On Autonomous Humanoid Robots: Contact Planning for Locomotion and Manipulation

Karim Bouyarmane

Université Montpellier 2

École Doctorale Information Structures Systèmes

Thèse de Doctorat

Discipline : Systèmes Automatiques et Micro-électroniques

Présentée par

Karim Bouyarmane

De l'Autonomie des Robots Humanoïdes : Planification de Contacts pour Mouvements de Locomotion et Tâches de Manipulation

---

On Autonomous Humanoid Robots: Contact Planning for Locomotion and Manipulation

---

Soutenue le 22 novembre 2011 devant le jury composé de :

Prof. Bernard Espiau	Directeur de Recherche, INRIA	Président
Prof. Jean-Claude Latombe	Professor, Stanford University	Rapporteur
Prof. Jean-Paul Laumond	Directeur de Recherche, CNRS	Rapporteur
Prof. André Crosnier	Professeur, Université Montpellier 2	Examinateur
Prof. Hirohisa Hirukawa	Director IS, AIST Japan	Examinateur
Prof. Abderrahmane Kheddar	Directeur de Recherche, CNRS	Directeur de thèse

CNRS-AIST JRL (Joint Robotics Laboratory) UMI 3218/CRT
National Institute of Advanced Industrial Science and Technology (AIST Japan)
AIST Central 2, Umezono 1-1-1,
Tsukuba, Ibaraki 305-8568,
Japan

Abstract. We propose a unified planning approach for autonomous humanoid robots that perform dexterous locomotion and manipulation tasks. These tasks are based on contact transitions; for instance between the locomotion limbs of the robot and the environment, or between the manipulation end-effector of the robot and the manipulated object. We plan these contact transitions for general abstract systems made of arbitrary numbers of robots, manipulated objects, and environment supports. This approach allows us to erase distinction between the locomotion and manipulation nature of the tasks and to extend the method to various other planning problems such as collaborative manipulation and locomotion between multiple agents. We introduce our non-decoupled locomotion-and-manipulation planning paradigm by exhibiting the induced stratification of the configuration space of example simplified systems for which we analytically solve the problem comparing geometric path planning, kinematic non-holonomic planning, and dynamic trajectory planning methods. We then present the contact planning algorithm based on best-first search. The algorithm relies on an inverse kinematics solver that handles general robot-robot, robot-object, robot-environment, object-environment, non-horizontal, non-coplanar, friction-based, multi-contact configurations, under static equilibrium and physical limitation constraints. The continuous dynamics-consistent motion is generated in the locomotion case using a quadratic programming formulation. We finally envision the extension to deformable environment contact support by considering linear elasticity behaviours solved using the finite element method.

Keywords. Contact Planning; Motion Generation; Humanoid Robots; Autonomous Robots; Locomotion; Manipulation.

Résumé. Nous proposons une approche de planification unifiée pour robots humanoïdes réalisant des tâches de locomotion et de manipulation nécessitant une dextérité propre aux systèmes anthropomorphes. Ces tâches sont basées sur des transitions de contacts ; contacts entre les extrémités des membres locomoteurs et l'environnement dans le cas du problème de locomotion par exemple, ou entre les extrémités de l'organe préhensible effecteur et l'objet manipulé dans le cas du problème de manipulation. Nous planifions ces transitions de contacts pour des systèmes abstraits constitués d'autant de robots, d'objets, et de supports dans l'environnement que désiré/nécessaire pour la modélisation du problème. Cette approche permet de s'affranchir de la distinction de nature entre tâches de locomotion et de manipulation et s'étend à une variété d'autres problèmes tels que la coopération entre plusieurs agents. Nous introduisons notre paradigme de planification non-découplée de locomotion et de manipulation en exhibant la stratification induite dans l'espace des configurations de systèmes simplifiés pour lesquels nous résolvons analytiquement le problème en comparant des méthodes de planification géométrique, non-holonome, et dynamique. Nous présentons ensuite l'algorithme de planification de contacts basé sur une recherche *best-first*. Cet algorithme fait appel à un solveur de cinématique inverse qui prend en compte des configurations de contacts générales dans l'espace pouvant être établis entre robots, objets, et environnement dans toutes les combinaisons possibles, le tout sous contraintes d'équilibre statique et de respect des limitations mécaniques des robots. La génération de mouvement respectant l'équation de dynamique Lagrangienne est obtenue par une formulation en programme quadratique. Enfin nous envisageons une extension à des supports de contact déformables en considérant des comportements linéaires-élastiques résolus par éléments finis.

Mots-Clés. Planification de Contacts ; Génération de Mouvement ; Robot Humanoïde ; Robot Autonome ; Locomotion ; Manipulation.

Acknowledgements

I would like to thank my thesis advisor Abderrahmane Kheddar for providing me with the directions while giving me the total liberty to carry out this work. I would like also to thank Kazuhito Yokoi and Eiichi Yoshida for hosting me at AIST.

It has been a great honor for me to have Jean-Claude Latombe and Jean-Paul Laumond review this dissertation, and to have the thesis examined by André Crosnier, Bernard Espiau, and Hirohisa Hirukawa. Their comments, feedback, and future work suggestions, were all very precious in completing this work.

Many thanks to Emmanuel Dhooge, Joris Vaillant, Benjamin Chrétien, for their contributions to various aspects of this thesis and for their implementation efforts. I am indebted to François Keith, as well as to Antoine Bussy, for their time setting up the C++ development environment under the different computers I worked on. I would also like to thank Paul Evrard and all the AMELIF modeling and simulation framework team, and Thomas Moulard and the Roboptim non-linear optmization framework team, for being reactive to my interrogations and for their support using these frameworks. Thanks to Pierre-Brice Wieber for the advice on the optimization aspect of this work.

During this thesis I enjoyed the visits and the opportunities to present this work and discuss some of its aspects with many humanoid research teams. In particular thanks to Katsu Yamane and Jessica Hodgins for inviting me at Disney Research Pittsburgh. Thanks to Yoshihiko Nakamura for the presentation of the research activities at the University of Tokyo, Thanks to Christian Ott for the visit of the DLR in Munich, and to Mitsuo Kawato and Jun Morimoto for their invitation at ATR in Kyoto.

Finally, many thanks to my family for their support.

Contents

Acknowledgements i

Introduction 1

Chapter 1. Fundamentals for Non-Decoupled Locomotion and Manipulation Planning for Low-Dimensional Systems 5
1.1. Introduction 5
1.2. Systems 7
1.3. Geometric Motion Planning Approach 12
1.4. Kinematic Control-Theoretic Approach 21
1.5. Dynamic Trajectory Planning Approach 31
1.6. Conclusion 37

Chapter 2. Multi-Contact Stances Planning for Multiple Agents 39
2.1. Introduction 39
2.2. Preliminary Notations and Definitions 40
2.3. Algorithm 46
2.4. Results 50
2.5. Conclusion 52
2.6. Appendix: Additional Examples 54

Chapter 3. Static Multi-Contact Inverse Problem for Multiple Humanoid Robots and Manipulated Objects 59
3.1. Introduction 59
3.2. Related Work 60
3.3. Problem Formulation 61
3.4. Gradients Derivations 66
3.5. Results 69
3.6. Conclusion 69

Chapter 4. Using a Multi-Objective Controller to Synthesize Simulated Humanoid Robot Motion with Changing Contact Configurations 73
4.1. Introduction 73
4.2. Related Work and Contribution 74
4.3. Overview of the Method 75
4.4. Multi-Objective Controller 75
4.5. Finite-State Machine 79
4.6. Playback Simulation Results 80
4.7. Conclusion 83
4.8. Appendix: Dynamics Equation 83

Chapter 5. FEM-based Static Posture Planning for a Humanoid Robot on Deformable Contact Support 87
5.1. Introduction 87
5.2. The Finite Element Method 88
5.3. Formulation of the Planning Problem 91
5.4. Simulation Results 96
5.5. Conclusion 98

Conclusion 101

Appendix A. Potential Field Guide for Humanoid Multi-Contact Acyclic Motion Planning 103
A.1. Introduction 103
A.2. Solution 103
A.3. Results 111
A.4. Conclusion 112

Bibliography 113

Introduction

A humanoid robot can be defined as an anthropomorphic controlled mechanical system. See Fig. 0.1.

The anthropomorphic design is an intuitive choice that stems from a desire of the human being to re-create an avatar of themselves. We can call it the artificial human quest. This innate desire is sufficient to make us perform research on the subject. But this choice has also been tried to be rationally justified by a number of arguments, necessary for the research endeavour to survive within the economic structure of the society.

