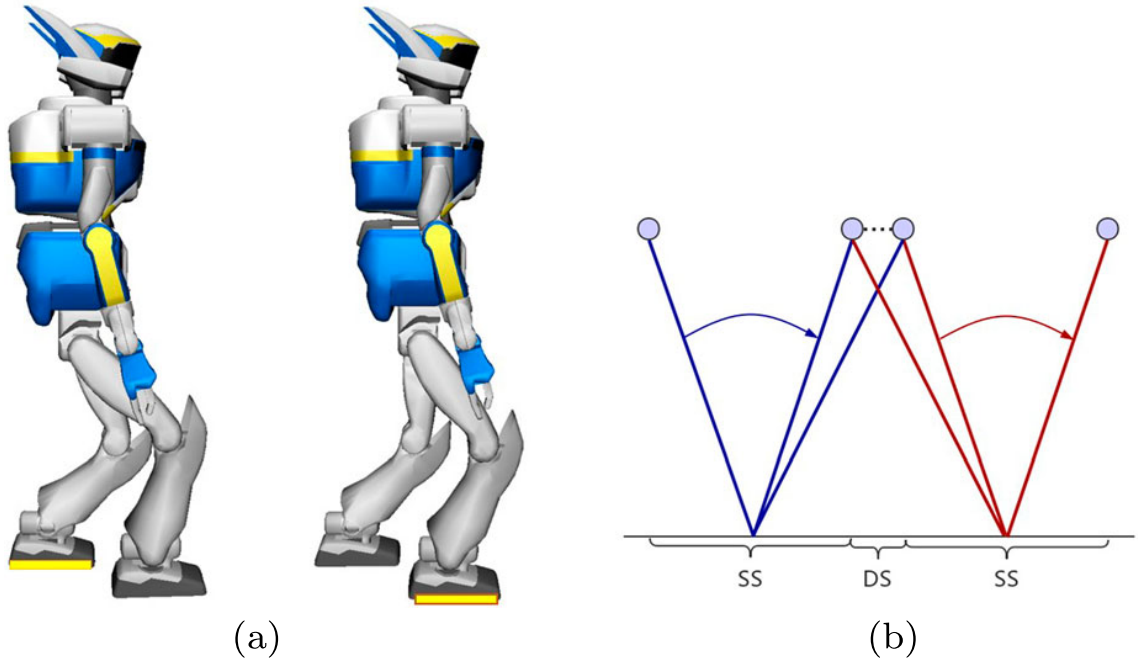


Figure 4.2: During DS period, the CoP moves from one foot to the other. (a) The yellow marks indicate the support feet in SS before and after the DS period. (b) A full step includes SS and DS.

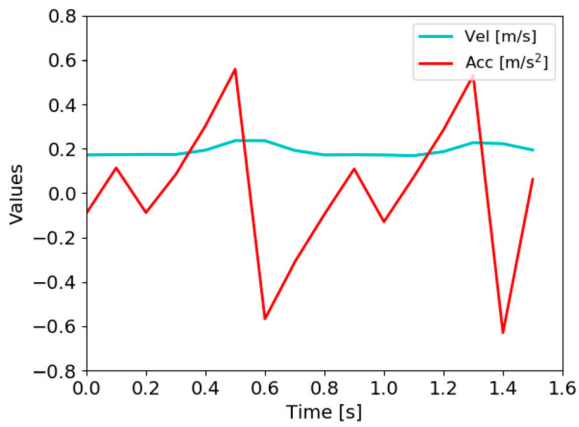
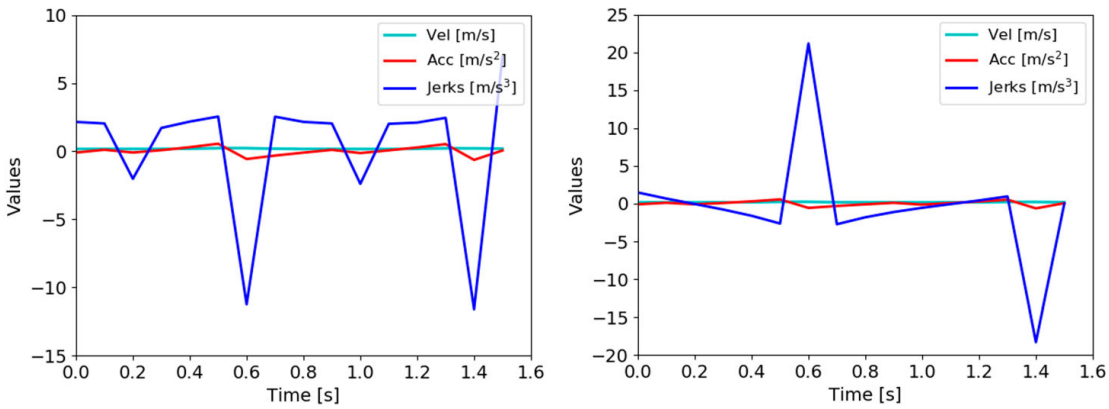


Figure 4.3: Detailed information about the velocity and acceleration of the robot in X direction during a prediction horizon. At the time periods $0.5 - 0.6s$ and $1.3 - 1.4s$, the acceleration jumps from maximum to minimum.



(a) (b)

Figure 4.4: The velocity, acceleration and jerks of CoM along X and Y directions. Extreme values of jerks happened in the DS phase. (a) Along X direction, the jerk has two deep lows at $t = 0.6, 1.4s$. (b) Along Y direction, the jerk has a peak and a low at $t = 0.6, 1.4s$.

4.3.2 The jerks

The changes of acceleration in DS require a big jerk as shown in Figure 4.4. Unlike the conventional MPC, we use basis functions to approximate control inputs U_k^C which are the jerks of CoM, $\dddot{X}_k^C$. However, two deep lows in the jerk make the control input hard to be represented. To solve this problem, we introduce Haar functions and Laguerre functions which are able to represent this kind of signal. As depicted in Figure 4.4, jerks along X and Y axis have peaks. The jerk along the Y axis is more difficult to catch as it changes sign due to the feet transition.

4.4 Approximation with basis functions

In the conventional discrete time MPC [Herdt 2010b, Herdt 2010a], the control input $\dddot{X}_k^C$ in Figure 4.4 is represented by a series of pulse basis functions. In pulse basis functions, as shown in Figure 4.5, information is carried at certain sampling time. In order to cover the whole time horizon($N_p = 16$), the number of pulse functions required is N_p . The length of the control input is the number of basis functions, which is $N_c = 16$. To reduce the number of basis functions, Haar basis functions and Laguerre basis functions are implemented to approximate the control input of the MPC. There are two reasons for applying Laguerre and Haar functions. Firstly, both a single Laguerre function (in Figure 4.6) and a single Haar function (in Figure 4.7) carry more information compared with pulse functions. As a result, to represent the same control signal, fewer basis functions are needed which also means the dimension of the control input is reduced. Secondly, Haar and Laguerre functions specially suit the property of the WPG's control input. From Figure 4.4,

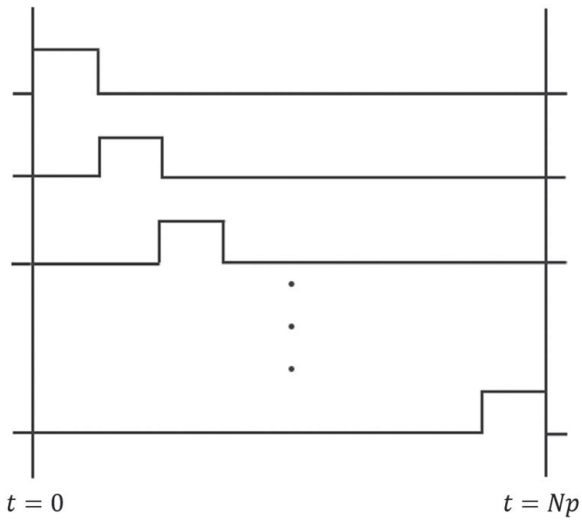


Figure 4.5: The control input of the MPC in discrete time can be considered as a combination of pulse functions.

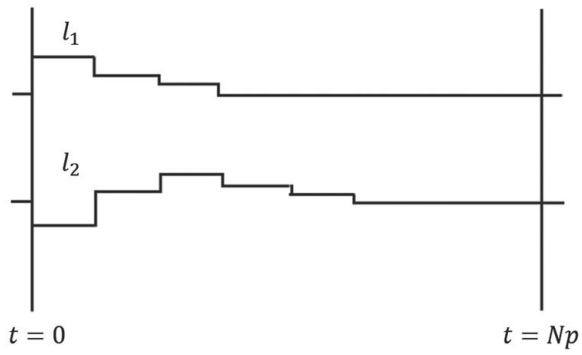


Figure 4.6: An example of Laguerre functions in discrete time with $a = 0.5$.

it can be seen that two extreme values (at $t = 0.6$ and $1.4s$) appear in the control input due to the LIPM and such a sharp change makes it hard for basis functions to approximate the control input. However, Haar functions can naturally represent the values and steep changes in inputs. Laguerre functions are embedded with a tuning parameter a which can modify the shape of Laguerre functions to fit the property of the desired signal. Details of Haar functions and Laguerre functions are discussed in this section.

4.4.1 Input in conventional MPC

Before introducing Laguerre functions, let us recall the control inputs of the MPC from Equation 4.2 and express them in the time horizon.

$$U_k^C = \begin{bmatrix} u_k^C & u_{k+1}^C & \cdots & u_{k+N_c-1}^C \end{bmatrix}^T \tag{4.5}$$

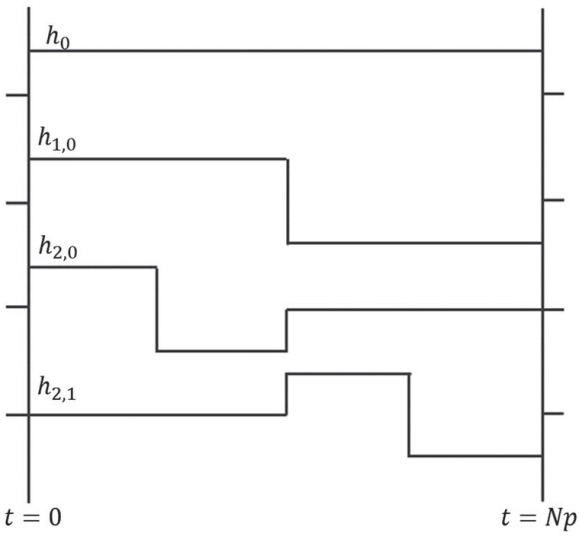


Figure 4.7: Haar basis functions.

The dimension of the control input is N_c which is the control horizon. At time k, elements with U_k^C can be represented by the discrete δ functions combined with U_k^C ,

$$u_{k+i}^C = \begin{bmatrix} \delta_i & \delta_{i+1} & \cdots & \delta_{i-N_c+1} \end{bmatrix} U_k^C \tag{4.6}$$

where $\delta_i = 1$ if the subscript $i = 0$; $\delta_i = 0$ if $i \neq 0$. Here the δ functions are considered as a pulse operator and the δ_{i-d} shifts the pulse forward as index d increases. Therefore, the pulse functions (or step functions) are representing the control input if we take U_k^C as the coefficient vector. Other thing that should be noted is that the input trajectories u_{k+i}^C can also be approximated by a discrete time polynomial function.

4.4.2 Laguerre basis functions

4.4.2.1 Constructions

The discrete time Laguerre functions are derived from the discretization of continuous-time Laguerre functions [Wang 2009]. In this work, the state space form of Laguerre functions will be implemented. Let the initial Laguerre function be:

$$L_0 = \sqrt{1-a^2} \begin{bmatrix} 1 & -a & a^2 & -a^3 & \cdots & (-1)^{N-1}a^{N-1} \end{bmatrix}^T \tag{4.7}$$

where a is the pole of the discrete time Laguerre function and $0 \leq a < 1$. N shows the division of Laguerre vectors. Let, $L_k = \begin{bmatrix} l_k^1 & l_k^2 & \cdots & l_k^N \end{bmatrix}$, then,

$$L_{k+1} = A_l L_k \tag{4.8}$$

where the matrix $A_l(N \times N)$ is a function of a. For example, for $N = 3$,

$$A_l = \begin{bmatrix} a & 0 & 0 \\ \sqrt{1-a^2} & a & 0 \\ -a\sqrt{1-a^2} & \sqrt{1-a^2} & a \end{bmatrix}$$

$$L_0 = \sqrt{1-a^2} \begin{bmatrix} 1 \\ -a \\ a^2 \end{bmatrix} \tag{4.9}$$

Laguerre functions are orthonormal functions, therefore have the following property,

$$\sum_{k=0}^{n_p} l_k^i l_k^j = 0 \quad for \quad i \neq j$$
$$\sum_{k=0}^{n_p} l_k^i l_k^j = 1 \quad for \quad i = j$$

4.4.2.2 Control inputs

At the beginning of this section, we mentioned that the control inputs in MPC are represented by a set of pulse functions. Here, the Laguerre functions are used to describe the control inputs:

$$u_k^C = \sum_{i=1}^{N} c^i l_k^i \tag{4.10}$$

In a vector form,

$$U_k^C = L_k^T C_l \tag{4.11}$$

where

$$C_l = \begin{bmatrix} c^1 & c^2 & \cdots & c^N \end{bmatrix}^T$$

and c are parameters which will be derived by solving the cost function. If Laguerre functions are used to approximate the control inputs, the new control inputs in Equation 4.11 are C_l . Now the length of the control inputs is the number of Laguerre functions used (N) while in the conventional MPC, the length of the control inputs of Equation 4.5 is N_c . As a single Laguerre function carries more information in comparison with the pulse function, fewer number of Laguerre functions are required to formulate the input, $N < N_c$. As a result, the length of the control input is shortened and the computational load is also decreased.

4.4.2.3 Motion with Laguerre functions

The MPC with Laguerre basis functions is closely related to the conventional MPC. If we put Equation 4.11 in Equations 4.2 and 4.4, the future states of the CoM and

feet can be generated in the form of Laguerre functions as

$$\hat{X}^C_{k+N_p} = A_c^{N_p}\hat{X}^C_k + \sum_{i=0}^{N_p-1} A_c^{N_p-i-1}B_cL_i^TC_l \tag{4.12}$$

$$Y^C_{k+N_p} = C_cA_c^{N_p}\hat{X}^C_k + \sum_{i=0}^{N_p-1} C_cA_c^{N_p-i-1}B_cL_i^TC_l \tag{4.13}$$

4.4.2.4 Selection of parameters

There are two explicit tuning parameters in the design of Laguerre functions which are the pole of the discrete time Laguerre function a, and the number of Laguerre functions, N. The parameter a determines the configuration of Laguerre functions. If the control horizon N_c is known, a can be selected in the area near $a \approx exp(-(k_l/N_c))$ where k_l is set between 5 and 10 according to the number of Laguerre functions that are applied. However, for the MPC used in the WPG, it is recommended to select a small value of a, due to the steep change of the control input that appears in the double support phase. In order to guarantee feasible solutions, a small value of a is our first choice. The number of terms N indicates how many Laguerre functions are applied to approximate the target. The complexity of the calculation is influenced by the choice of N. As N increases, a better description of the target is provided while computational load is also added. A small value of N means a fast calculation however it may lead to an unfeasible solution. At the beginning, N can be chosen with the same value as N_c and then designers can reduce N until an acceptable balance between accuracy and efficiency is achieved.

4.4.2.5 Notes

(1) Fast and steep changes (happened in double support phase) in the control inputs can be handled with Laguerre polynomials.
(2) It is easy to tune the performance of the MPC with Laguerre functions as there are two explicit parameters which are a and N.
(3) The MPC with Laguerre functions is the same as the traditional MPC when $a = 0$ and $N = N_c$.
(4) The complexity of solving the cost function is reduced

4.4.3 Haar basis functions

Besides applying Laguerre basis functions, Haar basis functions can also be used to represent the control input.