First, building a humanoid robot allows us to study human motion by trying to reproduce it. Reproducing the motion is a demonstration that we have understood it. Another possible justification is that one of the objectives of building robots is to make them replace humans in performing tasks that are too hazardous or fastidious for them. In that case the humanoid design is the most suitable to move around and operate in an environment that was originally designed and optimized to withstand human operators/operations. One last argument is that humanoid robots are aimed at evolving and interacting together with other real humans, who are assumed not to necessarily have neither the scientific background to understand technical limitations nor the precise knowledge of robot mechanical and cognitive capabilities. Thus making a human interact with a human-like mechanical avatar greatly simplifies this latter task.

However-though, the humanoid-design constraint of a mechanical system makes it particularly complex to control. For the purposes listed above, direct low-level joint control by human operator is inadequate given the complexity of the tasks at stake. Tele-operation, playback of pre-recorded motion, imitation, would all obviously be failing options in that context. This is why autonomy is the most desirable control strategy to achieve. However, full autonomy is a tremendously ambitious goal to attain just as well. Many layers are involved and the complexity of the resulting control architectures grows exponentially relative to the targeted level of autonomy. Reducing this complexity and making simple, thus elegant, intelligence emulators is a challenging, but not impossible,

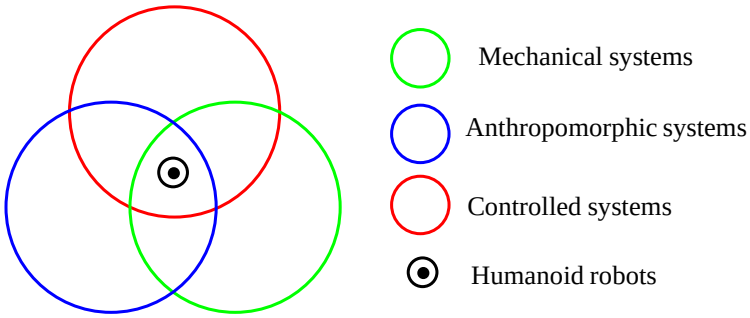


FIGURE 0.1. Classification of systems

task. One idea towards this is to build autonomy around some unifying concept and concentrate the development efforts around this concept.

This brings us to the thesis that we try to establish in the present dissertation. Multi-contact planning can be one of these unifying autonomy-enhancing concepts, contributing to providing humanoids with some level of autonomy to achieve tasks in a human-like manner (for the sake of interactivity) within a human-mechanical-capability optimized operation environment (for the sake of replacement). To abide by the above-stated simplicity requirement, we taxonomically identify two main tasks: locomotion and manipulation, both of which are made of successive contact making and breaking in order to get from an initial configuration to a goal. Generality is ensured by not restricting ourselves to cyclic sequences, to "gaits", and by using all possible contact spots and configurations. Achieving autonomy can thus be broken-down to planning these sequences of contacts for locomotion and manipulation in the simplest, thus the most-unified possible, way. This is our thesis, and our contribution.

Fig 0.2 shows a quick overview of the organization of the dissertation. Chapter 1 lays down the fundamentals behind unifying locomotion and manipulation that have systematically been treated separately in the planning and control literature. The problem is analytically solved for low-dimensional planar systems. Chapter 2 then tackles the problem for the full-scale humanoid robots presenting the algorithm, but most importantly the conceptual modelling framework, for multi-contact planning applied to general systems and situations. One essential component of this algorithm, a particular inverse kinematics solver, deserved more detailed development, that can be found in Chapter 3. At that point of the dissertation, a sequence of static postures is planned. In Chapter 4 we propose an approach to generate the full motion in-between these static postures. Finally Chapter 5 goes one step farther towards generality by considering deformable environment when planning a static posture, as an extended version of the problem solved in Chapter 3. We then conclude the dissertation.

Chapters 2 to 5 have been published in the proceedings of scientific conferences related to humanoids. See Table 0.1 for the corresponding references.

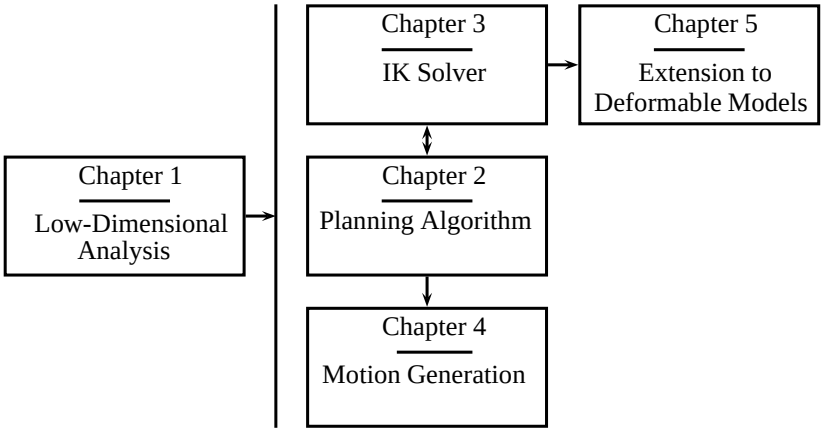


FIGURE 0.2. Structure of the dissertation

Chapter 2	[**BK11b**]
Chapter 3	[**BK10**]
Chapter 4	[**BK11c**]
Chapter 5	[**BK11a**]
Appendix A	[**BELK09**]

TABLE 0.1. References of the publications of the chapters

CHAPTER 1

Fundamentals for Non-Decoupled Locomotion and Manipulation Planning for Low-Dimensional Systems

This opening chapter demonstrates the possibility of solving planning problems interleaving locomotion and manipulation in a non-decoupled way. We choose three low-dimensional minimalistic robotic systems and use them to illustrate our paradigm: a basic one-legged locomotor, a two-link manipulator with manipulated object, and a simultaneous locomotion-and-manipulation system. The approach followed in the chapter is 1) to depart from existing motion planning and control methods initially designed for either locomotion or manipulation tasks, 2) see how they apply to both our locomotion-only and manipulation-only systems through parallel derivations, and finally 3) extend these methods to the simultaneous locomotion-and-manipulation system. Motion planning is solved for these three systems using two different methods, a geometric path-planning-based one and a kinematic control-theoretic-based one. Motion control is then derived for dynamically realising the geometric paths or kinematic trajectories under Couloumb friction hypothesis using torques as control inputs. All three methods apply successfully to all three systems, showing that the non-decoupled planning is possible.

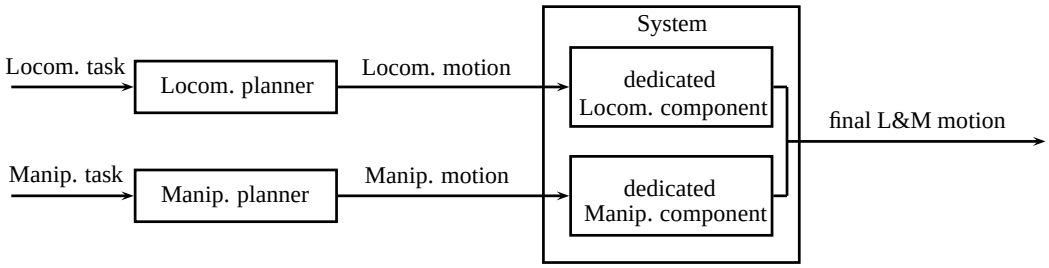
1.1. Introduction

Robots are traditionally categorized into fixed-base manipulators [**MLS94, Cra04**] and mobile navigation robots (wheeled [**Lau98**] or legged [**KE08**]) (e.g. [**SSV08**]). Many of them, however, do not fall strictly into one of these two categories as they feature both locomotion and manipulation capabilities and are designed for performing indifferently both kinds of tasks, falling thus into a third locomotion-and-manipulation category. Humanoid robots [**KFH$^+$08**], which constitute the initial motivation that inspired this work, are typical examples of such locomotion-and-manipulation integrated systems.

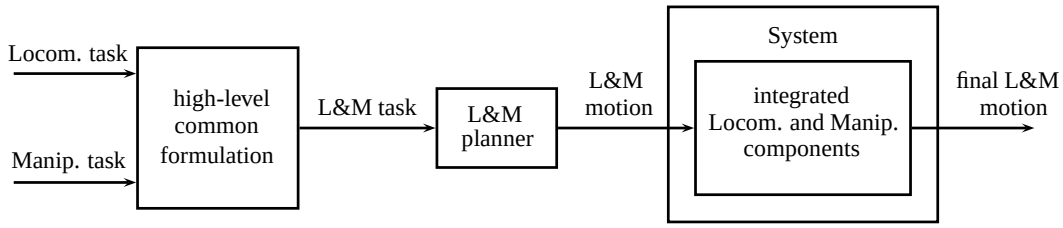
It is well known that, from a motion planning and control point of view, locomotion and manipulation are conceptually the same problems. Their commonality comes from their inherent under-actuation that is solved through the contact forces: a locomotion system is under-actuated in the sense that the position of the mobile base is not controlled directly through actuators torques, but rather results from both the actuation torques action and the contact forces with the support environment; a manipulation system (by *manipulation system* we mean both the manipulator and the manipulated object) is also under-actuated in a strictly equivalent way: the degrees of freedom of the manipulated object are not actuated and its position is an indirect result of the actuation of the manipulator through the contact forces that it establishes with the manipulated object. Besides, they both obey Lagrangian dynamics, they both involve friction, and they both have contact strata of various dimensions.

Though being equivalent, these two problems have usually been tackled in a decoupled way for integrated manipulation-and-locomotion systems ([**KL10, KLY11**] can be considered as an exception given that the planned motion display locomotion and manipulation components that had not been pre-specified, but they choose to model the under-actuation as the redundancy of a virtual manipulator while we tackle it directly as the core problem). A decoupled approach might be pertinent for classes of systems in which the initial design imposes totally unrelated locomotion and manipulation components, consider for example a particular wheeled mobile robot equipped with a manipulator arm, in which the wheeled base is entirely dedicated to the navigation task and the manipulator arm is solely devoted to the manipulation task. In that case the decoupled strategy is arguably the most adequate one. However, for systems such as humanoid robots, the frontier between the two kinds of tasks is more blurred, and it is restrictive to exclusively assign upper-body limbs to manipulation and lower-body limbs to locomotion. For instance, a humanoid robot might be required to use its arms to climb a ladder or to crawl under a table, it might also need to use its legs to push an object on the floor while walking. In such situations, decoupled approaches using an upper-body joint-space or task-space controller for manipulation and an independent lower-body walking subsystem controller for locomotion [**HNTHGB07, YLE+08**] can be restrictive and not use the full potential of the human-inspired design. Of course this is not a property of humanoid robots exclusively, and, as an example among others, [**BTGZ08, BT08**] nicely demonstrate non-decoupled locomotion and manipulation capabilities of a tracked mobile robot equipped with a manipulator arm.

Our driving objective is to erase high-level distinction between manipulation and locomotion objectives, both in terms of specification of the task and of the planning method to plan the motion to realise the task. In the resulting motion, interleaved manipulation and locomotion should emerge with no prior high-level distinctive formulation. See Fig. 1.1.



(a) The existing decoupled approach.



(b) The proposed non-decoupled approach. The L&M abbreviation stands for Locomotion-and-Manipulation

FIGURE 1.1. Overview of the approaches.

The methodology chosen is to capture the locomotion and manipulation problems into the lowest-dimensional possible systems. The systems are only theoretical planar "toy" systems but we believe they are still pertinent enough to illustrate our point. As such, this chapter is primarily focused on this theoretic and conceptual level and the extension to the above-mentioned humanoid problems is beyond its scope. The three systems we chose are representative of the three categories of robots we mentioned earlier:

- one exclusively locomotion-oriented system,
- one exclusively manipulation-oriented system,
- one hybrid locomotion-and-manipulation (L&M) system.

We then investigate two main existing motion planning methods from the literature applicable to our systems:

- a geometric path planning approach based on a *reduction property* proved initially in [**ALS95**] and used in a randomized planning algorithm in [**SLCS04**].
- a control-theoretic BVP (Boundary Value Problem) approach for kinematics systems based on a *controllability theorem* proved in [**GB01**] and a BVP resolution algorithm developed in [**GB02**].

The first approach deals directly with the obstacle avoidance problem. The second is more adequate for dealing with the velocity constraints and nonholonomy which may not translate directly into geometric terms. To make our study complete and self-contained we also tackle the dynamic trajectory generation problem along the geometric paths resulting from these motion planning algorithms, using the works of [**SEM05**] and [**BDG85**] as a basis. We derive time-optimal open-loop torques control law that realizes a given contact motion.

Taking each one of these three motion planning and control techniques, we first apply it the the locomotion system, then we show formal equivalence with the manipulation system, before finally extending it to the locomotion-and-manipulation system, which is the main contribution of this work.

Following this methodology, the rest of the chapter is structured as follows: Section 1.2 introduces the three robots we will study with their configuration spaces, Section 1.3 applies the geometric path planning approach to our motion planning problem, Section 1.4 uses control theory for solving the motion planning problem seen as a BVP, finally Section 1.5 synthesizes time-optimal control law that realizes the geometric paths provided in previous sections. Each of these sections is divided into three subsections: one for the locomotion robot, one for the manipulation robot, and one for the locomotion-and-manipulation robot.

1.2. Systems

Throughout this chapter we will thoroughly study three low-dimensional planar mechanical systems:

- R_1: a locomotion robot
- R_2: a manipulation robot
- R_3: a locomotion-and-manipulation (L & M) robot

We have chosen these robots for they have the lowest-dimensional possible configuration spaces but yet can capture higher dimensional locomotion and manipulation related concepts. This low dimensionality allows visualizing the configuration spaces in 3D at the

expense of simple projections and homeomorphisms. The other purpose of these low-dimensional planar systems is to have explicit analytical expressions for our problems and their solutions.

For all these systems $\mathcal{C}$ denotes the configuration space, also known as the "C-space". A configuration is denoted $q \in \mathcal{C}$, q is the generalized coordinates vector of the system [**MLS94**]. An important mathematical property in our study is the fact that $\mathcal{C}$ is a smooth manifold. This makes it suited for being described inside the framework of differential geometry theory [**Ish99**]. Velocities $\dot{q}$ are as such elements of the tangent spaces and generalized forces are elements of the cotangent spaces.

We can classify all the possible forms that the C-space can take for systems commonly considered in robotics. A free-flyer yields the manifold $SE(3) = SO(3) \rtimes \mathbb{R}^3$ (semi-direct product). Let $\mathbb{S}^n$ be the n-dimensional sphere. A revolute joint yields the manifold $\mathbb{S}^1$, a spherical joint yields $\mathbb{S}^3$. Let $\mathbb{T}^n = (\mathbb{S}^1)^n$ be the n-dimensional torus. A prismatic joint yields the manifold $\mathbb{R}$. In most robotics systems the configuration space $\mathcal{C}$ is a Cartesian product of a given number of these elementary smooth manifolds, thus it is a smooth manifold.

Let O be the obstacle region in the Euclidean workspace. O is a compact subset of $\mathbb{R}^2$. Let $\mathcal{C}_{\text{obs}}$ be the image of O in the configuration space, consisting of all configurations where the robot collides with O. $\mathcal{C}_{\text{obs}}$ is a compact subset of $\mathcal{C}$ [**Lat91**]. Let $\mathcal{C}_{\text{free}}$ be the subspace of $\mathcal{C}$ consisting of all configurations that are not in collision with obstacles, within the joint limits, and not in self-collision. $\mathcal{C}_{\text{free}}$ is an open subset of $\mathcal{C}$ [**Lat91**]. Studies such as [**AB88**] are concerned with the computation of explicit representation of the frontier of $\mathcal{C}_{\text{obs}}$ in particular cases, for instance polygonal robots and obstacles in planar world.

We detail now the models and notations for each of the three robotic systems.

1.2.1. Locomotion robot. The robot R_1 is made of a sliding base along the x-axis and a two-link planar manipulator linked with two revolute joints, see Fig. 1.2. Two actuators control the two revolute joints; the sliding joint is passive, *i.e.* not actuated and frictionless. The sliding of the base along the x-axis can be performed by using friction of end-effector's rubber on the ground.

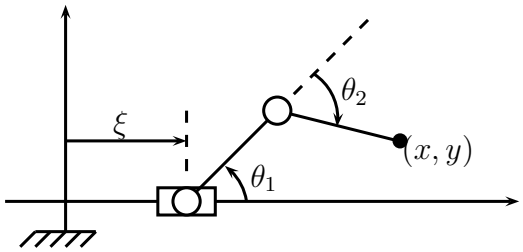


FIGURE 1.2. R_1 and its configuration variables. Rectangles symbolize a prismatic joint while circles represent a revolute joint.