4.4.3.1 Construction

Let the initial Haar function h_0 be the signal which assumes a constant value on the unit interval $[0, 1)$ as follows:

$$h_0 = \frac{1}{(\sqrt{2})^m} \tag{4.14}$$

where m is an integer. From the point of view of signal representation, a large m indicates smaller resolution and higher frequency.

A set of Haar wavelets are orthogonal functions defined as

$$h_{m,k} = \frac{1}{(\sqrt{2})^m}\psi\left(\frac{1}{2^m}t - k\right) \tag{4.15}$$

where k relates to the phase shift of the wavelets and $\psi(t)$ is the indicator function

$$\psi(t) = \begin{cases} 1 & if \quad t \in [0, \frac{1}{2}) \\ -1 & if \quad t \in [\frac{1}{2}, 1) \\ 0 & otherwise \end{cases} \tag{4.16}$$

Haar wavelets are orthogonal and square-integrable in l_2 norm. The approximation of the control input can be achieved by applying Haar series.

4.4.3.2 Transformation from pulse basis

The input signal is represented by a finite discrete form of Equation 4.5, and it can be expressed with Haar functions by

$$U_k^C = \sum_{i=1}^{2^l-1} \delta_{0,k}\phi_{0,k} = \sum_{k=0}^{2^l-m-1} \sum_{m=1}^{l-1} c_{m,k}h_{m,k} \tag{4.17}$$

where δ, c are coefficients for pulse basis and Haar basis, respectively, and ϕ is the shifting pulse. Changing the input signal from the standard pulse function into the Haar basis is a unitary transformation. The transformation matrix H is orthogonal, $H^T H = I$,

$$U_k^C = H^T C_h \tag{4.18}$$

$$C_h = HU_k^C \tag{4.19}$$

where U_k^C, C_h are the matrices of coefficients δ, c respectively. For example, if $m = 2$,

$$\begin{bmatrix} c_{0,0} \\ c_{1,0} \\ c_{2,0} \\ c_{2,1} \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} & \frac{-1}{2} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \delta_{0,0} \\ \delta_{1,0} \\ \delta_{2,0} \\ \delta_{3,0} \end{bmatrix} \tag{4.20}$$

Figure 4.7 shows the Haar basis functions for $h_0, h_{1,0}, \{h_{2,k}\}_{k=0,1}$.

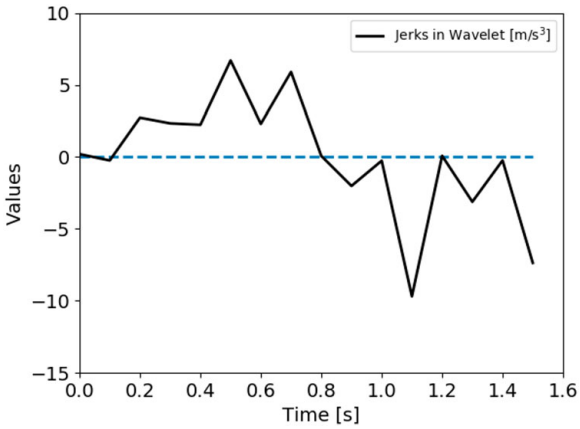


Figure 4.8: Jerks after transformation from time domain (Figure 4.4) to wavelet domain, where it is easier to find zeros in the input that can be blocked.

4.4.3.3 Blocking

Besides giving acceptable performances, a control algorithm should be computationally efficient for inexpensive on-line implementation. Blocking is a strategy to reduce the size of optimization problems. In this method, the number of vectors to be calculated is decreased by projecting the vector space onto a lower subspace. As a result, the insignificant linear combinations of the input are eliminated in the calculation. Let the input vector after the blocking be U_B and block matrix B_{blk},

$$U_B = B_{blk} U \tag{4.21}$$

B_{blk} is an orthogonal matrix and it is designed to eliminate some of the rows that are related to the particular linear combinations of the input which are set to zero a priori. Though blocking is an efficient way to reduce computational load, it is hard to apply blocking directly in MPC; since the proper choice of blocking matrix is usually obscure in time domain. Here, we propose a general method to use the blocking matrix, that is representing the control input in wavelet domain with Haar functions. In Figure 4.8, wavelet transformation provides a new perspective to view the input, where the blocking techniques can be implemented in an insightful and convenient way. As the transformation from time domain to wavelet domain is a unity one, we do not lose any information from the input. Therefore the accuracy is preserved. Here is an example of blocking. Suppose the control input in wavelet domain is U_w which has a dimension of 8,

$$U_w = \begin{bmatrix} u_{0,0} & u_{1,0} & u_{2,0} & u_{2,1} & u_{3,0} & u_{3,1} & u_{3,2} & u_{3,3} \end{bmatrix}^T \tag{4.22}$$

and blocking matrix B_{blk} is designed as

$$B_{blk} \begin{bmatrix} |1| & 0 & 0 & 0 \\ 0 & |1| & 0 & 0 \\ 0 & 0 & |1| & 0 \\ 0 & 0 & |0| & 0 \\ 0 & 0 & 0 & |1| \\ 0 & 0 & 0 & |0| \\ 0 & 0 & 0 & |0| \\ 0 & 0 & 0 & |0| \end{bmatrix} \tag{4.23}$$

The variables after blocking, U_B , are calculated by Equation 4.21,

$$U_B = \begin{bmatrix} u_{0,0} & u_{1,0} & u_{2,0} & u_{3,0} \end{bmatrix}^T \tag{4.24}$$

As a result, the length of the input vector is shortened from 8 to 4.

4.4.3.4 Motion with Haar functions

If we put Equation 4.18 in Equations 4.2 and 4.4, the future states of the CoM and feet are built in the form of Haar functions as

$$\hat{X}^C_{k+N_p} = A_c^{N_p} \hat{X}^C_k + \sum_{i=0}^{N_p-1} A_c^{N_p-i-1} B_c H_i^T C_h \tag{4.25}$$

$$Y^C_{k+N_p} = C_c A_c^{N_p} \hat{X}^C_k + \sum_{i=0}^{N_p-1} C_c A_c^{N_p-i-1} B_c H_i^T C_h \tag{4.26}$$

Blocking matrix can be added to eliminate part of the control input

$$\hat{X}^C_{k+N_p} = A_c^{N_p} \hat{X}^C_k + \sum_{i=0}^{N_p-1} A_c^{N_p-i-1} B_c B_{blk} H_i^T C_h \tag{4.27}$$

$$Y^C_{k+N_p} = C_c A_c^{N_p} \hat{X}^C_k + \sum_{i=0}^{N_p-1} C_c A_c^{N_p-i-1} B_c B_{blk} H_i^T C_h \tag{4.28}$$

4.4.3.5 Notes

(1) The transformation of signals from time domain to wavelet domain is a unity one. Thus information is kept when transforming the input from time to wavelet domain.
(2) After transforming an input into wavelet domain, a blocking matrix can be introduced to reduce the dimension of the input.
(3) The MPC with Haar functions is the same as the conventional MPC when blocking is not implemented.

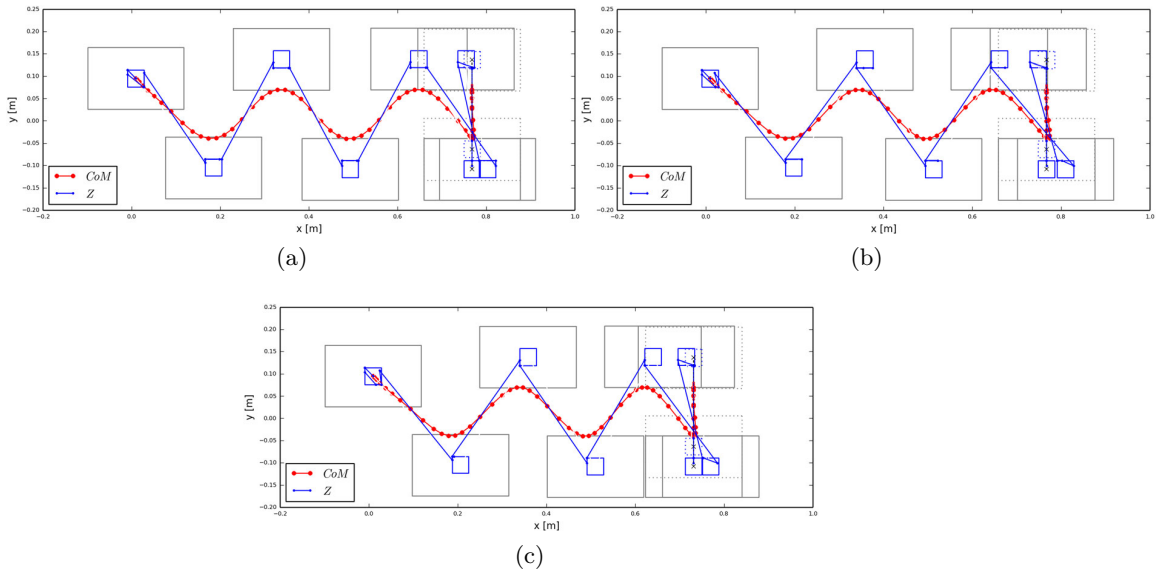


Figure 4.9: Forward walking patterns. MPC with Haar functions generates the most accurate trajectory while MPC with Laguerre functions runs faster. (a) Patterns by conventional MPC. (b) Patterns by MPC with Haar functions. (c) Patterns by MPC with Laguerre functions.

4.5 Simulation

In this section, the walking gaits produced by the conventional MPC [Herdt 2010a], MPC with Laguerre and Haar basis functions are compared. The walking pattern generator is tested on the HRP-2 humanoid robot. A step is made regularly every $0.8s$ where $0.7s$ for single support and $0.1s$ time duration for double support. The time of prediction for all the MPC tested is set to 1.6 s which means it predicts two full steps in the future. Three walking strategies are tested which are walking forward, sidewalking and walking with feet rotations. The humanoid robot starts walking with the right foot, then tracks the targeting velocity and finally take $2s$ to reach a rest position, which is a double support condition.

The robot walks forward with a velocity of $Vel_{k+1}^{ref} = \begin{bmatrix} 0.2 & 0 & 0 \end{bmatrix}^T$ for $4s$. The three components in Vel_{k+1}^{ref} represent the CoM's velocity along X, Y directions and angular velocity along vertical axis respectively. At the beginning, the jerks of the CoM along x axis $U_k^{C_x}$ are represented by basis functions. In Figure 4.9, all the MPC-based WPGs generated feasible trajectories.

From Table 4.1, it can be seen that the running time of the three controllers is different. In the conventional MPC, computational cost is $21.9ms$ while the MPC with Haar and the MPC with Laguerre save 14.6% and 29.7% of calculation time respectively, in comparison with conventional MPC. The numbers of the control inputs $U_k^{C_x}$ in the MPC, MPC with Haar and MPC with Laguerre are 16,13,10 respectively. The reduction in the numbers of the control inputs leads

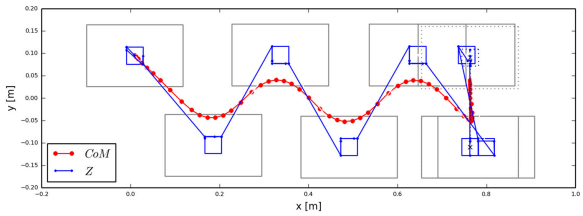
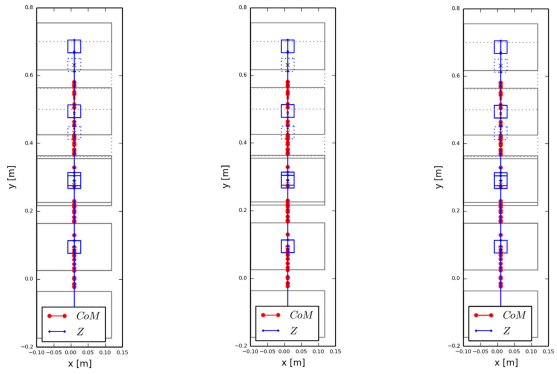


Figure 4.10: Walking patterns generated by approximation of both jerks of x and y with Haar basis functions.



(a) (b) (c)

Figure 4.11: Sidewalking patterns of the three MPCs tested. From left to right, MPC, MPC with Haar functions and MPC with Laguerre functions. Both of the MPC with basis functions work faster than the conventional MPC. (a) MPC. (b) MPC with Haar functions. (c) MPC with Laguerre functions.

Controller	Dimension of input	Errors	Time (ms)	Time saving
MPC	16	−4.7%	21.9	0
MPC with Haar	13	−0.2%	18.7	−14.6%
MPC with Laguerre	10	−7.5%	15.4	−29.7%

Table 4.1: Walking forward with approximation of jerks of x.

Controller	Dimension of input	Errors	Time (ms)	Time saving
MPC	16	−4.7%	21.9	0
MPC with Haar	12	−0.2%	22.3	+1.8%

Table 4.2: Walking forward with approximation of jerks of y.

Controller	Dimension of input	Errors	Time (ms)	Time saving
MPC	16	−3.8%	18.2	0
MPC with Haar	5	−3.3%	10.2	−43.9%
MPC with Laguerre	1	−3.3%	8.2	−54.9%

Table 4.3: Sidewalking.

Controller	Dimension of input	Errors	Time (ms)	Time saving
MPC	16	-4.8%	10.8	0
MPC with Haar	14	-5.7%	8.8	-18.5%
MPC with Laguerre	9	-7.3%	5.8	-46.2%

Table 4.4: Orientation controller.

to a faster calculation. The tuning parameters of the MPC with Laguerre are, $a = 0.3, N = 10$. Although the MPC with Haar does not work as fast as the MPC with Laguerre, it provides the most accurate result among the three tested MPCs while the computational cost is less compared with the conventional MPC. In addition to operations on the jerk of $x, U_k^{C_x}$, we then apply basis functions to represent the jerk of $y, U_k^{C_y}$. However, in the case of implementing basis functions in $U_k^{C_y}$, the MPC of Laguerre cannot generate a feasible trajectory. The MPC with Haar can produce stable foot trajectory (see Figure 4.10), but the running time of it, is higher than that of conventional MPC, see Table 4.2. The reason for such performances is that there are both a peak and a low in the jerk of y (see Figure 4.4), which makes it difficult for basis functions to approximate. In order to design a fast and stable realtime WPG, only $U_k^{C_x}$ is represented with basis functions. In the case of sidewalking, see Figure 4.11, the MPCs with basis functions have a huge improvement compared with traditional MPC. Table 4.3 shows that the reference velocity is assigned to $0.2m/s$ and the running time of process is reduced by 43.9% in the MPC with Haar and by 54.9% in the MPC with Laguerre functions. More than half of the calculation time is saved when the Laguerre functions are used. The reason of improvement in both MPC with basis functions is that when the robot walks in y direction, the CoM's trajectory along x axis does not change much. Therefore, the control inputs along x direction can be greatly reduced and it results in a fast calculation.