The configuration space of the system is

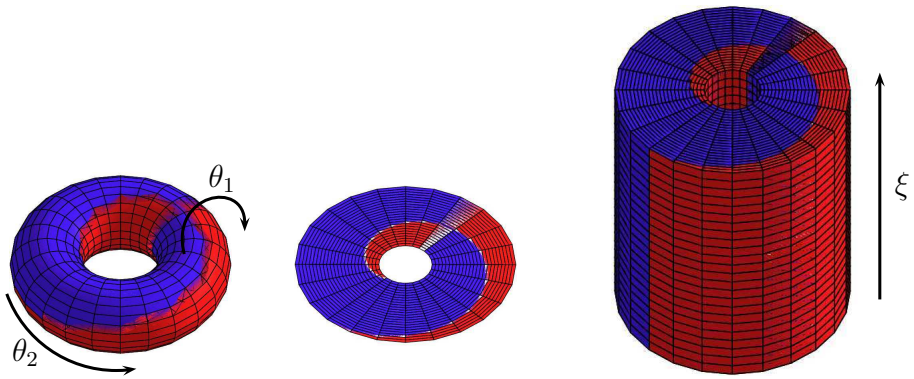
$$\mathcal{C} = \mathbb{R} \times \mathbb{T}^2 \tag{1.1}$$

On the manifold $\mathcal{C}$ we use the following coordinates chart $(\xi, \theta_1, \theta_2) \in \mathbb{R}^3$. We denote by $(x(q), y(q))$ the end effector coordinates in the (x, y)-plane. We do not take into account

self-collision of the robot. There are no joints limits. l denotes the length of the two links. The robot is not allowed to traverse the ground, thus:

$$\mathcal{C}_{\text{free}} = \mathcal{C} \setminus \mathcal{C}_{\text{obs}} = \{q \in \mathcal{C} \mid \theta_1 > 0 \text{ and } 2\,\theta_1 + \theta_2 > 0\} \tag{1.2}$$

$\mathbb{T}^2$, the 2-dimensional torus, is naturally embedded in $\mathbb{R}^3$ which means that $\mathcal{C}$ is embedded in $\mathbb{R}^4$, thus $\mathcal{C}_{\text{free}}$ is also embedded in $\mathbb{R}^4$. However a projection trick will make it embedded in $\mathbb{R}^3$. We simply notice that the projection of $\mathcal{C}_{\text{free}}$ onto $\mathbb{T}^2$ is a 2D manifold which is homeomorphic to a subspace of $(0, \pi) \times \mathbb{S}^1$ and thus $\mathcal{C}_{\text{free}}$ is homeomorphic to a subspace of $\mathbb{R} \times (0, \pi) \times \mathbb{S}^1$ which is naturally embedded in $\mathbb{R}^3$, see Fig. 1.3.



(a) $\mathbb{T}^2$ embedded in $\mathbb{R}^3$

(b) Free part of $\mathbb{T}^2$ embedded in $\mathbb{R}^2$

(c) Adding the ξ dimension

FIGURE 1.3. Embedding the C-space in $\mathbb{R}^3$. The blue part represents $\mathcal{C}_{\text{free}}$, the red part is $\mathcal{C} \setminus \mathcal{C}_{\text{free}}$.

The 3D representation of the C-space of R_1 allows for an explicit representation of $\mathcal{C}_{\text{obs}}$ for any obstacle O for which the frontier is a parametrized 2D curve $\partial O : s \mapsto (x_O(s), y_O(s)))$.

First let us consider a point obstacle O located at the (x_O, y_O) coordinates. The configurations q that make the robot in collision with O can be computed by giving the inverse kinematics solution for the end-effector of a copy robot of R_1, but with the second link having a parameter length λ. Then we make λ vary in $[0, l]$, and we get all the configurations q that make the second link of the robot collide with O. We use the same method by removing the second link, vary the length of the first link, and compute inverse kinematics for this robot, which gives us the second component of $\mathcal{C}_{\text{obs}}$. See Fig. 1.4.

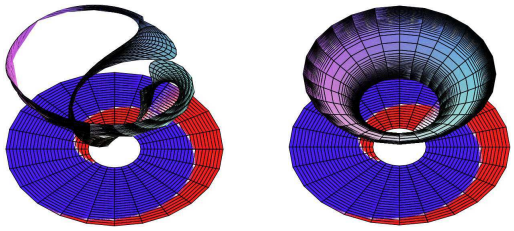
Now for the full obstacle $\partial O : s \mapsto (x_O(s), y_O(s)))$ we apply the method we have just described by varying the parameter s.

1.2.2. Manipulation robot. The robot R_2 is a standard two-link planar manipulator fixed to the ground, manipulating a sliding object. See Fig. 1.5. The manipulated object is pictured in red; it consists of a theoretically infinitely long sliding platform. The manipulator has to put its rubber end-effector on the platform and use friction force to push or pull the object.

The configuration space of R_2 is the same as R_1

$$\mathcal{C} = \mathbb{R} \times \mathbb{T}^2 \tag{1.3}$$

However we use a different notation for the coordinates chart $(\alpha, \theta_1, \theta_2)$ where α denotes the horizontal position of any reference point on the red sliding base. Similarly to R_1,



(a) $\mathcal{C}_{\text{obs},1}$, corresponding to the configurations that bring the second link into collision with the point.

(b) $\mathcal{C}_{\text{obs},2}$, corresponding to the configurations that bring the first link into collision with the point.

FIGURE 1.4. Components of $\mathcal{C}_{\text{obs}}$ for a point obstacle for the robot R_1.

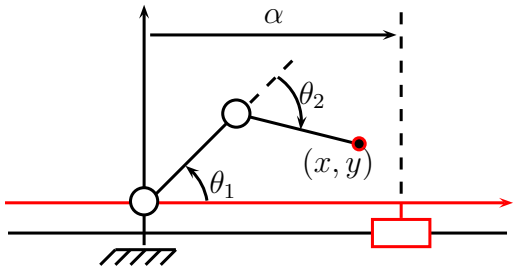


FIGURE 1.5. R_2 and its configuration variables. Joints symbols are the same as Fig. 1.2. In red the infinitely long sliding platform.

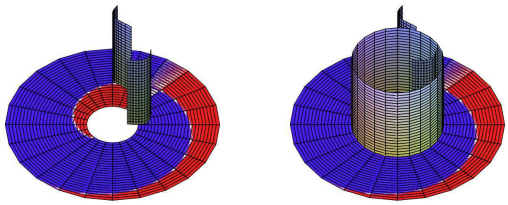
we consider no self-collision, no joint limits, and Fig. 1.3 provides a 3D visualization of R_2's C-space (in the caption read "adding the α dimension" instead of "adding the ξ dimension"). The only difference with R_1 is the representation of the obstacle region in the C-space, which is basically the $\mathcal{C}_{\text{obs}}$ of a standard two-link manipulator, as detailed in the following paragraph.

To get a parametric representation of $\mathcal{C}_{\text{obs}}$ we use the same trick that we introduced in the computation of R_1's $\mathcal{C}_{\text{obs}}$. For a point obstacle (x_O, y_O) we compute the inverse kinematics solution of a robot similar to R_2 but varying the length of the second link as a parameter $\lambda \in [0, l]$, then we extrude in the α dimension (given that the obstacle region does not depend on the position of the sliding base), we thus get a first component of $\mathcal{C}_{\text{obs}}$ as a 2D submanifold of $\mathcal{C}$. The second component comes simply from removing the second link and computing the trivial inverse kinematics of a one-link robot, which reduces to a constant θ_1. See Fig. 1.6.

For an obstacle given by a parametrization of its contour $s \mapsto (x_O(s), y_O(s))$, we directly add s as a third parameter of our manifold, and we get the representation depicted in Fig. 1.7 for a circular obstacle for example.

1.2.3. L & M robot. The robot R_3 combines R_1 and R_2. It is made of sliding two-link planar manipulator manipulating a infinitely long sliding platform. See Fig. 1.8. Its configuration space is

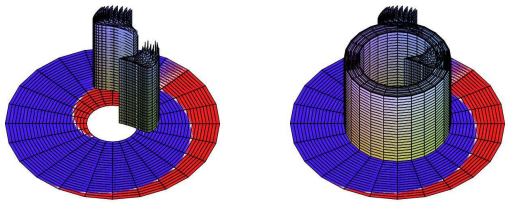
$$\mathcal{C} = \mathbb{R}^2 \times \mathbb{T}^2 \tag{1.4}$$



(a) $\mathcal{C}_{\text{obs},1}$, corresponding to the configurations that bring the second link into collision with the point.

(b) $\mathcal{C}_{\text{obs},2}$, corresponding to the configurations that bring the first link into collision with the point.

FIGURE 1.6. Components of $\mathcal{C}_{\text{obs}}$ for a point obstacle for the system R_2.



(a) $\mathcal{C}_{\text{obs},1}$, corresponding to the configurations that bring the second link into collision with the circle.

(b) $\mathcal{C}_{\text{obs},2}$, corresponding to the configurations that bring the first link into collision with the circle.

FIGURE 1.7. Components of $\mathcal{C}_{\text{obs}}$ for a circular obstacle for the system R_2.

It is a four-dimensional smooth manifold that cannot this time be embedded in $\mathbb{R}^3$. We skip the representation of the C-space and its obstacle region but we will come back to this issue later (section 1.3) as we will restrain to a special 3D submanifold of the C-space.

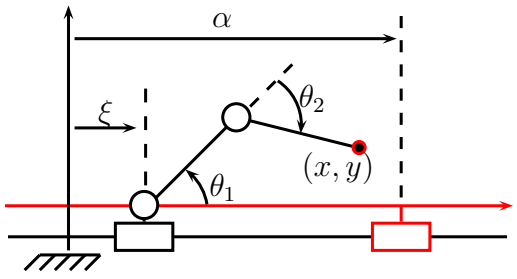


FIGURE 1.8. R_3 and its configuration variables. Rectangles symbolize prismatic joints and circle represent revolute joints. The infinitely long sliding platform is pictured in red.

1.3. Geometric Motion Planning Approach

The systems introduced in the previous section are underactuated systems. We can geometrically visualize this underactuation as a foliated stratification structure in the C-space.

1.3.1. Locomotion robot. First let us consider the robot R_1. Its configuration space $\mathbb{R} \times \mathbb{T}^2$ is stratified into two different strata, see Fig. 1.9. The first stratum $\mathcal{S}_0$ (zero contact) corresponds to the situation in which the end-effector is not in contact with the ground. It is a submanifold of $\mathcal{C}$ made of all the corresponding configurations

$$\mathcal{S}_0 = \{(\xi, \theta_1, \theta_2) \in \mathbb{R} \times (0, \pi) \times [-\pi, \pi] \mid \theta_2 > -2\,\theta_1\} \tag{1.5}$$

The second stratum $\mathcal{S}_1$ (one contact) is the submanifold corresponding to all configurations that bring the end-effector in contact with the ground. It is a 2-dimensional submanifold of $\mathcal{C}$

$$\mathcal{S}_1 = \{(\xi, \theta_1, \theta_2) \in \mathbb{R} \times (0, \pi) \times [-\pi, \pi] \mid \theta_2 = -2\,\theta_1\} \tag{1.6}$$

On this submanifold we use the coordinates chart (ξ, θ_1)

$$\mathcal{S}_1 : \begin{cases} \xi &= \xi \\ \theta_1 &= \theta_1 \\ \theta_2 &= -2\,\theta_1 \end{cases} \tag{1.7}$$

Each of these two strata is foliated into a continuum of leafs. A leaf is a submanifold of the stratum in which the robot is fully actuated. A single leaf of $\mathcal{S}_0$ corresponds to a fixed position of the base ξ, meaning $\xi =$ constant. We call this foliation the ξ-foliation, and for a given $\xi \in \mathbb{R}$ we denote the corresponding leaf $\mathcal{Q}_{0,\xi}$

$$\mathcal{Q}_{0,\xi} = \{(\xi, \theta_1, \theta_2) \mid (\theta_1, \theta_2) \in (0, \pi) \times [-\pi, \pi] \text{ and } \theta_2 > -2\,\theta_1\} \tag{1.8}$$

On $\mathcal{Q}_{0,\xi}$, R_1 can move its two links freely in their workspace but does not slide. A single leaf of $\mathcal{S}_1$ corresponds to fixed position x of the end-effector on the ground, *i.e.* $x =$ constant. We call this foliation the x-foliation. For a given $x \in \mathbb{R}$ we denote the corresponding leaf $\mathcal{Q}_{1,x}$

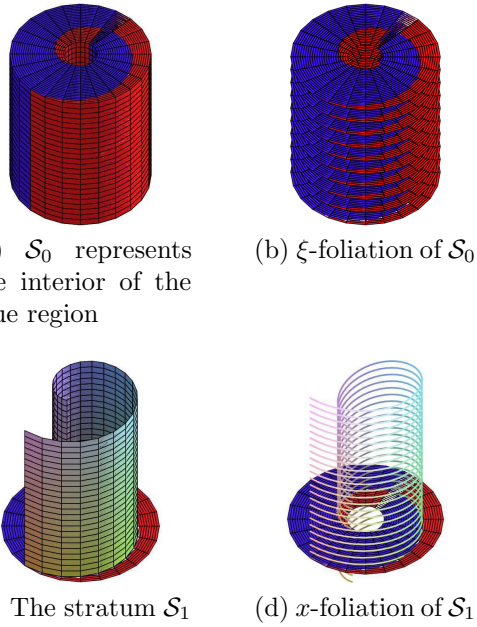
$$\mathcal{Q}_{1,x} = \{(\xi, \theta_1, \theta_2) \in \mathbb{R} \times (0, \pi) \times [-\pi, \pi] \mid \theta_2 = -2\,\theta_1 \text{ and } \xi + 2\,l\cos(\theta_1) = x\} \tag{1.9}$$

or, using the parameter θ_1 as coordinate chart,

$$\mathcal{Q}_{1,x} : \begin{cases} \xi &= x - 2\,l\cos(\theta_1) \\ \theta_1 &= \theta_1 \\ \theta_2 &= -2\,\theta_1 \end{cases} \tag{1.10}$$

On such a leaf the robot takes fixed support on the ground and the applied torques result in the sliding of the base.

The purpose of geometric motion planning is to plan a continuous path in the C-space from an initial point to a destination point avoiding the $\mathcal{C}_{\text{obs}}$ region. However in our foliated structure the actuators can only make the robot move smoothly along an isolated leaf of the C-space, so the only valid paths should be made of a finite succession of elementary paths along single leafs. This makes the classical techniques of exploring the C-space [**Lat91, CLH$^+$05, LaV06**] not directly applicable to our motion planning



(a) $\mathcal{S}_0$ represents the interior of the blue region

(b) ξ-foliation of $\mathcal{S}_0$

(c) The stratum $\mathcal{S}_1$

(d) x-foliation of $\mathcal{S}_1$

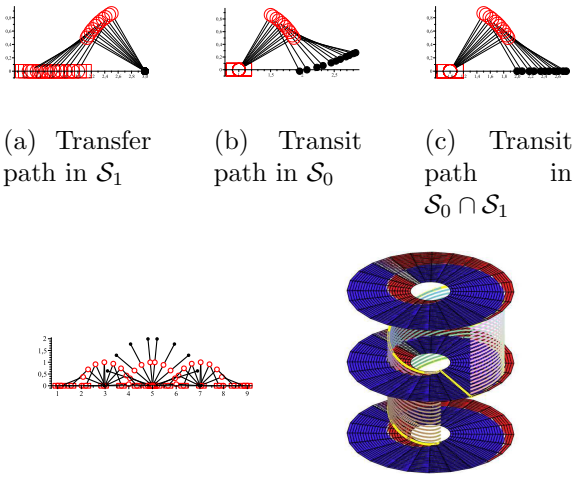
FIGURE 1.9. The strata $\mathcal{S}_0$, $\mathcal{S}_1$ and their foliations for the robot R_1.

problem. However, authors in [**SLCS04**] provide a way to overcome this foliation structure and reduce the problem to a classical motion planning problem in a non-foliated C-space.

In [**SLCS04**], a *manipulation path* through the C-space is defined as a sequence of *transit paths* and *transfer paths*. A transit path is a path in which the object lies at rest on the ground not being manipulated while the manipulator moves freely in its workspace, a transfer path is a path in which the manipulator is grasping the object at a fixed grasp location and the object is "stuck" to the manipulator end-effector. These two kinds of paths are paths along two different strata of the configuration space, respectively the *object-stable* stratum and the *object-grasped* stratum. The uncountable infinite stable positions of the object resting on the ground define a foliation of the object-stable stratum, and the uncountable infinite positions of grasps of the end-effector on the object define a foliation on the object-grasped stratum. As shown above, our robot R_1 fits directly inside this problem formulation. Following the manipulation planning terminology, we will call a path through a leaf of $\mathcal{S}_0$ a *transit path* and a path through a leaf of $\mathcal{S}_1$ a *transfer path*. See Fig. 1.10.

The planning approach developed in [**SLCS04**] is the following: uncover the different connected components of $\mathcal{S}_0 \cap \mathcal{S}_1$ as if there was no foliation structure[1] (this is done by building a roadmap and connecting the nodes with linear edges thus violating the foliation structure), then try to connect these different components using only transit or transfer paths. In a post-processing step, The *reduction property* allows to transform any collision-free path of $\mathcal{S}_0 \cap \mathcal{S}_1$ into a finite sequence of transfer and transit paths. This reduction property has first been proved in [**ALS95**]. The following works (e.g. [**SLCS04, SSEKP07**]) based on this property usually assume that the extension of the property

[1]In the remaining of this chapter, we will denote by $\mathcal{S}_0 \cap \mathcal{S}_1$ the stratum $\mathcal{S}_1$ endowed with both $\mathcal{S}_0$ and $\mathcal{S}_1$ foliation ($\mathcal{S}_1$ is seen as a subset of the topological closure of $\mathcal{S}_0$).