To demonstrate the behavior of the orientation controller, a constant velocity $V_{k+1}^{ref} = \begin{bmatrix} 0.2 & 0 & 0.2 \end{bmatrix}$ including rotation is sent to the walking pattern generator for $4s$. Both the position and the orientation controller are designed with basis functions. The setting for the position controller with basis functions is the same as in the case of walking forward. The control inputs, which are jerks along left and right direction, in the orientation controller are approximated by basis functions. Table 4.4 shows the length of the input and the calculation time of three orientation controllers. Simulation shows that the orientation controller with Laguerre functions has a quick response and it reduces by 46.2% the calculation time. By applying Haar basis functions, the calculation time is saved by 18.5% compared with conventional MPC. In Figure 4.12, all the tested MPCs generated feasible walking patterns while the MPC with basis functions work faster. In order to test the efficiency of our proposed method, simulations and experiments are done. The simulation is shown in Figure 4.13 where the HRP-2 robot walked forward with a velocity of $0.2m/s$. Experiments using the HRP-2 robot with a reference velocity of $V_{k+1}^{ref} = \begin{bmatrix} 0.2 & 0 & 0 \end{bmatrix}$

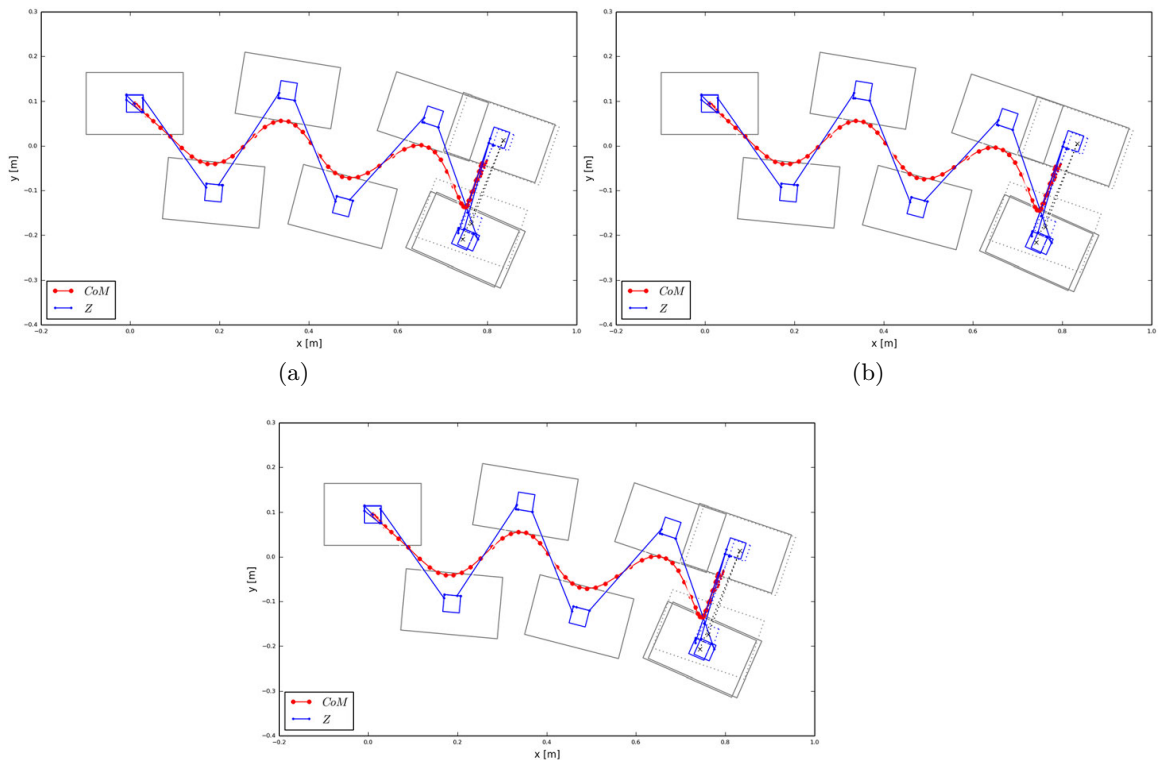


Figure 4.12: Foot trajectories when rotation is involved. MPC with Laguerre functions is the fastest controller. (a) Patterns by MPC. (b) Patterns by MPC with Haar functions. (c) Patterns by MPC with Laguerre functions.

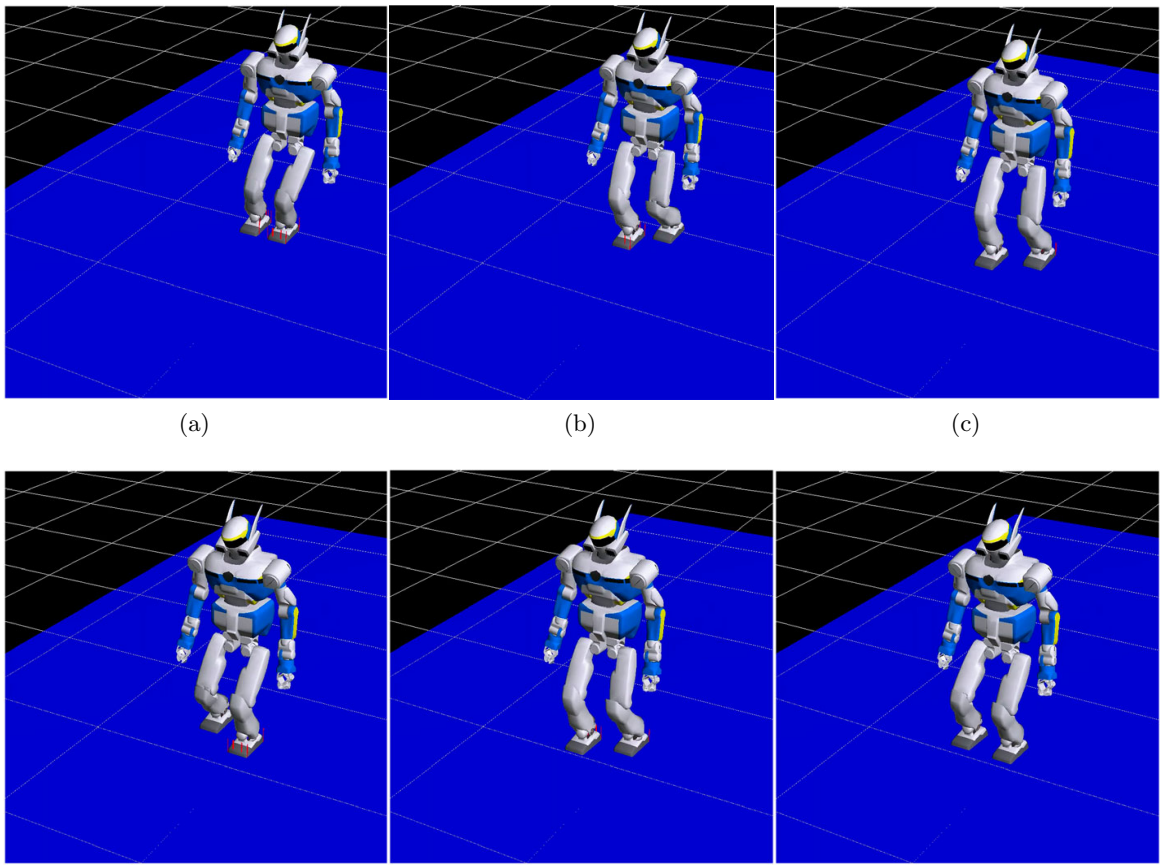


Figure 4.13: Bipedal walking using the WPG with Laguerre basis functions. (a) $t = 0s$ (b) $t = 0.8s$ (c) $t = 2.0s$ (d) $t = 3.0s$ (e) $t = 4.0s$ (f) $t = 5.0s$

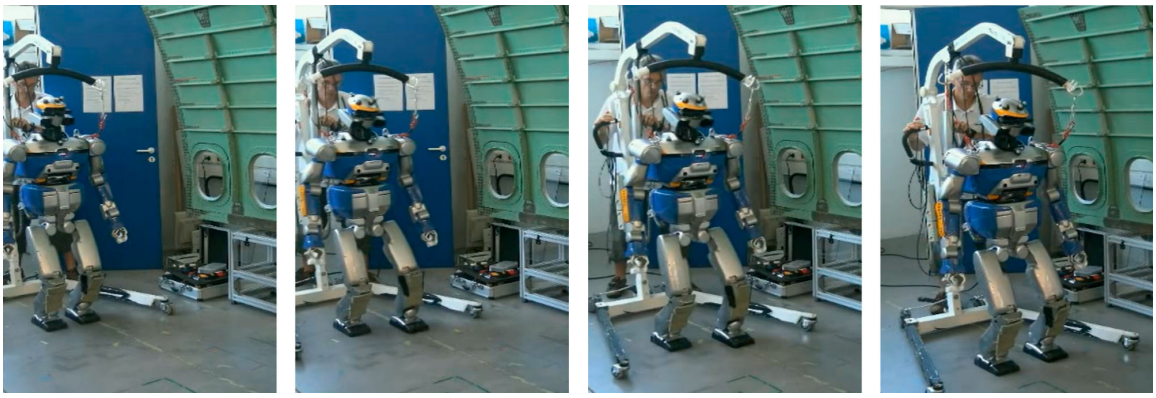


Figure 4.14: The experiment of walking forward when using the WPG with Laguerre basis functions.

and $V_{k+1}^{ref} = \begin{bmatrix} 0.2 & 0 & -0.2 \end{bmatrix}$ are shown in Figures 4.14 and 4.15, respectively.

4.6 Conclusion

The main idea of this work is to introduce basis functions into the design of control inputs for online WPG based on MPC. The control input represented by basis functions, like Haar functions and Laguerre functions, requires less free variables therefore the computational efficiency is improved. Simulations and experiments showed that feasible CoM trajectories can be generated while computational time can be reduced in both orientation and position controller when using basis functions. The MPC with Laguerre basis functions has the fastest performance because the configuration of Laguerre functions can be tuned to fit the property of the control input. We also showed that the MPC with Haar functions can always provide feasible and accurate solutions. The basis functions are not restricted to Laguerre functions and Haar functions, researchers can choose different basis functions for various purposes.

4.7 Future work

More basis functions to approximate control input could be tested. Also more detailed analysis about experiment results would be realized. With a fast walking pattern generator, we can let humanoids challenge more complex tasks while walking, like manipulation and cooperation with humans. As mentioned in Chapter 1, this computation time is decisive especially if we want to create reactive behavior based on exteroceptive sensors.

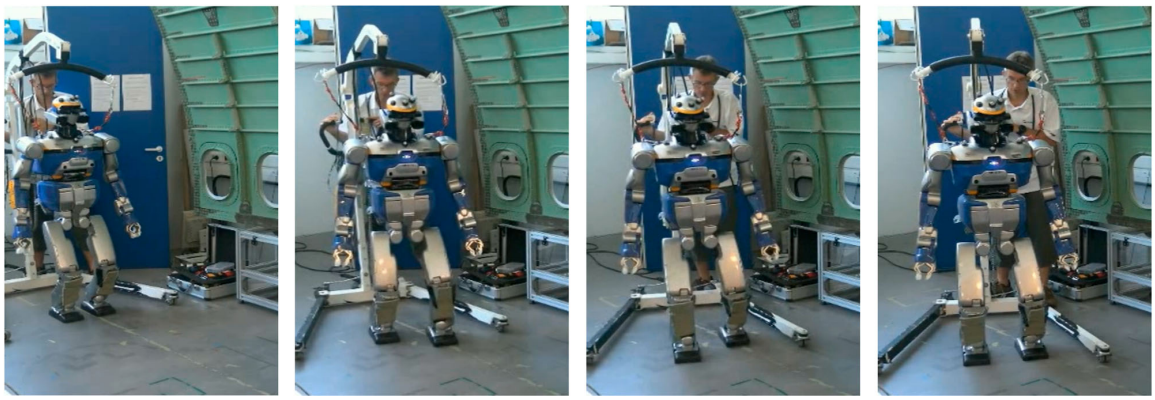


Figure 4.15: The experiment of walking forward and turning right when using the WPG with Laguerre basis functions.

Chapter 5

Visual servoing in Differential Dynamic Programming

Contents

We have seen in the previous chapter a way to speed-up the computation of a Pattern Generator, but a feedback loop based on exteroceptive sensors is still needed to correct the disturbances and make the motions more robust. Section 1.3 presented a way to embed visual information directly in an optimisation problem for generating locomotion. Visual information was included in a Pattern Generator algorithm. But it was not multicontact capable. Referring to the decomposition workflow described in section 1.2 and the choice explained in subsection 1.3.10, I will present in this chapter the tests realized using a non-linear and multicontact capable solver called DDP (Differential Dynamic Programming) used as a Whole Body Trajectory Generator (WBTG), see Fig. 1.2. This chapter is based on a conference paper for which I am the first author. This paper has been published in SII 2020.

5.1 Introduction

The expectations for a humanoid robot stand in its capability to navigate in structured and unstructured environments, potentially using other parts of its body than only feet. It should also fulfill tasks of manipulation with its upper body

for instance. These expectations imply the use of multiple loops of control on different sensors. Specifically, the exteroceptive sensors like cameras are needed to react to an environment modifications or to modeling errors. From the results of the DRC (Darpa Robotic Challenge) it appears that the methods available in the robotic field do not live up to these expectations. Considering assumptions or simplifications in either the environment or robot models, a part of the goal has been reached. Assuming flat floor, absence of obstacle and only biped locomotion (without multicontact), a significant body of work exists to use a representation of the environment, plan foot steps and execute them on humanoid robots of various sizes [Naveau 2017, Missura 2016, Imanishi 2018, Hildebrandt 2015]. An approach could be to use Model Predictive Control (MPC) on a Linear Inverted Pendulum where footsteps, Center-of-Mass (CoM) and Center-of-Pressure (CoP) trajectories are solved together with only the desired CoM velocity as input. Often, to reach online execution of the algorithm, the model of the robot dynamics is simplified as Angular Momentum is not taken into account in the centroidal dynamics (non-linearity). For instance, the model can be kinematically expressed in an Optimal Controller problem (less time consuming) and then rectified with a dynamic filter as in [Naveau 2017] that calculates dynamics errors and corrects the previous solution.

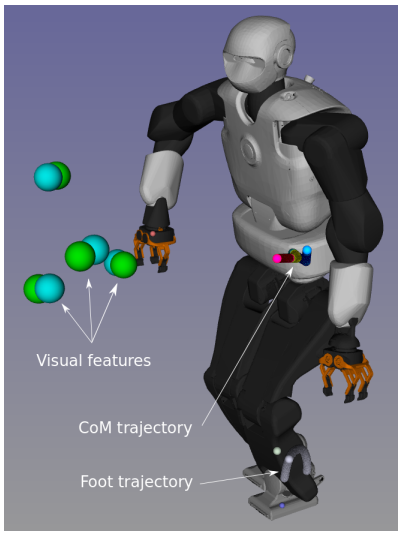


Figure 5.1: Resulting position of the robot after simulation with three contacts and visual task. TALOS' right hand (frame represented by a small red point) is equal with the target one. Center of mass trajectories are displayed: each color represents a phase corresponding to a contact change. Big green spheres represent referenced features for the visual task, blue ones for the output last position.