(a) Transfer path in $\mathcal{S}_1$

(b) Transit path in $\mathcal{S}_0$

(c) Transit path in $\mathcal{S}_0 \cap \mathcal{S}_1$

(d) Valid path in the operational space

(e) Valid path in the C-space (in yellow)

FIGURE 1.10. Types of paths for the robot R_1.

is straightforward in their particular problem. However, we believe that the property takes a very specific form in each particular problem and thus needs to be proven on a case-by-case basis, inspired by the general principles of the initial proof. We follow this approach in this section. Moreover, only a constructive proof is candidate to be used as an actual motion planning algorithm. For similar reduction-property-based planning approaches, see [**HW86**].

Fig. 1.11 and Fig. 1.12 represent the foliation structure on $\mathcal{S}_0 \cap \mathcal{S}_1$. The representation of the obstacle region in Fig. 1.11 uses the technique presented in section 1.2. Fig. 1.12 illustrates the application of the reduction property in a simple case.

PROBLEM 1.3.1. *Given* $(q_{\text{initial}}, q_{\text{final}}) \in \mathcal{C}^2_{\text{free}}$ *find* $N \in \mathbb{N}$, *a sequence* $(k_i)_{i=1..N} \in \{0,1\}^N$, *a sequence* $(\zeta_i)_{i=1..N} \in \mathbb{R}^N$, *and a sequence of continuous paths* $p_i : [0,1] \to \mathcal{Q}_{k_i,\zeta_i} \cap \mathcal{C}_{\text{free}}$, *such that* $p_0(0) = q_{\text{initial}}$, $p_N(1) = q_{\text{final}}$, *and* $\forall i \in \{0,\dots,N-1\}$ $p_i(1) = p_{i+1}(0)$.

PROPOSITION 1.3.2. *If there exists for* R_1 *a collision-free path in unfoliated* $\mathcal{S}_0 \cap \mathcal{S}_1$ *from* q_{initial} *to* q_{final} *then there exists a finite sequence of transfer and transit paths that links* q_{initial} *and* q_{final}.

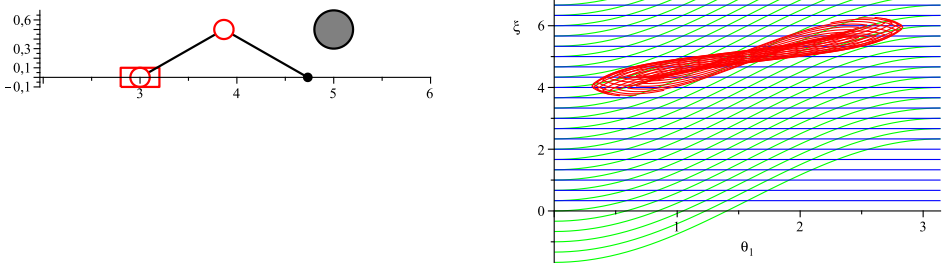
PROOF. The two foliations of $\mathcal{S}_0 \cap \mathcal{S}_1$ can be respectively represented by the two families of functions:

$$\begin{aligned} f_\alpha : (0,\pi) &\to \mathbb{R} \\ \theta_1 &\mapsto \xi = f_\alpha(\theta_1) = \text{cste} = \alpha \quad , \alpha \in \mathbb{R} \end{aligned} \tag{1.11}$$

which represents the horizontal foliation (the ξ-foliation), and

$$\begin{aligned} g_\beta : (0,\pi) &\to \mathbb{R} \\ \theta_1 &\mapsto \xi = g_\beta(\theta_1) = -2\cos(\theta_1) + \beta \quad , \beta \in \mathbb{R} \end{aligned} \tag{1.12}$$

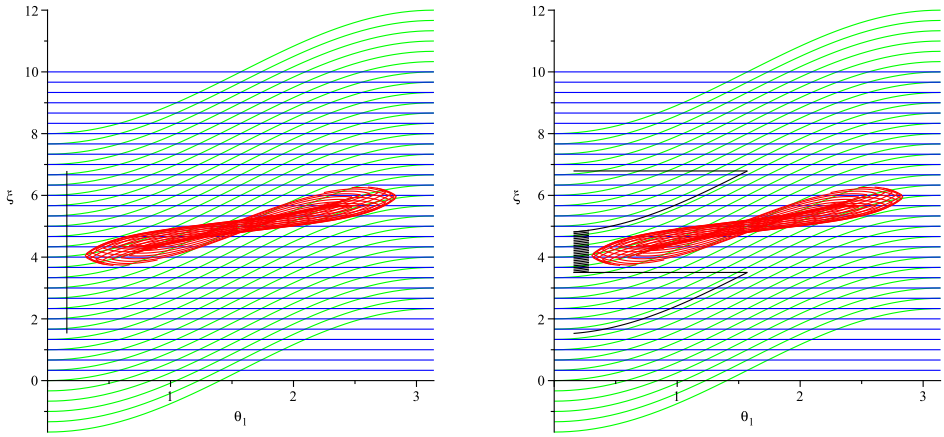
which represents the curved inclined foliation (the x-foliation).



(a) A circular obstacle in the operational space.

(b) The obstacle region in the foliated $\mathcal{S}_0 \cap \mathcal{S}_1$ in represented in red. The blue foliation is the ξ-foliation, the green foliation is the x-foliation.

FIGURE 1.11. Example of an obstacle and its mapping in the foliated spaces.



(a) Original path (black vertical path on the left)

(b) Valid path

FIGURE 1.12. Illustration of the reduction property. In the first figure the black vertical linear path in the left of the figure violates the foliation. In the second figure the path is deformed in order to comply with the foliation.

For more convenience in the notations we replace the (θ_1, ξ) coordinate chart notation on $\mathcal{S}_0 \cap \mathcal{S}_1$ by the more usual plane coordinates (x, y). We also denote $\mathcal{C} = (0, \pi) \times \mathbb{R}$ our ambient metric space, and the obstacle region $\mathcal{O}$ which is a non-empty compact (*ie.* closed and bounded) subset of $\mathcal{C}$. The complementary set of $\mathcal{O}$ that we denote $\mathcal{O}^c = \mathcal{C} \setminus \mathcal{O}$ is an open subset of $\mathcal{C}$. The distance between two subsets A and B of $\mathcal{C}$ is defined as:

$$d(A, B) = \inf_{a \in A, b \in B} d(a, b) \tag{1.13}$$

The two foliations on $\mathcal{C}$ are now represented by the two families of functions: $f_\alpha(x) = \alpha,\ \alpha \in \mathbb{R}$ and $g_\beta(x) = g(x) + \beta,\ \beta \in \mathbb{R}$ where $g : (0, \pi) \to \mathbb{R}$ is a continuous strictly increasing function.

In our demonstration we first consider the case of an initial vertical path. Let $p_v : [0, 1] \to \mathcal{O}^c$ be a normal parametrization of our vertical path (arc-length parametrization) from the bottom extremity, *ie.* $p_v(t) = (x_0, y_0 + t \cdot l)$ where l is the length of the path. Let $\mathrm{Tr}(p_v) = \{p_v(t) \mid t \in [0, 1]\}$. Since $\mathrm{Tr}(p_v)$ and $\mathcal{O}$ are two non-empty compact subsets of $\mathcal{C}$, their distance is finite $d(\mathrm{Tr}(p_v), \mathcal{O}) < +\infty$. Since they are closed sets with empty intersection $\mathrm{Tr}(p_v) \cap \mathcal{O} = \varnothing$ their distance is strictly positive $d(\mathrm{Tr}(p_v), \mathcal{O}) > 0$. Let $\varepsilon = \frac{d(\mathrm{Tr}(p_v), \mathcal{O})}{2}$.

We will now give a recursive construction of a finite sequence of collision-free transit paths and transfer paths that links (x_0, y_0) to $(x_0, y_0 + l)$.