To get closer to the initial expectations, the CoM velocity can be driven by desired visual features as done in [Dune 2010] where the authors tackle the sway motion generated by walk oscillations. In addition to all the previous assumptions, the CoM velocity cannot always be achieved due to the constraints of the foot placements and

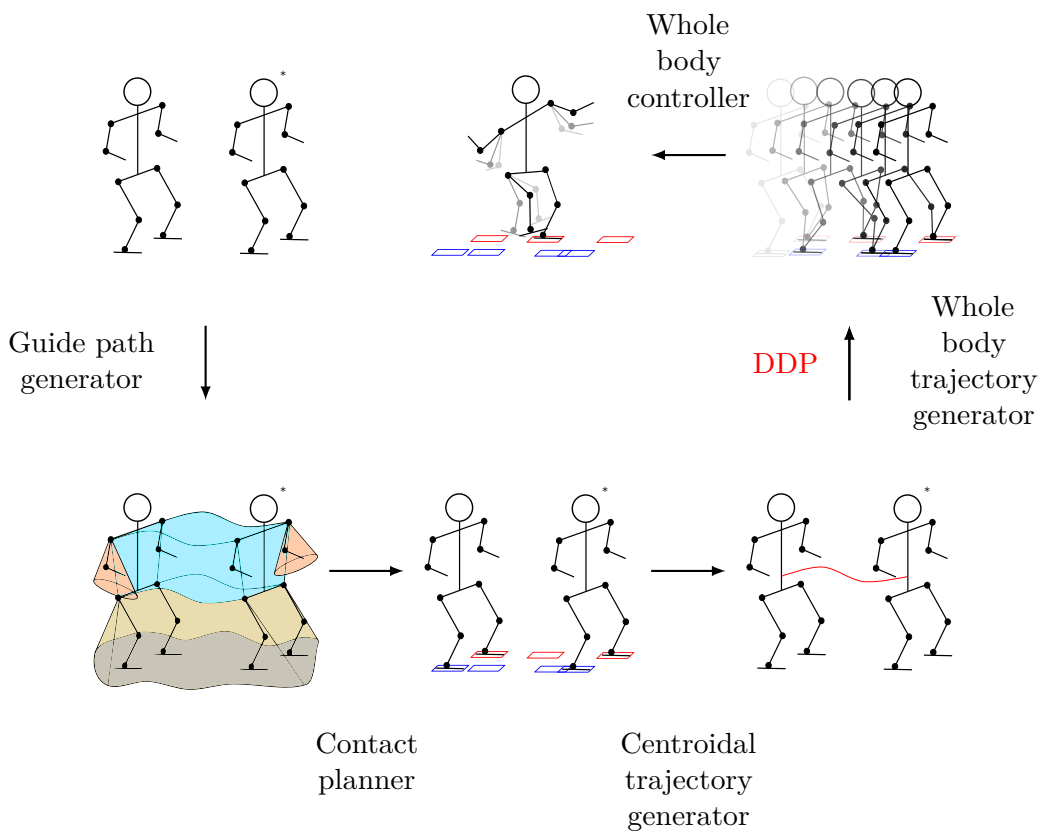


Figure 5.2: Overall approach: The guide path generator (also called reachability planner) takes a starting configuration ($\mathcal{C}_S$) and a goal configuration $\mathcal{C}_G$. The motion planner described in section 1.2 provides a contact sequence, and a centroidal dynamics trajectory (it works as a Contact planner and a centroidal trajectory genrator). The Differential Dynamic Programming (DDP) described in section 5.2 generates a whole body trajectory which is consistent with the contact dynamics and the complete model of the robot.

the balance which are of high priority. Another work with exteroceptive sensors copes with dynamical model simplification: in [Ramirez-Alpizar 2016] HRP-2 humanoid robot was able to carry a fire hose while walking. The system was using an external localization by motion capture feedback and a real-time pattern generator able to decide by itself foot step locations and generate a balanced Center-of-Mass trajectory.

The final goal is to remove the assumptions and simplifications commonly made, such as assuming a flat floor, convex obstacles, a gaited motion, ignoring the self-collisions, approximating the dynamic model, ... Being able to leverage such functionalities with a computation time suitable for online execution is quite challenging.

As described in 1.2, a common approach is to decouple the entire problem

in smaller sub-problems solved sequentially as shown in figure 5.2, that could be handled more efficiently [Carpentier 2016]. From a desired final position, the Planner (first and second block) provides steps and contact locations to a Centroidal Dynamics generator (third block) giving CoM dynamical trajectories to a Whole Body Trajectory Generator (fourth block) that generates joint trajectories. Then, these joint trajectories are sent to a Stabilizer looped on robot motions.

Exteroceptive sensors providing SLAM or visual references could be looped on these blocks at different levels. In this more general setup, combination of these blocks is very tough as highlighted during the DRC [Spe 2017]. Recent work has proposed a solution to introduce multiple contacts together with vision in [Tanguy 2016]. The authors use this decoupled approach to generate a motion in simulation. The main drawback lies in the use of local Quadratic Programming formulation and the assertion of an axis of forces on CoM dynamics that makes total Centroidal Dynamics linear (even on angular momentum).

The aim of our work, our first contribution, is to build and evaluate this kind of generic motion generation pipeline for a TALOS humanoid robot. We test all the pipeline from the motion planner to the stabilizer block on simulation.

The second contribution of this work lies in the connection of visual features in this workflow. We decided to use visual tasks in image plane as input of the Whole Body Trajectory Generator. This choice was motivated by the fact that visual features are directly embedded in the generator that can locally manage modifications of the motion accordingly to those features. Moreover the chosen generator can cope with the non-linearities produced by the projection in the image plane on all the time horizon (contrary to [Garcia 2014] that first order linearizes projections around the first point of the trajectory). Indeed, to tackle multicontact objective and non-linearities of the dynamics such as angular momentum equations, we decided to use Differential Dynamic Programming (DDP) solver that is an Optimal Control algorithm managing non-linearities on the state function. It was described in [Mayne 1966]. More recently, it has been used successfully for the DRC in [Yamaguchi 2015], and also proposed for humanoid robots in [Erez 2013]. In [Budhiraja 2018], the DDP is used to generate whole body motions (corresponding to the fourth subpart previously mentioned).

In this work we report our first tests in integrating a fast multicontact planner used to set a DDP problem which in turns provides reference trajectories to a local whole body instantaneous controller (stabilizer block, being part of subsection 1.2.5). It was tested in dynamical simulation (Gazebo/ODE) on the TALOS humanoid robot. In Section 5.2 we give some reminders about DDP algorithm and visual tasks elements to show how it can be integrated. Experiments and results are shown respectively in Section 5.3 and 5.4 on a TALOS robot with multicontact and visual tasks.

5.2 DDP and visual servoing

In this section we describe our visual servoing approach under multi-contact events based on DDP. For that, we first introduce our DDP algorithm tailored to multiphase rigid dynamics [Budhiraja 2018]. And later, we explain the visual task formulation within our multi-contact DDP. This work is based on the DDP solver implemented in Crocoddyl [Mastalli 2019], which computes efficiently the rigid body dynamics and its derivatives using Pinocchio [Carpentier 2019].

5.2.1 Differential dynamic programming

DDP belongs to the family of Optimal Control (OC) and trajectory optimization [Mayne 1966]. It locally approximates the optimal flow (feedback gains), and as a consequence, the OC problem is split into simpler and smaller subproblems (sparse structure). The DDP promises to handle whole-body MPC on a humanoid thanks to its sparse structure [Erez 2013]. However, the main drawback lies on the fact that it poorly handles constraints.

Let us consider a generic multi-contact OC problem as follows:

$$\begin{aligned} \mathbf{X}^*, \mathbf{U}^* &= \arg\min_{\mathbf{X},\mathbf{U}} l_T(\mathbf{x}_N) + \sum_{k=0}^{T-1} l_k(\mathbf{x}_k, \mathbf{u}_k) \\ \text{s.t.} \quad & \mathbf{x}_0 = \tilde{\mathbf{x}}_0, \\ & \mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k, \mathbf{u}_k), \end{aligned} \tag{5.1}$$

where T is the given horizon, the state $\mathbf{x} = (\mathbf{q}, \mathbf{v})$ lies in a Lie manifold with $\mathbf{q} \in SE(3) \times \mathbb{R}^{n_j}$ and $\mathbf{v} \in T_{\mathbf{x}}\mathcal{Q}$, $\tilde{\mathbf{x}}_0$ is the initial condition, the system is underactuated $\mathbf{u} = (\mathbf{0}, \boldsymbol{\tau})$ with $\boldsymbol{\tau}$ are the torque commands, the discrete dynamics $\mathbf{f}_k(\cdot)$ describes different contact phases, and $l_k(\mathbf{x}_k, \mathbf{u}_k)$ describes the different tasks (or running costs) and $\mathbf{X} = \{\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_T\}$ and $\mathbf{U} = \{\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_{T-1}\}$ are the tuple of states and controls along the defined horizon. Note that both – cost and dynamics – often are time varying functions.

DDP breaks the dynamic problem into simpler subproblem thanks to the "Bellman's principle of optimality". Indeed, moving backward in time, the approximated value function $V(\cdot)$ can be found by minimizing the local policy for a given node, i.e.

$$V_k(\delta\mathbf{x}_k) = \min_{\delta\mathbf{u}_k} l_k(\delta\mathbf{x}_k, \delta\mathbf{u}_k) + V_{k+1}(\mathbf{f}_k(\delta\mathbf{x}_k, \delta\mathbf{u}_k)), \tag{5.2}$$

and this is locally approximated by a quadratic function (a.k.a. as Gauss-Newton approximation) as follows:

$$\begin{aligned} &\delta\mathbf{u}_k^*(\delta\mathbf{x}_k) = \\ &\arg\min_{\delta\mathbf{u}_k} \frac{1}{2} \begin{bmatrix} 1 \\ \delta\mathbf{x}_k \\ \delta\mathbf{u}_k \end{bmatrix}^T \begin{bmatrix} 0 & \mathbf{q}_{\mathbf{x}_k}^T & \mathbf{q}_{\mathbf{u}_k}^T \\ \mathbf{q}_{\mathbf{x}_k} & \mathbf{q}_{\mathbf{x}\mathbf{x}_k} & \mathbf{q}_{\mathbf{x}\mathbf{u}_k} \\ \mathbf{q}_{\mathbf{u}_k} & \mathbf{q}_{\mathbf{x}\mathbf{u}_k}^T & \mathbf{q}_{\mathbf{u}\mathbf{u}_k} \end{bmatrix} \begin{bmatrix} 1 \\ \delta\mathbf{x}_k \\ \delta\mathbf{u}_k \end{bmatrix}, \end{aligned} \tag{5.3}$$

where $\delta\mathbf{x} = \bar{\mathbf{x}} \ominus \mathbf{x}$ is the deviation with respect to the local linearization $\bar{\mathbf{x}}$ and belongs to the tangential space ($\in T_{\mathbf{x}}\mathcal{Q}$), and the Jacobian and Hessian of the Hamiltonian are defined as:

$$\begin{aligned}
\mathbf{q}_{\mathbf{x}_k} &= \mathbf{l}_{\mathbf{x}_k} + \mathbf{f}_{\mathbf{x}_k}^T V_{\mathbf{x}_{k+1}}, \\
\mathbf{q}_{\mathbf{u}_k} &= \mathbf{l}_{\mathbf{u}_k} + \mathbf{f}_{\mathbf{u}_k}^T V_{\mathbf{x}_{k+1}}, \\
\mathbf{q}_{\mathbf{xx}_k} &= \mathbf{l}_{\mathbf{xx}_k} + \mathbf{f}_{\mathbf{x}_k}^T V_{\mathbf{xx}_{k+1}} \mathbf{f}_{\mathbf{x}_k}, \\
\mathbf{q}_{\mathbf{xu}_k} &= \mathbf{l}_{\mathbf{xu}_k} + \mathbf{f}_{\mathbf{x}_k}^T V_{\mathbf{xx}_{k+1}} \mathbf{f}_{\mathbf{u}_k}, \\
\mathbf{q}_{\mathbf{uu}_k} &= \mathbf{l}_{\mathbf{uu}_k} + \mathbf{f}_{\mathbf{u}_k}^T V_{\mathbf{xx}_{k+1}} \mathbf{f}_{\mathbf{u}_k}.
\end{aligned} \tag{5.4}$$

We obtain the local policy by solving the Quadratic Programming (QP) (5.3) as:

$$\delta\mathbf{u}_k^*(\delta\mathbf{x}_k) = \mathbf{k}_k + \mathbf{K}_k \delta\mathbf{x}_k \tag{5.5}$$

where $\mathbf{k}_k = -\mathbf{q}_{\mathbf{uu}_k}^{-1}\mathbf{q}_{\mathbf{u}_k}$ and $\mathbf{K}_k = -\mathbf{q}_{\mathbf{uu}_k}^{-1}\mathbf{q}_{\mathbf{ux}_k}\delta\mathbf{x}$ are the feedforward and feedback terms, respectively. And for the next node, we update the quadratic approximation of the value function by injecting $\delta\mathbf{u}_k^*$ expression into (5.3):

$$\begin{aligned}
\Delta V(i) &= -\frac{1}{2}\mathbf{q}_{\mathbf{u}_k}\mathbf{q}_{\mathbf{uu}_k}^{-1}\mathbf{q}_{\mathbf{u}_k} \\
V_{\mathbf{x}_k} &= \mathbf{q}_{\mathbf{x}_k} - \mathbf{q}_{\mathbf{u}_k}\mathbf{q}_{\mathbf{uu}_k}^{-1}\mathbf{q}_{\mathbf{ux}_k} \\
V_{\mathbf{xx}_k} &= \mathbf{q}_{\mathbf{xx}_k} - \mathbf{q}_{\mathbf{ux}_k}\mathbf{q}_{\mathbf{uu}_k}^{-1}\mathbf{q}_{\mathbf{ux}_k}
\end{aligned} \tag{5.6}$$

This backward pass allows us to compute the search direction during the numerical optimization. Then DDP runs a nonlinear rollout (a.k.a. forward pass) of the dynamics to try the computed direction along a step length α, i.e.

$$\begin{aligned}
\hat{\mathbf{x}}_0 &= \tilde{\mathbf{x}}_0 \\
\hat{\mathbf{u}}_k &= \mathbf{u}_k + \alpha\mathbf{k}_k + \mathbf{K}_k(\hat{\mathbf{x}}_k \ominus \mathbf{x}_k) \\
\hat{\mathbf{x}}_{k+1} &= \mathbf{f}_k(\hat{\mathbf{x}}_k, \hat{\mathbf{u}}_k)
\end{aligned} \tag{5.7}$$

in which we perform a typical *backtracking* line search by trying first the full step ($\alpha = 1$).

The DDP solver iterates on these two phases – backward and forward passes – until convergence to the result (gradient approximately equals zero).

5.2.2 Handling tasks and constraints

A task is usually formulated as a regulator:

$$h_{i,task}(\mathbf{x_i}, \mathbf{u_i}) = s_{task}^* - s_{task}(\mathbf{x_i}) \tag{5.8}$$

where s_{task}^* is a desired value vector for a feature and $s_{task}(\mathbf{x_i})$ the value vector of this feature according to state $\mathbf{x_i}$. As one wants to minimize this value such that

$lim_{t\to+\infty}h(x,u)=0$, the task at each node is implemented as a penalty:

$$l_i(\mathbf{x_i},\mathbf{u_i}) = \sum_{j\in Tasks} w_{i,j}h_{i,j}(\mathbf{x_i},\mathbf{u_i}) \tag{5.9}$$

with $w_{i,j}$ the weight assigned at time i to task j. In our case $Tasks \subseteq \{CoM, RH_{SE(3)}, RF_{SE(3)}, LF_{SE(3)}, EE_{se(3)}^{eeName}, VT\}$ with CoM the task tracking the Center-of-Mass, $RH_{SE(3)}$ the task tracking the right hand pose, $RF_{SE(3)}$ the task tracking the right foot pose, $LF_{SE(3)}$ the task tracking the left foot pose, $EE_{se(3)}^{eeName}$ the task tracking an end effector velocity which should be null during the impact, $eeName$ is the name of the end effector (RH for right hand for instance), VT the visual task expressed in the image plane.

5.2.3 Handling dynamic constraints

Although this basic formulation of DDP does not handle constraints it is possible to integrate them in the cost function using Lagrangian relaxation. Thus [Budhiraja 2018] modifies the problem formulation to enforce contacts. The dynamics of the robot are expressed as follows :

$$\mathbf{M}\dot{\boldsymbol{\nu}} = \boldsymbol{S\tau} - \boldsymbol{b} + \mathbf{J}_c\boldsymbol{\lambda} \tag{5.10}$$

with $\mathbf{M}$ the inertial matrix, $\dot{\boldsymbol{\nu}}$ the derivative of the state velocity, $\boldsymbol{S}$ the selection matrix corresponding to the actuated degrees of freedom (dof), $\boldsymbol{\tau}$ the vector of torques of actuated joints and $\mathbf{b}$ the bias term consisting in Coriolis and gravitational effects. $\mathbf{J}_c\boldsymbol{\lambda}$ is the term expressing the external forces at joint level. $\mathbf{J}_c$ is the stacked Jacobian corresponding to application points, $\boldsymbol{\lambda}$ is the positive value of the force applied to the application point, on the force application direction. In the formulation this term is viewed as the dual variables. To constrain the dynamic of the contact, [Budhiraja 2018] express null acceleration at the contact point by :

$$\begin{aligned} (\dot{\mathbf{J}_c v_c}) &= \mathbf{0} \\ &\Leftrightarrow \\ \mathbf{J}_c\dot{\mathbf{v}}_c &= -\dot{\mathbf{J}}_c\mathbf{v} \end{aligned} \tag{5.11}$$

With 5.11 and 5.10, using Gauss principle, KKT (*Karush-Kuhn-Tucker*) conditions are given by :

$$\begin{bmatrix} \mathbf{M} & \mathbf{J}_c^T \\ \mathbf{J}_c & \mathbf{0} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{\nu}} \\ -\boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{S\tau} - \boldsymbol{b} \\ \dot{\mathbf{J}}_c\mathbf{v} \end{bmatrix} \tag{5.12}$$

To take into account the dual variable in the resolution of the problem, the dynamic equation is augmented as follows :

$$\begin{aligned} \mathbf{x}_{i+1} &= \mathbf{f}(\mathbf{x_i},\mathbf{u_i}) \\ \boldsymbol{\lambda}_i &= \mathbf{g}(\mathbf{x_i},\mathbf{u_i}) \end{aligned} \tag{5.13}$$

where g is the dual solution of 5.12. The action-value function Q and the cost l are now depending on $\boldsymbol{\lambda}$. Making the assumption that second derivative of the dynamic f are zero as ILQR algorithm does, but taking into account the derivative of the cost l by $\boldsymbol{\lambda}$, the new equations of the second order approximation are :

$$Q_x = l_x + f_x^T V_x' + g_x^T l_\lambda \tag{5.14}$$
$$Q_u = l_u + f_u^T V_x' + g_u^T l_\lambda \tag{5.15}$$
$$Q_{xx} = l_{xx} + f_x^T V_{xx}' f_x + g_x^T l_{\lambda\lambda} g_x \tag{5.16}$$
$$Q_{uu} = l_{uu} + f_u^T V_{xx}' f_u + g_u^T l_{\lambda\lambda} g_u \tag{5.17}$$
$$Q_{ux} = l_{ux} + f_u^T V_{xx}' f_x + g_u^T l_{\lambda\lambda} g_x \tag{5.18}$$

This method takes into account the contact constraints in the dynamic level and prevents the solver to allocate resources to manage these constraints during solving run. Since the main principles underlying DDP are exposed in this paragraph, visual servoing is briefly presented in the next paragraph in order to derive its integration and implementation.

5.2.4 Visual servoing

As the DDP algorithm needs residuals (or regulators) and derivatives of the tasks, this paragraph describes the formulation of the visual task and its derivatives.

Given the type of sensor / camera, the formulation of a visual task can differ. If the sensor provides depth information, the approach is called Point-Based Visual Servoing (PBVS). The formulation of that kind of task lies in SE(3) space. If the camera does not provide depth information (or if that data is not trustful due to errors, bias, noise), one will use the Image Based Visual Servoing (IBVS). This approach is detailed here.

Let us first consider the desired features $s*$ and the actual features s. These last could refer to perceived information from camera or calculated by a simulator. The features can be points of interests, moments or more complex visual features. For the sake of simplicity this study consider the simpler case of points.

The error of the task is then :

$$e = s - s* \tag{5.19}$$

In our case, $s*$ is considered as fixed, not depending on time. The error e is also considered as the residual of the cost l defined by :

$$l = \frac{1}{2} \|e\|^2 \tag{5.20}$$

The model commonly used is a first order motion model:

$$\dot{e} = L_e v_c \tag{5.21}$$

where v_c is the velocity of the camera in the camera frame, and Le is the interaction matrix. This matrix can be considered as the features Jacobian By derivating the position of one feature in the 3D space, [Chaumette 2006] has shown Le can be written as follows :

$$L_e = \begin{bmatrix} \frac{-1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & \frac{-1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{bmatrix} \tag{5.22}$$

Now, let us call J_c the Jacobian of the camera in the camera frame, and $\dot{q}$ the velocity of the degrees of freedom (DoF). Combining the well known expression $v_c = J_c\dot{q}$ with (5.21), we find :

$$\dot{e} = L_e J_c \dot{q} \tag{5.23}$$

Contrary to the common visual servoing control law that enforces the exponential decrease by writing this relation: $\dot{\mathbf{e}} = -\lambda\mathbf{e}$, DDP needs the derivative of the task with respect to the state $\mathbf{x}$ and the control $\mathbf{u}$ as expressed in (5.4). As mentioned earlier, the state is composed of the robot configuration $\mathbf{q}$ and of its joint-space velocity $\dot{\mathbf{q}}$, and its Jacobians are:

$$\frac{\partial e}{\partial \begin{bmatrix} q \\ \dot{q} \end{bmatrix}} = \left[L_e J_c, 0_{2\times n_{\dot{q}}} \right] \tag{5.24}$$

$$\frac{\partial e}{\partial u} = 0_{2\times n_u} \tag{5.25}$$

The Hessian of the visual task (i.e. $l_{\mathbf{xx}}$, $l_{\mathbf{xu}}$ and $l_{\mathbf{uu}}$) are zeros.
All the elements are gathered to implement the visual task in the DDP algorithm. Let us now explain the experimental conditions and tasks we have created for this work.

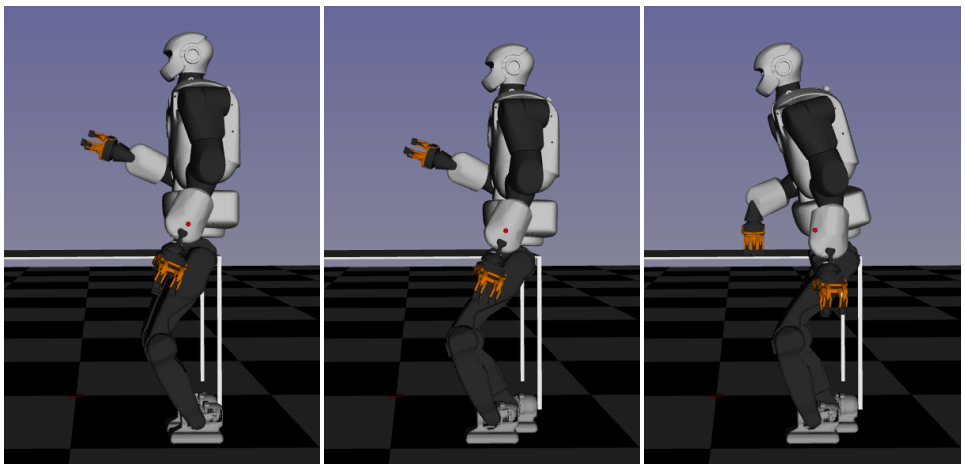
5.3 Simulations and experiments

In this section we describe the situation of the robot and the tasks it has to manage, the software architecture used to generate appropriate motion and the results obtained in simulation and then on the real robot.

5.3.1 Simulation setup

In our setup, Talos begins in an initial double support standing configuration. It should reach a contact surface (like a table) to create a third contact in order to bend sufficiently while maintaining balance to be able to see a target in its field of view. Then, keeping the three contacts, it should visual servo the target with predefined desired features positions in the image plane. The main goal here is to be able to see an object while the posture needs a third (or more) contact.

Figure 5.3 shows the output of the contact planner: a sequence of three configurations in contact, with one contact change between each configurations.



(a) (b) (c)

Figure 5.3: Sequence of configuration in contact produced by the contact planner. In (a) and (b) only the two feet are in contact, in (c) both feet and the right hand are in contact.

In Crocoddyl, for each time step the dynamic and the cost of the problem are redefined so that tasks can be independently managed following a predetermined time line given from the previous stages, namely the contact planner and the centroidal trajectory generation method. In that way, our time line is divided as follow:

- First phase set of tasks : $\{CoM\}$. A first phase to make the robot center of mass go down and on the right to be above the next foot of support. The CoM trajectory is followed through a task added in the cost function (through Lagrangian relaxation). The posture is regularized around the initial position (figure 5.3-a). Contacts are enforced on both feet in Eq.5.12.

- Second phase set of tasks : $\{CoM, LF_{SE(3)}, RH_{SE(3)}\}$. The second phase enforces only the right foot on the ground while a task is provided on the position of the left foot ($SE(3)$ task). This task is roughly constructed by interpolating the position of foot between initial and final position, with an offset of 10cm along the vertical axis. We see here that even if the reference for the foot position suffers from discontinuities, DDP can provide feasible foot trajectories for the foot. For collision avoidance reasons, a $SE(3)$ task for the hand with relatively low weight is provided (staying as the same place). The last point of this phase is one time step before the contact creation between the flying feet and the ground. Here the set of tasks is $\{CoM, RF_{SE(3)}, RH_{SE(3)}, EE^{LF}_{se(3)}\}$. It is augmented by an impact model that enforces again the double contact of the feet and manages the different tasks weights to improve the contact. For example, regulation and $SE(3)$ task weights are increased, $EE_{se(3)}$ task for the flying foot is provided with high

cost on null velocity reference. From this point, the position is regulated around the second configuration given by the planner (figure 5.3-b).

- Third tasks set is $\{CoM, RH_{SE(3)}\}$. Third phase is made similarly as the first one. We only bring a new $SE(3)$ task for the right hand, referenced by an interpolation between the hand position at the beginning of this phase and the contact point position provided by the contact-planner. The final point is managed as creation contact point like previously, enforcing three contacts. At this point tasks set is $\{CoM, RH_{SE(3)}, EE^{RH}_{se(3)}\}$.

- Fourth tasks set is $\{CoM\}$. Final phase is regulated around the next position from the contact planner (figure 5.3-c). CoM task is kept and the three contacts enforced. The final point is regulated around the last planner position and includes the visual task. For this point set tasks is $\{CoM, VT\}$. Even if it seems to appear lately in the time line, it does not make a noticeable difference in the resulting motion. The DDP propagates the image plane based non linear visual error on previous time steps, hence the motion is smooth and the task is solved up to the concurrent tasks solutions. The visual task is made from targets that are 3D space points and projected on the image plane of the camera by a pin-hole model. We need at least four points to avoid multiple possible solutions to place the camera with respect to the points and the references. In figure 5.1 the green balls are the references, the blue ones are how they are positioned at the end of the motion.

The DDP algorithm is shown to converge on tasks expressing a walking pattern with null initialisation of the problem (command and state over the time line). But in our case, the impact on the hand and the three contacts enforced did not allow to find a convergence without any good initialization (warm-start). The motion found is made iteratively by warm-starting the previous parts of the motion and letting null initialization for the next. For instance, in our case, the motion until the flying foot touching the ground was generated by solving the first phase alone with null initial guess, until time $t = T_{footTakeOff}$, and then solve the problem for first and second phase together, warmstarting from $t = 0$ to $t = T_{footTakeOff}$ with previous solution while initilization from $t = T_{footTakeOff}$ to $t = T_{footLanding}$ was null. Another heuristic was used to help the solver converge: the posture regulation weight has been set higher during the complete sequence convergence research, then turned lower to avoid high velocity motion during phase transitions.

Unfortunately, collision avoidance is currently not implemented in the DDP algorithm. To generate a motion which can be tried on the robot, we checked the bounds limits violation and self-collision for each time step. For that purpose we used the tools provided by the Humanoid Path Planner framework [Mirabel 2016]. However, if we found out that the motion produced by the DDP violate one of these constraints, we cannot directly add the constraint to the formulation of the problem in order to produce a valid motion. We found an iterative heuristic to avoid this issue: knowing that the reference configurations given by the planner are valid

and away from these bounds, we increased the weight of the postural task for the corresponding joints. In case of joint limit violation, we increased only the weight corresponding to that joint in postural task (regularization task). If an autocollision appeared, all the joints of the kinematics chain from that body to the torso are involved.

To that point, DDP algorithm generates the references for the next algorithm blocks: joint trajectories, feet trajectories and dynamic whole body CoM trajectory. To be consistent with the next section, we have to notice here that the CoM reference trajectory taking as input in the DDP algorithm is discretized at $100Hz$. The output is then naturally discretized at 100Hz too. The next block of code needs 1kHz as input, so then the trajectories were interpolated with the cubic mode of scipy interpolation, except joint trajectories that were interpolated in linear manner after the output. Until this point, all the verification were handled in the gepetto-viewer.

5.3.2 Control architecture

The motors of the robot are position-controlled. Rather than just sending the reference joint trajectory to the motors, we employ a stabilizing control scheme in order to improve the stability all along the motion. Note that the motion generated by the DDP alone did not work with the Gazebo simulator. This stabilization was necessary to make the simulation successful.

The DDP output is first decomposed into separate kinematic tasks, which are then sent to the hierarchical inverse-kinematics solver, namely the Stack of Tasks [Mansard 2009]. The tasks are, in decreasing order of priority:

- Pose of each foot
- Center of Mass position
- Upper body posture
- Waist orientation

It is important to notice that the order of priority of the tasks is crucial, as each task is projected in the null space of the previous one.

The dynamic stabilization is based on the Zero Moment Point (ZMP). We are applying the ZMP control by CoM acceleration strategy [Kajita 2014] as described in [Caron 2019]. First, the current CoM position and velocity are estimated from joint sensors readings. Then, a commanded ZMP reference is computed based on the deviation between the desired CoM and the estimated value. Further feedback is obtained from the force sensors in order to estimate the current ZMP. Finally, the CoM reference is corrected to achieve the desired ZMP. The stabilizer can be integrated seamlessly in the hierarchical inverse kinematics architecture, by simply replacing the desired CoM reference with the adjusted one.