From the foliation definition we know that $\exists! \beta_0 \in \mathbb{R},\ g_{\beta_0}(x_0) = y_0$. Let B_0 be the closed ball of center (x_0, y_0) and of radius ε. From the construction of ε we have $B_0 \subset \mathcal{O}^c$. Let $y = a(x)$ be the equation of the closed upper right quarter circle boundary of B_0. We have $a(x_0) = g_{\beta_0}(x_0) + \varepsilon > g_{\beta_0}(x_0)$ and $g_{\beta_0}(x_0 + \varepsilon) > g_{\beta_0}(x_0) = a(x_0 + \varepsilon)$. The intermediate value theorem applied to the continuous strictly increasing function $g_{\beta_0} - a$ (a being continuous strictly decreasing function) gives us a unique point (x'_0, y_1) of intersection between the graphs of g_{β_0} and a such that $(x'_0, y_1) \in (x_0, x_0 + \varepsilon) \times (y_0, y_0 + \varepsilon)$. B_0 being strictly convex, the horizontal line segment between the points (x'_0, y_1) and (x_0, y_1) is inside B_0. Let $\alpha_0 = y_1$. Finally we have constructed a sequence of two paths

$$\begin{aligned} \mathrm{Transfer}_0 : [x_0, x'_0] &\to \mathcal{O}^c \\ x &\mapsto (x, g_{\beta_0}(x)) \end{aligned} \tag{1.14}$$

and

$$\begin{aligned} \mathrm{Transit}_0 : [-x'_0, -x_0] &\to \mathcal{O}^c \\ x &\mapsto (-x, f_{\alpha_0}(-x)) \end{aligned} \tag{1.15}$$

that link (x_0, y_0) to (x_0, y_1). Let $d = y_1 - y_0$. $d > 0$ from the above definition of y_1. let $N = \lfloor \frac{l}{d} \rfloor$. Repeating the previous procedure from the point (x_0, y_1) we define recursively a sequence of points along $\mathrm{Tr(p_v)}$, $(x_0, y_n)_{0 \le n \le N}$ where $y_n = y_0 + nd$ and the corresponding sequences of paths $(\mathrm{Transfer}_n, \mathrm{Transit}_n)_{0 \le n \le N-1}$ that link (x_0, y_n) to (x_0, y_{n+1}). To end the recursion, Let $y_{N+1} = y_0 + l$ and $x'_N = g^{-1}_{\beta_N}(y_{N+1})$. The last transit and transfer paths of the sequence are defined as:

$$\begin{aligned} \mathrm{Transfer_N} : [x_N, x'_N] &\to \mathcal{O}^c \\ x &\mapsto (x, g_{\beta_N}(x)) \end{aligned} \tag{1.16}$$

and

$$\begin{aligned} \mathrm{Transit_N} : [-x'_N, -x_N] &\to \mathcal{O}^c \\ x &\mapsto (-x, f_{\alpha_N}(-x)) \end{aligned} \tag{1.17}$$

Finally, the sequence $(\mathrm{Transfer}_n, \mathrm{Transit}_n)_{0 \le n \le N}$ link the initial and final point of our vertical path p_v, which ends the first part of the demonstration.

Let us now consider a given non-necessarily vertical path from (x_0, y_0) to (x_f, y_f), $p : [0, 1] \to \mathcal{O}^c$. We suppose that p is a normal (arc-length) parametrization, otherwise we can re-parametrize under the condition that p is regular, meaning that $\forall t \in [0, 1],\ \dot{p}(t) \neq (0, 0)$. Let l be the length of the path.

We will first show that we can find a finite sequence of collision-free vertical and horizontal paths that link (x_0, y_0) to (x_f, y_f). Once again we define $\varepsilon = \frac{d(\mathrm{Tr}(p),\mathcal{O})}{2}$. Let $N = \min\left\{n \in \mathbb{N} \mid \frac{l}{n} < \varepsilon\right\}$. We define the sequence of points along $\mathrm{Tr}(p)$, $(x_n, y_n)_{0\leq n\leq N}$ such that $(x_n, y_n) = p(\frac{n}{N})$, for $0 \leq n \leq N$. Now for each $0 \leq n \leq N-1$ We define the following sequence of horizontal and vertical paths:

$$\begin{aligned} \mathrm{Horizontal_n} : [x_n, x_{n+1}] &\to \mathcal{O}^c \\ x &\mapsto (x, y_n) \end{aligned} \tag{1.18}$$

and

$$\begin{aligned} \mathrm{Vertical_n} : [y_n, y_{n+1}] &\to \mathcal{O}^c \\ y &\mapsto (x_{n+1}, y) \end{aligned} \tag{1.19}$$

(the notations of the intervals above depends on the relative ordering of x_n and x_{n+1}, and of y_n and y_{n+1}). $[(x_n, y_n), (x_{n+1}, y_{n+1})]$ is the hypotenuse of the triangle (x_n, y_n), (x_{n+1}, y_n), (x_{n+1}, y_{n+1}) so the length of the two paths above are less than the length of the chord $[(x_n, y_n), (x_{n+1}, y_{n+1})]$ which is less than the arc-length from (x_n, y_n) to (x_{n+1}, y_{n+1}) which is by construction equal to $\frac{l}{N} < \varepsilon$. This means that the two sequences of paths $\mathrm{Horizontal}_n$ and $\mathrm{Vertical}_n$ are effectively included in $\mathcal{O}^c$, *ie.* are collision-free.

All in all we constructed a finite sequence of collision-free vertical and horizontal paths from q_{initial} to q_{final}. Each horizontal path is already a transit path. Each vertical path can be decomposed using the first part of this demonstration in a finite sequence of transfer and transit path. This means that we constructed a finite sequence of transfer and transit paths that link q_{initial} and q_{final}.

□

1.3.2. Manipulation robot. All the development provided in the previous section for R_1 is strictly valid for R_2 *modulo* some slight changes of referential and notations. The system being a manipulation system, the terminology in [**SLCS04**] applies now directly to R_2.

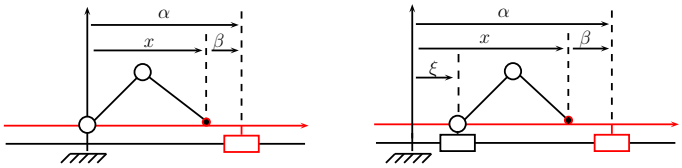
To adapt the development of the previous section from R_1 to R_2 we first need to replace all the occurrences of ξ by α, so for example we call α-foliation instead of ξ-foliation for $\mathcal{S}_0$. For a fixed $\alpha \in \mathbb{R}$, a leaf $\mathcal{Q}_{0,\alpha}$ of this foliation corresponds to a fixed location of the sliding platform while the manipulator moves freely in its workspace.

For the stratum $\mathcal{S}_1$ the foliation should correspond to the different possible locations of the contact point which will be fixed in the inertial frame of the sliding platform. So we introduce a new variable $\beta = \alpha - x$ (see Fig. 1.13a) which becomes the new co-parameter of $\mathcal{S}_1$ foliation, that we will call the β-foliation (instead of the x-foliation for R_1). See Fig. 1.14. For $\beta \in \mathbb{R}$, a leaf $\mathcal{Q}_{1,\beta}$ is written as

$$\mathcal{Q}_{1,\beta} = \{(\alpha, \theta_1, \theta_2) \in \mathbb{R} \times (0, \pi) \times [-\pi, \pi] \mid \theta_2 = -2\,\theta_1 \text{ and } 2\,l\cos(\theta_1) + \beta = \alpha\} \tag{1.20}$$

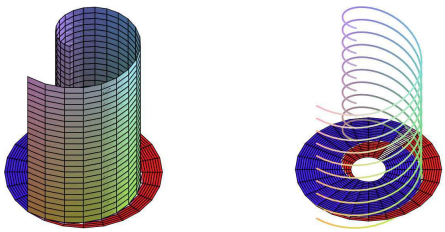
or, using the parameter θ_1 as coordinate chart,

$$\mathcal{Q}_{1,\beta} : \begin{cases} \alpha &= \beta + 2\,l\cos(\theta_1) \\ \theta_1 &= \theta_1 \\ \theta_2 &= -2\,\theta_1 \end{cases} \tag{1.21}$$



(a) R_2 (b) R_3

FIGURE 1.13. The β variable.



(a) The stratum $\mathcal{S}_1$ of R_2 (b) β-foliation of $\mathcal{S}_1$

FIGURE 1.14. The stratum $\mathcal{S}_1$ and its foliation for the system R_2.

PROPOSITION 1.3.3. *If there exists for R_2 a collision-free path in unfoliated $\mathcal{S}_0 \cap \mathcal{S}_1$ from q_{initial} to q_{final} then there exists a finite sequence of transfer and transit paths that links q_{initial} and q_{final}.*

PROOF. For R_2, the two foliations of $\mathcal{S}_0 \cap \mathcal{S}_1$ can be respectively represented by the two families of functions:

$$\begin{aligned} f_\mu : (0,\pi) &\to \mathbb{R} \\ \theta_1 &\mapsto \alpha = f_\mu(\theta_1) = \text{constant} = \mu, \quad \mu \in \mathbb{R} \end{aligned} \tag{1.22}$$

which represents the horizontal foliation (the α-foliation), and

$$\begin{aligned} g_\nu : (0,\pi) &\to \mathbb{R} \\ \theta_1 &\mapsto \alpha = g_\nu(\theta_1) = 2\cos(\theta_1) + \nu, \quad \nu \in \mathbb{R} \end{aligned} \tag{1.23}$$

which represents the curved inclined foliation (the β-foliation).