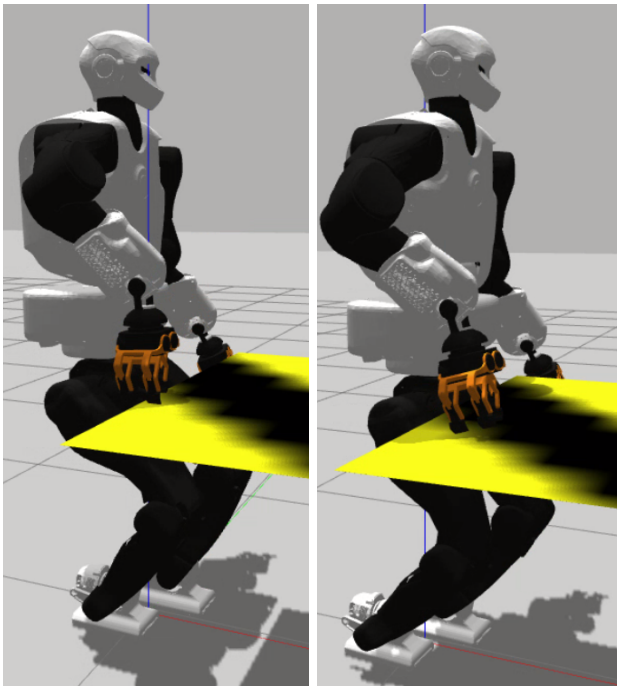


Figure 5.4: Left: The robot is touching the table too early. Right: After a little bound and slide, the hand and the robot reach desired positions

5.4 Results

We will now describe the results of this work. The DDP algorithm took several minutes for the sum of all phases, knowing that the motion lasts almost 9 seconds. The code is currently written in python and a work to implement a c++ version is ongoing, we expect an increase of performance from this future implementation. A first stage of simulation was made in a viewer called Gepetto-viewer. The algorithm is based on a weighted optimization process so errors of some tasks could remain. For instance, visual task with the relatively low weight suffers from several centimeters of errors for all the four points as it can be seen in Fig. 5.1 with big cyan and green spheres in front of the robot. The trajectories of the center of mass and the reference are also displayed, only one trajectory is visible because points are too close to be distinguished. Even if these two trajectories are very close, they are not perfectly equivalent for two reasons. Firstly, the task of the CoM struggles against other task and regularization during the optimization process. Secondly the DDP takes the complete dynamics of the system into account, contrary to the previous stages. So then, the DDP behaves as a dynamic filter without another calculation layer like in [Naveau 2017].
Concerning the simulation in a simulator, the motion was tested in Gazebo in the same way it would be tested on the real robot : Stack of Tasks controller, stabilizer and ROS architecture. The environment of simulation is a fixed plane positioned

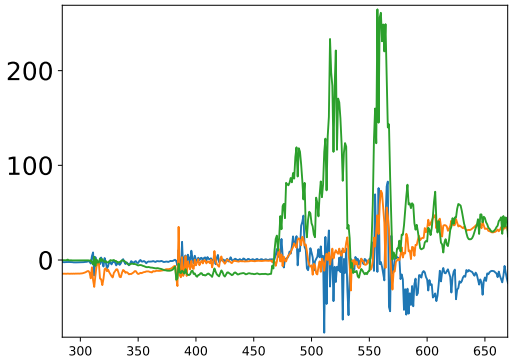


Figure 5.5: Blue, orange and green curves are respectively x, y and z forces on contact hand, got from simulation. Bounds are recognizable on z forces going to 0 after first contact with the table. Values are expressed in Newtons.

at $75cm$ from the ground. As shown in figure 5.4-Left the robot is first touching the table before having a little leap forward of the right gripper until final stable position displayed in figure 5.4-Right. This motion of the hand on the table is not expected and may be due to a lack of a $SE(3)$ hand task provided directly in stack of task controller. With the input reference configuration given to the DDP, the results shown in Fig. 5.5 indicate that the forces on the right gripper are around $50N$ at the end of the motion with $250N$ in peak on z axis.

5.5 Conclusion

We have generated a multicontact motion motivated by vision. The multicontact planner provides a feasible CoM trajectory to be followed and reference postures of phases corresponding to contact changes, used as input for the DDP algorithm. Allowing to solve non linear problems, it computes the complete dynamics of the robot and acts as a dynamic filter on the previous inputs. It also embeds the contact formulation directly in the dynamics.

By expressing its derivatives on state and control in the image plan, a visual task is integrated in the DDP to drive the motion to the target. The outputs of this algorithm, namely the joints and end effector trajectories are then sent to the stabilizer to be played in a Gazebo simulation through the Stack of Tasks hierarchical controller. The simulation shows a slight unexpected sliding of the hand on the table, nonetheless data show that force peaks are not prohibitive to play such a motion on the robot. We consider playing this motion on the real robot with appropriate experimental setup very soon.

Chapter 6

Conclusion

6.0.1 Contributions

Expectations of humanoid robots are manipulation and navigation. The main goals are to generate stable motions in structured or unstructured environments using multiple contacts (not only feet). Motion generation should be looped on exteroceptive feedback sensors such as cameras.

In Chapter 1, we have seen why the locomotion problem is complex and how it can be modeled to solve it. I have especially shown a decoupled structure used in the Gepetto team for dividing the problem in sub-problems easier to handle. From the actual and the reference positions of the robot, the planner block (separated into two blocks in Chapter 1 : the Guide path generator and the Contact planner) generates a feasible contact sequence. This is given to the centroidal trajectory generator that provides a dynamical consistent trajectory of the robot center of mass corresponding to this contact sequence. In turn, these information are used to generate joints trajectories in the whole body trajectory generator. Those trajectories are given to the whole body controller in charge of stabilizing the robot during the locomotion. In this chapter, some prior related works about walk generation with ZMP equations model on a time horizon are presented. Methods were improved by solvers using constraints and allowing the foot-prints to be found as they were passed in free variables of the problem. Last improvement in this approach was to embed non-linearities in the problem. On the other hand, vision servoing for driving locomotion was not embedded in the locomotion problem with its non-linearities. These elements were the founding basements of the topic of this thesis: taking visual information to correct locally the whole body motion, which could be in multicontact.

Chapter 2 deals with mechanical phenomena observed on the different robots of the team, that induces strong modification in the control loops. It highlights the different flexibilities of the robotic platforms: the flexibilties that must be controlled (in the sole of HRP-2 and in the hip of Pyrène) and the flexibilities that can help for torque estimation (in the Romeo actuator). It points out elements that complexify the control as acyclic friction phenomena in the actuator, kinematics or power limitations.

These two chapters (1 and 2) introduce necessary elements to make the following chapters clearer. Indeed they deal with my contributions.

KoroiBot project led to two contributions in Chapter 3. The first aims at generating intermediate motions to place the robot in front of different obstacles to be crossed (stairs and beam). The loop closure was provided by exteroceptive

robot position information given from a motion capture system. A simple contact sequence generator has been implemented to feed a pattern generator available in the Gepetto team. Even if results were satisfying, a repeatability issue was observed on foot placements of the robot preventing it to cross the beam in a acceptable number of attempts. This did not allow us to reach the objective.

Secondly, the KoroiBot project led to a procedure to test the robot abilities. It defines Key Performance Indicators that we used to benchmark the robot HRP-2 on different platforms proposed by our collaborators of the laboratory of metrology and experiments (LNE). Using this KPI, I have compared the different algorithms available in the team at that time on the different platforms available (climatic room, slopes, pushes, weights, horizontal translating ground). Major outcomes are the temperature role in the energy consumption (the colder, the more expensive) and the injected energy increase near the limit of stability.

Another collaboration has borne fruits with the Graduate School of Engineering Science, Osaka University in Japan. The objective was to speed up computation of the walking pattern generators played with a model predictive control style by using a pulse basis functions for expressing the control variable trajectories. Decreasing the size of the basis where a trajectory is expressed, allows to decrease also the number of variables of the optimization problem. This results in a faster speed computation but also weaker precision in the trajectory generated compared to the full problem. Haar and Laguerre functions are compared to find a trade off between computation velocity and errors created in the trajectory. I was especially in charge of generating the whole body motion given from the contact sequence and the CoM trajectory resulting of this variation on traditional algorithms. I also realized the experiments on the robots which were successful but without any online implementation.

In the last Chapter 5, a choice was made between using a non-linear solver for planar walk that could be run online or a multicontact-able non-linear solver that was in progress in the team. The objective of embedding visual information would have been reachable in both cases. Our choice was the second option since generalized locomotion was a main objective of this thesis, this ongoing solver was a part of the workflow of the team and because its performances for being played online are promising. Indeed the DDP uses the sparsity of the problem. Its main drawback remains in poorly handling constraints. No implementation of constraints was available when I was using it. I provided visual task implementation in this framework. Using output of the previous blocks of the pipeline described in Chapter 1, the DDP acts as a whole body trajectory generator. I integrated that block with the stabilizer. Results were generated on simulation with a step and a contact on a table with visual tasks.

6.0.2 Future work and expectations

The expectations for humanoid robots presented in the introduction of this thesis are clearly out of reach for now. For instance, most of the algorithms presented in Chapter 1 are computed offline, and the visual feed back loop in the locomotion

decomposition is only an ongoing work. This means that the targeted generalized locomotion is not mature enough. Two main approaches have been evoked in this thesis. A bottom-up approach with pattern generators that are online but do not provide multicontact functionality. They can also be driven by visual features but this topic needs improvements. A potential solution would be to extend these pattern generators to take into account 3D contact formulation as well as constraints like collisions, autocollisions, singularities or torque limits. The second approach, that can be seen as top-down approach, is to rely on improvements of the planner computational speed to be able to replan interactively the trajectories to be followed by the whole body controllers. An improvement could be warmstarting the problem with a good guess. Indeed a bench of motions can be generated and stored to feed a machine learning process. It could then fastly give a correct guess to the optimization problems that should speed up significantly the computation. This approach is taken in the European project MEMMO for instance.

Whatever the chosen approach, this should involve one or multiple feed back loops on exteroceptive sensors such as cameras. The motion needs to be reactive to the environment and the potential drifts of the robot. Again, the integration effort is long and can raise issues when running on the real robots. We have evoked in this thesis some calibration problems as the mechanical wear (see subsection 2.3.4). Mechanics can induce more unexpected phenomena as seen in the subsection 2.2.2.2. These acyclic, angle dependent, joint frictions could be learned for instance by machine learning during a calibration test or during the run by refining the model. Besides updating the model corresponding to the wear of the mechanical parts, it could also send warnings to the operator in case of mechanical breakdown. This could become a failure detection system.

Such a detection seems important for industrial or personal assistance purposes. Indeed, I personally hope the technologies developed all along this thesis could be extended for example to exoskeletons or prosthetic devices for disabled people.

Bibliography

[Alfayad 2011] Samer Alfayad, Fethi B Ouezdou, Faycal Namoun and Gordon Gheng. *High performance integrated electro-hydraulic actuator for robotics–Part I: Principle, prototype design and first experiments.* Sensors and Actuators A: Physical, vol. 169, no. 1, pages 115–123, 2011. (Cited in page 30.)

[Allibert 2008] Guillaume Allibert, Estelle Courtial and Youssoufi Touré. *Visual predictive control for manipulators with catadioptric camera.* In 2008 IEEE International Conference on Robotics and Automation, pages 510–515. IEEE, 2008. (Cited in page 21.)

[Benallegue 2015] Mehdi Benallegue, Alexis Mifsud and Florent Lamiraux. *Fusion of force-torque sensors, inertial measurements units and proprioception for a humanoid kinematics-dynamics observation.* In 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids), pages 664–669. IEEE, 2015. (Cited in page 34.)

[Bohórquez 2017] Néstor Bohórquez and Pierre-Brice Wieber. *Adaptive step duration in biped walking: a robust approach to nonlinear constraints.* In 2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids), pages 724–729. IEEE, 2017. (Cited in page 76.)

[Bouyarmane 2017] K. Bouyarmane, S. Caron, A. Escande and A. Kheddar. Humanoid robotics: a reference. Spring, 2017. (Cited in page 28.)

[Brandao 2017] M. Brandao, K. Hashimoto and A. Takanishi. *SGD for robot motion? The effectiveness of stochastic optimization on a new benchmark for biped locomotion tasks.* In Int. Conf. on Humanoid Robotics, 2017. (Cited in page 48.)

[Budhiraja 2018] Rohan Budhiraja, Justin Carpentier, Carlos Mastalli and Nicolas Mansard. *Differential dynamic programming for multi-phase rigid contact dynamics.* In 2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids), pages 1–9. IEEE, Nov 2018. (Cited in pages 100, 101, and 103.)

[Caron 2019] Stéphane Caron, Abderrahmane Kheddar and Olivier Tempier. *Stair Climbing Stabilization of the HRP-4 Humanoid Robot using Whole-body Admittance Control.* In IEEE International Conference on Robotics and Automation, May 2019. (Cited in pages 13, 40, and 108.)

[Carpentier 2016] J. Carpentier, S. Tonneau, M. Naveau, O. Stasse and N. Mansard. *A Versatile and Efficient Pattern Generator for Generalized Legged Locomotion.* In Int. Conf. on Robotics and Automation, 2016. (Cited in pages 9, 27, 62, 63, 64, 65, 72, 73, and 100.)

[Carpentier 2019] Justin Carpentier, Guilhem Saurel, Gabriele Buondonno, Joseph Mirabel, Florent Lamiraux, Olivier Stasse and Nicolas Mansard. *The Pinocchio C++ library – A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives.* In IEEE International Symposium on System Integrations (SII), 2019. (Cited in page 101.)

[Chaumette 2006] François Chaumette and Seth Hutchinson. *Visual servo control. I. Basic approaches.* IEEE Robotics & Automation Magazine, vol. 13, no. 4, pages 82–90, 2006. (Cited in pages 20, 21, and 105.)

[Chen 2018] Ming Chen, Ang Zhang and Kil To Chong. *A novel controller design for three-phase voltage source inverter.* International Journal of Control, Automation and Systems, vol. 16, no. 5, pages 2136–2145, 2018. (Cited in page 77.)

[Cherubini 2011] Andrea Cherubini, François Chaumette and Giuseppe Oriolo. *Visual servoing for path reaching with nonholonomic robots.* Robotica, vol. 29, no. 7, pages 1037–1048, 2011. (Cited in page 19.)

[Chestnutt 2010] Joel Chestnutt. Navigation and gait planning, pages 1–28. Springer London, 2010. (Cited in page 44.)

[Clever 2017] Debora Clever, Monika Harant, Katja Mombaur, Maximilien Naveau, Olivier Stasse and Dominik Endres. *Cocomopl: A novel approach for humanoid walking generation combining optimal control, movement primitives and learning and its transfer to the real robot hrp-2.* IEEE Robotics and Automation Letters, vol. 2, no. 2, pages 977–984, 2017. (Cited in page 49.)

[DeDonato 2017] Mathew DeDonato, Felipe Polido, Kevin Knoedler, Benzun P. W. Babu, Nandan Banerjee, Christoper P. Bove, Xiongyi Cui, Ruixiang Du, Perry Franklin, Joshua P. Graff, Peng He, Aaron Jaeger, Lening Li, Dmitry Berenson, Michael A. Gennert, Siyuan Feng, Chenggang Liu, X Xinjilefu, Joohyung Kim, Christopher G. Atkeson, Xianchao Long and Taşkın Padır. *Team WPI-CMU: Achieving Reliable Humanoid Behavior in the DARPA Robotics Challenge.* Journal of Field Robotics, vol. 34, no. 2, pages 381–399, 2017. (Cited in page 44.)

[Deits 2014] Robin Deits and Russ Tedrake. *Footstep Planning on Uneven Terrain with Mixed-Integer Convex Optimization.* In Int. Conf. on Humanoid Robotics, 2014. (Cited in page 48.)

[Del Pobil 2006] Angel P Del Pobil, Rad Madhavan and Elena Messina. *Benchmarks in robotics research.* In Workshop IROS, 2006. (Cited in page 48.)

[Diehl 2001] M. Diehl, D. B. Leineweber and A. Schäfer. *MUSCOD-II Users' Manual.* Universität Heidelberg, 2001. (Cited in pages 63 and 70.)

[Dune 2010] Claire Dune, Andrei Herdt, Olivier Stasse, P-B Wieber, Kazuhito Yokoi and Eiichi Yoshida. *Cancelling the sway motion of dynamic walking in visual servoing.* In 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 3175–3180. IEEE, 2010. (Cited in pages 18, 20, 21, and 98.)

[Englsberger 2011] Johannes Englsberger, Christian Ott, Máximo A Roa, Alin Albu-Schäffer and Gerhard Hirzinger. *Bipedal walking control based on capture point dynamics.* In 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 4420–4427. IEEE, 2011. (Cited in page 76.)

[Englsberger 2014] Johannes Englsberger, Alexander Werner, Christian Ott, Bernd Henze, Maximo A Roa, Gianluca Garofalo, Robert Burger, Alexander Beyer, Oliver Eiberger, Korbinian Schmid*et al. Overview of the torque-controlled humanoid robot TORO.* In 2014 IEEE-RAS International Conference on Humanoid Robots, pages 916–923. IEEE, 2014. (Cited in pages 26, 27, 28, and 29.)

[Englsberger 2015] Johannes Englsberger, Christian Ott and Alin Albu-Schäffer. *Three-dimensional bipedal walking control based on divergent component of motion.* IEEE Transactions on Robotics, vol. 31, no. 2, pages 355–368, 2015. (Cited in page 76.)

[Erez 2013] T. Erez, K. Lowrey, Y. Tassa, V. Kumar, S. Kolev and E. Todorov. *An integrated system for real-time Model Predictive Control of humanoid robots.* In IEEE/RAS Int. Conf. on Humanoid Robotics (ICHR), 2013. (Cited in pages 100 and 101.)

[Feng 2016] Siyuan Feng, X Xinjilefu, Christopher G Atkeson and Joohyung Kim. *Robust dynamic walking using online foot step optimization.* In 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 5373–5378. IEEE, 2016. (Cited in page 27.)

[Fernbach 2017] P. Fernbach, S. Tonneau, A. Del Prete and M. Taïx. *A kinodynamic steering-method for legged multi-contact locomotion.* In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 3701–3707, Sept 2017. (Cited in page 10.)

[Fernbach 2018] P. Fernbach, S. Tonneau and M. Taïx. *CROC: Convex Resolution Of Centroidal dynamics trajectories to provide a feasibility criterion for the multi contact planning problem.* In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018. (Cited in page 10.)

[Forget 2017] Florent Forget, Kevin Giraud-Esclasse, Rodolphe Gelin, Nicolas Mansard and Olivier Stasse. *Differential Dynamical Programming to control a cable driven actuator for the humanoid robot Romeo.* 2017. (Cited in pages 28 and 30.)

[Fujiwara 2006] K. Fujiwara, S. Kajita, K. Harada, K. Kaneko, M. Morisawa, F. Kanehiro, S. Nakaoka and H. Hirukawa. *Towards an Optimal Falling Motion for a Humanoid Robot.* In Int. Conf. on Humanoid Robotics, 2006. (Cited in page 73.)

[Garcia 2014] Mauricio Garcia, Olivier Stasse and Jean-Bernard Hayet. *Vision-driven walking pattern generation for humanoid reactive walking.* In 2014 IEEE International Conference on Robotics and Automation (ICRA), pages 216–221. IEEE, 2014. (Cited in pages 21, 22, and 100.)

[Garrec 2010] Philippe Garrec. *Design of an anthropomorphic upper limb exoskeleton actuated by ball-screws and cables.* Bulletin of the Academy of Sciences of the Ussr-Physical Series, vol. 72, no. 2, page 23, 2010. (Cited in pages 30 and 32.)

[Grey 2016] Michael X Grey, Caelan R Garrett, C Karen Liu, Aaron D Ames and Andrea L Thomaz. *Humanoid manipulation planning using backward-forward search.* In 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 5467–5473. IEEE, 2016. (Cited in page 76.)

[Grizzle 2010] Jessy W Grizzle, Christine Chevallereau, Aaron D Ames and Ryan W Sinnet. *3D bipedal robotic walking: models, feedback control, and open problems.* IFAC Proceedings Volumes, 2010. (Cited in page 46.)

[Harada 2006] Kensuke Harada, Shuuji Kajita, Kenji Kaneko and Hirohisa Hirukawa. *An analytical method for real-time gait planning for humanoid robots.* International Journal of Humanoid Robotics, vol. 3, no. 01, pages 1–19, 2006. (Cited in page 76.)

[Herdt 2010a] A. Herdt, N. Perrin and P.-B. Wieber. *Walking without thinking about it.* In Int. Conf. on Intelligent Robots and Systems, 2010. (Cited in pages 8, 17, 18, 19, 20, 21, 48, 60, 75, 76, 79, 81, and 89.)

[Herdt 2010b] Andrei Herdt, Holger Diedam, Pierre-Brice Wieber, Dimitar Dimitrov, Katja Mombaur and Moritz Diehl. *Online walking motion generation with automatic footstep placement.* Advanced Robotics, vol. 24, no. 5-6, pages 719–737, 2010. (Cited in pages 76, 79, and 81.)

[Hereid 2016] Ayonga Hereid, Eric A Cousineau, Christian M Hubicki and Aaron D Ames. *3D dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics.* In 2016 IEEE International Conference on Robotics and Automation (ICRA), pages 1447–1454. IEEE, 2016. (Cited in pages 26 and 27.)

[Hildebrandt 2015] A. Hildebrandt, D. Wahrmann, R. Wittmann, D. Rixen and T. Buschmann. *Real-time pattern generation among obstacles for biped robots.* In IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), 2015. (Cited in page 98.)

[Iagnemma 2015] Karl Iagnemma and Jim Overholt. *Introduction.* Journal of Field Robotics, vol. 32, no. 3, pages 313–314, 2015. (Cited in page 48.)

[Imanishi 2018] K. Imanishi and T. Sugihara. *Autonomous Biped Stepping Control Based on the LIPM Potential.* In IEEE/RAS Int. Conf. on Humanoid Robotics (ICHR), 2018. (Cited in page 98.)

[Johnson 2017] Matthew Johnson, Brandon Shrewsbury, Sylvain Bertrand, Duncan Calvert, Tingfan Wu, Daniel Duran, Douglas Stephen, Nathan Mertins, John Carff, William Rifenburgh, Jesper Smith, Chris Schmidt-Wetekam, Davide Faconti, Alex Graber-Tilton, Nicolas Eyssette, Tobias Meier, Igor Kalkov, Travis Craig, Nick Payton, Stephen McCrory, Georg Wiedebach, Brooke Layton, Peter Neuhaus and Jerry Pratt. *Team IHMC's Lessons Learned from the DARPA Robotics Challenge: Finding Data in the Rubble.* Journal of Field Robotics, vol. 34, no. 2, pages 241–261, 2017. (Cited in page 44.)

[Kajita 2001] S. Kajita, K. Yokoi, M. Saigo and K. Tanie. *Balancing a Humanoid Robot Using Backdrive Concerned Torque Control and Direct Angular Momentum Feedback.* In Int. Conf. on Robotics and Automation, pages 3376–3382, 2001. (Cited in pages 62, 70, 72, and 74.)

[Kajita 2003a] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi and H. Hirukawa. *Biped walking pattern generation by using preview control of zero-moment point.* In Int. Conf. on Robotics and Automation, 2003. (Cited in pages 14, 16, 19, 36, 48, 56, 63, 66, 67, 68, 69, 70, 72, and 76.)

[Kajita 2003b] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi and H. Hirukawa. *Resolved Momentum Control: Humanoid Motion Planning based on the Linear and Angular Momentum.* In Int. Conf. on Intelligent Robots and Systems, 2003. (Cited in page 48.)

[Kajita 2007] S. Kajita, T. Nagasaki, K. Kaneko and H. Hirukawa. *ZMP-Based Biped Running Control.* IEEE Robotics Automation Magazine, vol. 14, no. 2, pages 63–72, 2007. (Cited in pages 62, 70, 72, and 74.)

[Kajita 2014] Shuuji Kajita, Hirohisa Hirukawa, Kensuke Harada and Kazuhito Yokoi. Introduction to humanoid robotics, volume 101 of *Springer Tracts in Advanced Robotics.* Springer Berlin Heidelberg, 2014. (Cited in page 108.)

[Kaneko 2005] K. Kaneko, F. Kanehiro, S. Kajita, M. Morisawa, K. Fujiwara, K. Harada and H. Hirukawa. *Slip observer for walking on a low friction floor.* In Int. Conf. on Intelligent Robots and Systems, pages 634–640, 2005. (Cited in pages 69 and 73.)

[Kaneko 2008] Kenji Kaneko, Kensuke Harada, Fumio Kanehiro, Go Miyamori and Kazuhiko Akachi. *Humanoid robot HRP-3.* In 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 2471–2478. IEEE, 2008. (Cited in page 28.)

[Kaneko 2011] Kenji Kaneko, Fumio Kanehiro, Mitsuharu Morisawa, Kazuhiko Akachi, Go Miyamori, Atsushi Hayashi and Noriyuki Kanehira. *Humanoid robot hrp-4-humanoid robotics platform with lightweight and slim body.* In 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 4400–4407. IEEE, 2011. (Cited in page 28.)

[Kaneko 2015] Kenji Kaneko, Mitsuharu Morisawa, Shuuji Kajita, Shin'ichiro Nakaoka, Takeshi Sakaguchi, Rafael Cisneros and Fumio Kanehiro. *Humanoid robot HRP-2Kai—Improvement of HRP-2 towards disaster response tasks.* In 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids), pages 132–139. IEEE, 2015. (Cited in page 26.)

[Koch 2014] Kai Henning Koch, Katja Mombaur, Olivier Stasse and Philippe Soueres. *Optimization based exploitation of the ankle elasticity of HRP-2 for overstepping large obstacles.* In Int. Conf. on Humanoid Robotics, 2014. (Cited in pages 48, 63, 70, and 72.)

[Koenemann 2015] Jonas Koenemann, Andrea Del Prete, Yuval Tassa, Emanuel Todorov, Olivier Stasse, Maren Bennewitz and Nicolas Mansard. *Whole-body Model-Predictive Control Applied to the HRP-2 Humanoid.* In Int. Conf. on Intelligent Robots and Systems, 2015. (Cited in page 48.)

[Kojima 2015] Kunio Kojima, Tatsuhi Karasawa, Toyotaka Kozuki, Eisoku Kuroiwa, Sou Yukizaki, Satoshi Iwaishi, Tatsuya Ishikawa, Ryo Koyama, Shintaro Noda, Fumihito Sugai*et al. Development of life-sized high-power humanoid robot jaxon for real-world use.* In 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids), pages 838–843. IEEE, 2015. (Cited in page 26.)

[Krause 2012] Manuel Krause, Johannes Englsberger, Pierre-Brice Wieber and Christian Ott. *Stabilization of the capture point dynamics for bipedal walking based on model predictive control.* IFAC Proceedings Volumes, vol. 45, no. 22, pages 165–171, 2012. (Cited in page 76.)

[Kuindersma 2014] Scott Kuindersma, Frank Permenter and Russ Tedrake. *An efficiently solvable quadratic program for stabilizing dynamic locomotion.* In Int. Conf. on Robotics and Automation, 2014. (Cited in page 48.)

[Kuindersma 2016] Scott Kuindersma, Robin Deits, Maurice Fallon, Andrés Valenzuela, Hongkai Dai, Frank Permenter, Twan Koolen, Pat Marion and Russ Tedrake. *Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot.* Autonomous Robots, vol. 40, no. 3, pages 429–455, 2016. (Cited in page 27.)

[Lee 1995] Jay H Lee, Yugender Chikkula, Zhenghong Yu and Jeffrey C Kantor. *Improving computational efficiency of model predictive control algorithm*

using wavelet transformation. International Journal of Control, vol. 61, no. 4, pages 859–883, 1995. (Cited in page 77.)

[Lee 2016] Jinoh Lee, Wooseok Choi, Dimitrios Kanoulas, Rajesh Subburaman, Darwin G Caldwell and Nikolaos G Tsagarakis. *An active compliant impact protection system for humanoids: Application to walk-man hands.* In 2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids), pages 778–785. IEEE, 2016. (Cited in page 30.)

[Lim 2017] Jeongsoo Lim, Inho Lee, Inwook Shim, Hyobin Jung, Hyun Min Joe, Hyoin Bae, Okkee Sim, Jaesung Oh, Taejin Jung, Seunghak Shin, Kyungdon Joo, Mingeuk Kim, Kangkyu Lee, Yunsu Bok, Dong-Geol Choi, Buyoun Cho, Sungwoo Kim, Jungwoo Heo, Inhyeok Kim, Jungho Lee, In So Kwon and Jun-Ho Oh. *Robot System of DRC-HUBO+ and Control Strategy of Team KAIST in DARPA Robotics Challenge Finals.* Journal of Field Robotics, vol. 34, no. 4, pages 802–829, 2017. (Cited in page 44.)

[Lohmeier 2009] Sebastian Lohmeier, Thomas Buschmann and Heinz Ulbrich. *Humanoid robot LOLA.* In 2009 IEEE International Conference on Robotics and Automation, pages 775–780. IEEE, 2009. (Cited in page 26.)

[Mandery 2016] C. Mandery, Ö. Terlemez, M. Do, N. Vahrenkamp and T. Asfour. *Unifying Representations and Large-Scale Whole-Body Motion Databases for Studying Human Motion.* Transactions on Robotics, 2016. (Cited in page 49.)

[Mansard 2009] N. Mansard, O. Stasse, P. Evrard and A. Kheddar. *A versatile generalized inverted kinematics implementation for collaborative working humanoid robots: The stack of tasks.* In Int. Conf. on Autonomous Robots, 2009. (Cited in pages 19, 28, 63, 65, 72, and 108.)

[Marion 2017] Pat Marion, Maurice Fallon, Robin Deits, Andrés Valenzuela, Claudia Pérez D'Arpino, Greg Izatt, Lucas Manuelli, Matt Antone, Hongkai Dai, Twan Koolen, John Carter, Scott Kuindersma and Russ Tedrake. *Director: A User Interface Designed for Robot Operation with Shared Autonomy.* Journal of Field Robotics, vol. 34, no. 2, pages 262–280, 2017. (Cited in page 44.)

[Martin 2004] Alvin F Martin, John S Garofolo, Jonathan G Fiscus, Audrey N Le, David S Pallett, Mark A Przybocki and Gregory A Sanders. *NIST Language Technology Evaluation Cookbook.* In LREC, 2004. (Cited in page 71.)

[Mastalli 2019] Carlos Mastalli, Rohan Budhiraja, Maximilien Naveau, Bilal Hammoud, Guilhem Saurel and Nicolas Mansard. The Crocoddyl C++ library – A fast and flexible optimal control library for robot control under contact sequence. 2019. (Cited in page 101.)

[Mayne 1966] David Mayne. *A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems.* International Journal of Control, vol. 3, no. 1, pages 85–95, 1966. (Cited in pages 100 and 101.)

[Mirabel 2016] Joseph Mirabel, Steve Tonneau, Pierre Fernbach, Anna-Kaarina Seppälä, Mylene Campana, Nicolas Mansard and Florent Lamiraux. *HPP: A new software for constrained motion planning.* In 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 383–389. IEEE, 2016. (Cited in page 107.)

[Missura 2016] Marcell Missura. *Analytic and Learned Footstep Control for Robust Bipedal Walking.* PhD thesis, Bonn University, 2016. (Cited in page 98.)

[Mordatch 2012] Igor Mordatch, Emanuel Todorov and Zoran Popović. *Discovery of complex behaviors through contact-invariant optimization.* ACM Transactions on Graphics (TOG), 2012. (Cited in page 48.)

[Morisawa 2007] M. Morisawa, K. Harada, S. Kajita, S. Nakaoka, K. Fujiwara, F. Kanehiro, K. Kaneko and H. Hirukawa. *Experimentation of Humanoid Walking Allowing Immediate Modification of Foot Place Based on Analytical Solution.* In Int. Conf. on Robotics and Automation, 2007. (Cited in pages 17, 22, 36, 55, 56, 60, 63, 64, 65, 66, 69, 72, and 73.)

[Morisawa 2009] Mitsuharu Morisawa, Kensuke Harada, Shuuji Kajita, Kenji Kaneko, Joan Sola, Eiichi Yoshida, Nicolas Mansard, Kazuhito Yokoi and Jean-Paul Laumond. *Reactive stepping to prevent falling for humanoids.* In 2009 9th IEEE-RAS International conference on Humanoid Robots, pages 528–534. IEEE, 2009. (Cited in page 76.)

[Muehlebach 2016] Michael Muehlebach and Raffaello D'Andrea. *Parametrized infinite-horizon model predictive control for linear time-invariant systems with input and state constraints.* In 2016 American Control Conference (ACC), pages 2669–2674. IEEE, 2016. (Cited in page 76.)

[Mukovskiy 2017] Albert Mukovskiy, Christian Vassallo, Maximilien Naveau, Olivier Stasse, Philippe Soueres and Martin A Giese. *Adaptive synthesis of dynamically feasible full-body movements for the humanoid robot HRP-2 by flexible combination of learned dynamic movement primitives.* Robotics and Autonomous Systems, vol. 91, pages 270–283, 2017. (Cited in page 49.)

[Naveau 2016] M. Naveau. *Advanced human inspired walking strategies for humanoid robots.* PhD thesis, Université de Toulouse 3 Paul Sabatier, 2016. (Cited in page 51.)

[Naveau 2017] Maximilien Naveau, Manuel Kudruss, Olivier Stasse, Christian Kirches, Katja Mombaur and Philippe Souères. *A reactive walking pattern generator based on nonlinear model predictive control.* IEEE Robotics and Automation Letters, vol. 2, no. 1, pages 10–17, 2017. (Cited in pages 11, 13, 18, 19, 22, 55, 56, 63, 67, 68, 69, 72, 73, 76, 79, 98, and 109.)

[Negrello 2016] Francesca Negrello, Manolo Garabini, Manuel G Catalano, Przemyslaw Kryczka, Wooseok Choi, Darwin G Caldwell, Antonio Bicchi and

Nikolaos G Tsagarakis. *Walk-man humanoid lower body design optimization for enhanced physical performance.* In 2016 IEEE International Conference on Robotics and Automation (ICRA), pages 1817–1824. IEEE, 2016. (Cited in pages 26, 27, and 28.)

[Nishiwaki 2012] Koichi Nishiwaki and Satoshi Kagami. *Trajectory design and control of edge-landing walking of a humanoid for higher adaptability to rough terrain.* In 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 3432–3439. IEEE, 2012. (Cited in page 76.)

[Orin 2013] David E. Orin, Ambarish Goswami and Sung-Hee Lee. *Centroidal dynamics of a humanoid robot.* Autonomous Robots, 2013. (Cited in pages 48 and 79.)

[Paine 2015] Nicholas Paine, Joshua S Mehling, James Holley, Nicolaus A Radford, Gwendolyn Johnson, Chien-Liang Fok and Luis Sentis. *Actuator Control for the NASA-JSC Valkyrie Humanoid Robot: A Decoupled Dynamics Approach for Torque Control of Series Elastic Robots.* Journal of Field Robotics, vol. 32, no. 3, pages 378–396, 2015. (Cited in page 27.)

[Perrin 2011] Nicolas Perrin. *Footstep planning for humanoid robots: discrete and continuous approaches.* PhD thesis, 2011. (Cited in page 8.)

[Perrin 2015] N. Perrin, D. Lau and V. Padois. *Effective Generation of Dynamically Balanced Locomotion with Multiple Non-coplanar Contacts.* In the International Journal of Robotics Research, 2015. (Cited in page 48.)

[Perrin 2018] Nicolas Perrin, Darwin Lau and Vincent Padois. *Effective generation of dynamically balanced locomotion with multiple non-coplanar contacts.* In Robotics Research, pages 201–216. Springer, 2018. (Cited in page 16.)

[Ponton 2018] Brahayam Ponton, Alexander Herzog, Andrea Del Prete, Stefan Schaal and Ludovic Righetti. *On time optimization of centroidal momentum dynamics.* In 2018 IEEE International Conference on Robotics and Automation (ICRA), pages 1–7. IEEE, 2018. (Cited in page 10.)

[Pratt 1995] Gill A Pratt and Matthew M Williamson. *Series elastic actuators.* In Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots, volume 1, pages 399–406. IEEE, 1995. (Cited in page 30.)

[Pro 2016] *Proxi: Redefining humanoid efficiency.* Technical Report, S. R. Institute, 2016. (Cited in page 26.)

[Qiu 2011] Zhaopeng Qiu, Adrien Escande, Alain Micaelli and Thomas Robert. *Human motions analysis and simulation based on a general criterion of stability.* In International Symposium on Digital Human Modeling, 2011. (Cited in page 48.)

[Radford 2015] N. A. Radford, P. Strawser, K. Hambuchen, J. S. Mehling, W. K. Verdeyen, A. S. Donnan, J. Holley, J. Sanchez, V. Nguyen, L. Bridgwater, R. Berka, R. Ambrose, M. Myles Markee, N. J. Fraser-Chanpong, C. McQuin, J. D. Yamokoski, S. Hart, R. Guo, A. Parsons, B. Wightman, P. Dinh, B. Ames, C. Blakely, C. Edmondson, B. Sommers, R. Rea, C. Tobler, H. Bibby, B. Howard, L. Niu, A. Lee, M. Conover, L. Truong, R. Reed, D. Chesney, R. Platt, G. Johnson, C.-L. Fok, N. Paine, L. Sentis, E. Cousineau, R. Sinnet, J. Lack, M. Powell, B. Morris, A. Ames and J. Akinyode. *Valkyrie: NASA's First Bipedal Humanoid Robot.* Journal of Field Robotics, vol. 32, no. 3, pages 397–419, 2015. (Cited in pages 27 and 44.)

[Rainbow 2017] Rainbow. *http://www.rainbow-robotics.com/.* Technical Report, January 2017. (Cited in page 26.)

[Ramirez-Alpizar 2016] I. Ramirez-Alpizar, M. Naveau, C. Benazeth, O. Stasse, J.-P. Laumond and E. Yoshida. *Motion generation for pulling a fire hose by a humanoid robot.* In IEEE/RAS Int. Conf. on Humanoid Robotics (ICHR), 2016. (Cited in page 99.)

[Romualdi 2018] Giulio Romualdi, Stefano Dafarra, Yue Hu and Daniele Pucci. *A benchmarking of DCM based architectures for position and velocity controlled walking of humanoid robots.* In 2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids), pages 1–9. IEEE, 2018. (Cited in page 76.)

[Rotella 2015] Nicholas Rotella, Alexander Herzog, Stefan Schaal and Ludovic Righetti. *Humanoid Momentum Estimation Using Sensed Contact Wrenches.* In Int. Conf. on Humanoid Robotics, 2015. (Cited in page 48.)

[Samy 2015] V. Samy and A. Kheddar. *Falls control using posture reshaping and active compliance.* In Int. Conf. on Humanoid Robotics, 2015. (Cited in page 73.)

[Semini 2011] Claudio Semini, Nikos G Tsagarakis, Emanuele Guglielmino, Michele Focchi, Ferdinando Cannella and Darwin G Caldwell. *Design of HyQ–a hydraulically and electrically actuated quadruped robot.* Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering, vol. 225, no. 6, pages 831–849, 2011. (Cited in page 30.)

[Shafiee-Ashtiani 2017] Milad Shafiee-Ashtiani, Aghil Yousefi-Koma and Masoud Shariat-Panahi. *Robust bipedal locomotion control based on model predictive control and divergent component of motion.* In 2017 IEEE International Conference on Robotics and Automation (ICRA), pages 3505–3510. IEEE, 2017. (Cited in page 76.)

[Sherikov 2014] A. Sherikov, D. Dimitrov and P.-B. Wieber. *Whole body motion controller with long-term balance constraints.* In Int. Conf. on Humanoid Robotics, 2014. (Cited in pages 27 and 48.)

[Sherikov 2016] Alexander Sherikov. *Balance preservation and task prioritization in whole body motion control of humanoid robots.* PhD thesis, INRIA, 2016. (Cited in page 48.)

[Spe 2017] *The DARPA Robotics Challenge Finals*, 2017. (Cited in page 100.)

[Spenko 2017] Matthew Spenko, Steve Buerger and Karl Iagnemma. *Editorial.* Journal of Field Robotics, vol. 34, no. 2, pages 227–228, 2017. (Cited in page 48.)

[Stasse 2008] Olivier Stasse, Adrien Escande, Nicolas Mansard, Sylvain Miossec, Paul Evrard and Abderrahmane Kheddar. *Real-time (self)-collision avoidance task on a hrp-2 humanoid robot.* In 2008 IEEE International Conference on Robotics and Automation, pages 3200–3205. IEEE, 2008. (Cited in page 11.)

[Stasse 2009] Olivier Stasse, Paul Evrard, Nicolas Perrin, Nicolas Mansard and Abderrahmane Kheddar. *Fast foot prints re-planning and motion generation during walking in physical human-humanoid interaction.* In 2009 9th IEEE-RAS International Conference on Humanoid Robots, pages 284–289. IEEE, 2009. (Cited in page 18.)

[Stasse 2013] Olivier Stasse. *Vision based motion generation for humanoid robots.* PhD thesis, 2013. (Cited in page 6.)

[Stasse 2017] Olivier Stasse, Thomas Flayols, Rohan Budhiraja, Kevin Giraud-Esclasse, Justin Carpentier, Andrea Del Prete, Philippe Souères, Nicolas Mansard, Florent Lamiraux, Jean-Paul Laumond, Luca Marchionni, Hilario Tome and Francesco Ferro. *TALOS: A new humanoid research platform targeted for industrial applications.* In IEEE/RAS Int. Conf. on Humanoid Robotics (ICHR), 2017. (Cited in pages 26, 27, 28, 37, and 73.)

[Stasse 2018] Olivier Stasse, Kevin Giraud-Esclasse, Edouard Brousse, Maximilien Naveau, Rémi Régnier, Guillaume Avrin and Philippe Souères. *Benchmarking the HRP-2 humanoid robot during locomotion.* 2018. (Cited in page 44.)

[Takasugi 2017] Noriaki Takasugi, Kunio Kojima, Shunichi Nozawa, Kei Okada and Masayuki Inaba. *3d walking and skating motion generation using divergent component of motion and gauss pseudospectral method.* In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 5003–5010. IEEE, 2017. (Cited in page 76.)

[Takumi 2017] K. Takumi, K. Hiroyuki, K. Mitsuhide, T. Chiaki, S. Shinya, T. Masanori and Y. Takahide. *Dynamic Gait Transition between Walking, Running and Hopping for Push Recovery.* In Int. Conf. on Humanoid Robotics, 2017. (Cited in page 68.)

[Tanguy 2016] A. Tanguy, P. Gergondet, A. Comport and A. Kheddar. *Closed-loop RGB-D SLAM Multi-Contact Control for humanoid robots.* In IEEE Int. Symposium on System Integrations (SII), 2016. (Cited in page 100.)

[Tassa 2014] Yuval Tassa, Nicolas Mansard and Emo Todorov. *Control-limited differential dynamic programming.* In Int. Conf. on Robotics and Automation, 2014. (Cited in page 48.)

[Tonneau 2018] S. Tonneau, A. Del Prete, J. Pettré, C. Park, D. Manocha and N. Mansard. *An Efficient Acyclic Contact Planner for Multiped Robots.* IEEE Transactions on Robotics, vol. 34, no. 3, pages 586–601, 2018. (Cited in pages 10 and 55.)

[Tonneau 2019] Steve Tonneau, Daeun Song, Pierre Fernbach, Nicolas Mansard, Michel Taïx and Andrea Del Prete. *SL1M: Sparse L1-norm Minimization for contact planning on uneven terrain.* arXiv preprint arXiv:1909.09044, 2019. (Cited in page 56.)

[Torricelli 2015] D. Torricelli, J. González-Vargas, J.-F. Veneman, K. Mombaur, N. Tsagarakis, A. J. del Ama, A. Gil-Agudo, J. C. Moreno and J. L. Pons. *Benchmarking Bipedal Locomotion: A Unified Scheme for Humanoids, Wearable Robots, and Humans.* IEEE Robotics Automation Magazine, 2015. (Cited in pages 44, 45, 48, 50, 60, and 71.)

[Toussaint 2007] Marc Toussaint, Michael Gienger and Christian Goerick. *Optimization of sequential attractor-based movement for compact behaviour generation.* In 2007 7th IEEE-RAS International Conference on Humanoid Robots, pages 122–129. IEEE, 2007. (Cited in page 11.)

[Tsagarakis 2017] N. G. Tsagarakis, D. G. Caldwell, F. Negrello, W. Choi, L. Baccelliere, V.G. Loc, J. Noorden, L. Muratore, A. Margan, A. Cardellino, L. Natale, E. Mingo Hoffman, H. Dallali, N. Kashiri, J. Malzahn, J. Lee, P. Kryczka, D. Kanoulas, M. Garabini, M. Catalano, M. Ferrati, V. Varricchio, L. Pallottino, C. Pavan, A. Bicchi, A. Settimi, A. Rocchi and A. Ajoudani. *WALK-MAN: A High-Performance Humanoid Platform for Realistic Environments.* Journal of Field Robotics, 2017. (Cited in page 44.)

[Urata 2010] Junichi Urata, Yuto Nakanishi, Kei Okada and Masayuki Inaba. *Design of high torque and high speed leg module for high power humanoid.* In 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 4497–4502. IEEE, 2010. (Cited in pages 26 and 38.)

[van Donkelaar 1999] Edwin T van Donkelaar, Okko H Bosgra and Paul MJ Van den Hof. *Model predictive control with generalized input parametrization.* In 1999 European Control Conference (ECC), pages 1693–1698. IEEE, 1999. (Cited in page 76.)

[Wahlberg 1996] Bo Wahlberg and PM Mäkilä. *On approximation of stable linear dynamical systems using Laguerre and Kautz functions.* Automatica, vol. 32, no. 5, pages 693–708, 1996. (Cited in page 77.)

[Wang 2009] Liuping Wang. Model predictive control system design and implementation using matlab®. Springer Science & Business Media, 2009. (Cited in pages 77 and 83.)

[Wensing 2017] Patrick M Wensing, Albert Wang, Sangok Seok, David Otten, Jeffrey Lang and Sangbae Kim. *Proprioceptive actuator design in the MIT Cheetah: Impact mitigation and high-bandwidth physical interaction for dynamic legged robots.* IEEE Transactions on Robotics, vol. 33, no. 3, pages 509–522, 2017. (Cited in page 29.)

[Westervelt 2007] E. R. Westervelt, J. W. Grizzle, C. Chevallereau, J. H. Choi and B. Morris. Feedback control of dynamic bipedal robot locomotion. CRC Press, 2007. (Cited in page 46.)

[Westervelt 2018] Eric R Westervelt, Jessy W Grizzle, Christine Chevallereau, Jun Ho Choi and Benjamin Morris. Feedback control of dynamic bipedal robot locomotion. CRC press, 2018. (Cited in page 16.)

[Wieber 2006] Pierre-Brice Wieber. *Trajectory free linear model predictive control for stable walking in the presence of strong perturbations.* In 2006 6th IEEE-RAS International Conference on Humanoid Robots, pages 137–142. IEEE, 2006. (Cited in page 16.)

[Wieber 2008] Pierre-Brice Wieber. *Viability and predictive control for safe locomotion.* In Int. Conf. on Humanoid Robotics, 2008. (Cited in page 47.)

[Wieber 2015] Pierre-Brice Wieber, Russ Tedrake and Scott Kuindersma. Handbook of robotics, chapter Modeling and Control of Legged Robots. 2015. (Cited in page 47.)

[Wolf 2008] Sebastian Wolf and Gerd Hirzinger. *A new variable stiffness design: Matching requirements of the next robot generation.* In 2008 IEEE International Conference on Robotics and Automation, pages 1741–1746. IEEE, 2008. (Cited in page 30.)

[Yamaguchi 2015] A. Yamaguchi and C.G. Atkeson. *Differential Dynamic Programming with Temporally Decomposed Dynamics.* In IEEE/RAS Int. Conf. on Humanoid Robotics (ICHR), 2015. (Cited in page 100.)

[Zhang 2019] Ang Zhang, Ixchel G. Ramirez-Alpizar, Kévin Giraud Esclasse, Olivier Stasse and Kensuke Harada. *Humanoid walking pattern generation based on model predictive control approximated with basis functions.* Advanced Robotics, vol. 33, no. 9, pages 454–468, 2019. (Cited in page 75.)