The argument used in the proof of proposition 1.3.2 was that the function g is a strictly increasing function which allowed us to apply the intermediate value theorem. Actually, we only need strict monotony to reach the same conclusion. In our present case the corresponding function g is strictly decreasing. So the proof of proposition 1.3.2 is valid for proposition 1.3.3.

□

1.3.3. L & M robot. We now consider the robot R_3. Similarly to R_2 we define the variable $\beta = \alpha - x$ as pictured in Fig. 1.13b.

The configuration space of the robot is 4-dimensional $\mathbb{R}^2 \times \mathbb{T}^2$ parametrized by $(\xi, \theta_1, \theta_2, \alpha)$. We still have only two actuators at the revolute joints, therefore the degree of underactuation is $4 - 2 = 2$. However, we also still have only one possible contact force to resolve the underactuation and reduce its degree by one. One possible way of resolving the last remaining degree of underactuation is to add a discrete switching control variable

$u_d \in \{0, 1\}$ which will allow us to either block the manipulator's base and release the sliding platform (case $u_d = 0$) or release the manipulator's base and block the sliding platform (case $u_d = 1$).

Using the terminology of hybrid control theory, we will consider the following discrete "states" of the robot:

- *The free mode.* The manipulator's base and the sliding platform are fixed, *i.e.* $\xi =$ constant and $\alpha =$ constant. This defines a first state in which the manipulator's links (θ_1, θ_2) move freely in their workspace.
- *The manipulation mode.* The manipulator's base is fixed and the end-effector is in contact with the sliding platform at fixed position in the platform's frame, *i.e.* $\xi =$ constant and $\beta =$ constant. This defines a second state in which the manipulator pushes or pulls the platform.
- *The locomotion mode.* The sliding platform is fixed and the end-effector is in contact with the sliding platform at fixed position in the platform's frame, *i.e.* $\alpha =$ constant and $\beta =$ constant. This defines a last state in which the manipulator pushes or pulls itself.

.

We still have two strata: $\mathcal{S}_0 = \mathcal{C}$ and $\mathcal{S}_1 : \theta_2 = -2\theta_1$. However, $\mathcal{S}_1$ is now a three dimensional submanifold on which we use the coordinate chart (ξ, α, θ_1). The two states locomotion and manipulation are both defined in the stratum $\mathcal{S}_1$ and represent two cross foliations of the same stratum at the same time.

We thus get three foliations, one on $\mathcal{S}_0$ and two on $\mathcal{S}_1$, that we can visualize in $\mathcal{S}_0 \cap \mathcal{S}_1$ as represented in Fig. 1.15:

- On $\mathcal{S}_0$ we define the (α, ξ)-foliation and the leafs $\mathcal{Q}_{0,\alpha,\xi}$ in green on Fig. 1.15. A path along one of these leaves will be called a free path.
- On $\mathcal{S}_1$ we define the (β, ξ)-foliation and the leafs $\mathcal{Q}_{1,\beta,\xi}$ in blue on Fig. 1.15. A path along one of these leaves will be called a manipulation path.
- On $\mathcal{S}_1$ we define the (α, β)-foliation and the leafs $\mathcal{Q}_{2,\alpha,\beta}$ in red on Fig. 1.15. A path along one of these leaves will be called a locomotion path.

PROPOSITION 1.3.4. *If there exists for R_3 a collision-free path in unfoliated $\mathcal{S}_0 \cap \mathcal{S}_1$ from q_{initial} to q_{final} then there exists a collision-free finite sequence of free, manipulation, and locomotion paths that links q_{initial} and q_{final}.*

PROOF. Let us consider the 3D Cartesian space $\mathbb{R}^2 \times (0, \pi)$ provided with the system of coordinates (α, ξ, θ_1) in which we consider a compact subset O and the families of functions

$$
\begin{array}{rcl}
f_{\alpha,\beta} : (0,\pi) & \to & \mathbb{R}^3 \\
\theta_1 & \mapsto & \left\{ \begin{array}{ll} \alpha & \\ \beta - 2\cos(\theta_1) & \alpha, \beta \in \mathbb{R} \\ \theta_1 & \end{array} \right.
\end{array}
\tag{1.24}
$$

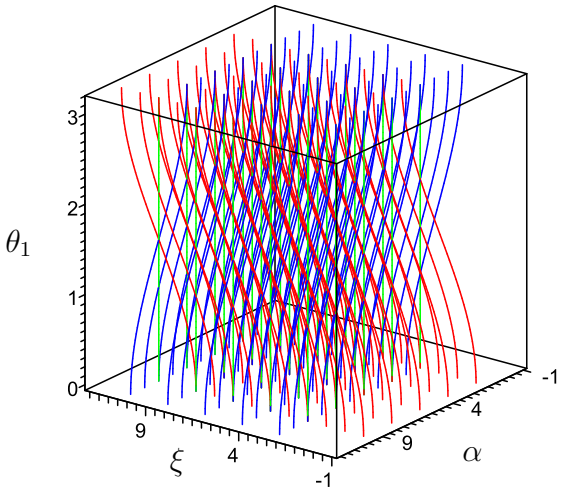


FIGURE 1.15. The foliations of the three strata for the system R_3. In blue the (β, ξ)-foliation, in red the (α, β)-foliation, in green the (α, ξ)-foliation.

which represents the red foliation,

$$\begin{aligned} g_{\beta,\xi} : (0,\pi) &\to \mathbb{R}^3 \\ \theta_1 &\mapsto \begin{cases} \beta + 2\cos(\theta_1) \\ \xi \qquad\qquad\qquad \beta, \xi \in \mathbb{R} \\ \theta_1 \end{cases} \end{aligned} \tag{1.25}$$

which represents the blue foliation,

$$\begin{aligned} h_{\alpha,\xi} : (0,\pi) &\to \mathbb{R}^3 \\ \theta_1 &\mapsto \begin{cases} \alpha \\ \xi \qquad \alpha, \xi \in \mathbb{R} \\ \theta_1 \end{cases} \end{aligned} \tag{1.26}$$

which represents the green foliation.

First we will prove that any collision-free path parallel to the α axis can be decomposed in a finite sequence of collision-free paths along the foliations. Let that α-parallel path be defined by $\theta_1 = \theta_{1_0}$ and $\xi = \xi_0$. The foliations $(g_{\beta,\xi_0})_\beta$ and $(h_{\alpha,\xi_0})_\alpha$ represent two foliations in the affine plan $\xi = \xi_0$, one strictly decreasing and one constant, for which we can directly apply proposition 1.3.3. Thus in that affine plan $\xi = \xi_0$ we can decompose the α-parallel path into a finite sequence of blue and green paths.

Similarly we prove that any collision-free path parallel to the ξ axis can be decomposed in a finite sequence of collision-free paths along the foliations. Let that ξ-parallel path be defined by $\theta_1 = \theta_{1_0}$ and $\alpha = \alpha_0$. The foliations $(f_{\alpha_0,\beta})_\beta$ and $(h_{\alpha_0,\xi})_\xi$ represent two foliations in the affine plan $\alpha = \alpha_0$, one strictly increasing and one constant, for which we can directly apply proposition 1.3.2. Thus in that affine plan $\alpha = \alpha_0$ we can decompose the ξ-parallel path into a finite sequence of red and green paths.

Any collision-free path parallel to the θ_1 axis is already a green path in the foliation.

Now extending the same method that we used in the proof of proposition 1.3.2 we can prove that any collision-free path in $\mathcal{S}_0 \cap \mathcal{S}_1$ can be decomposed in a finite sequence of collision-free paths parallel to the axes α, ξ and θ_1. □

One important remark has to be made at this point. The motion that we get by this planning is a succession of isolated locomotion and manipulation motions, with either $\alpha =$ constant or $\xi =$ constant. However, we can plan a motion in which both α and ξ are varying simultaneously, which would be equivalent to a locomotion-while-manipulating conceptual motion. This can be done simply by replacing one of the two foliations on $\mathcal{S}_1$ with a new foliation, let us call it the (λ_1, λ_2)-foliation, $\lambda_1 + \lambda_2 = 1$, for which we write a condition $\lambda_1\alpha + \lambda_2\xi =$ constant replacing one of the conditions $\alpha =$ constant or $\xi =$ constant. The (λ_1, λ_2)-foliation replacing one of the previous two on $\mathcal{S}_1$ makes it still possible to explore all the foliated space using the reduction property. Moreover, adding the (λ_1, λ_2)-foliation to the set of the previous three adds redundancy in the system and gives multiple solutions for the motion planning problem. Thus it is also possible to synthesize a locomotion-while-manipulating motion.

Let us call a path through the (λ_1, λ_2)-foliation a locomotion-while-manipulation path. The previous remark translates into the following corollary

COROLLARY 1.3.5. *If there exists for R_3 a collision-free path in unfoliated $\mathcal{S}_0 \cap \mathcal{S}_1$ from q_{initial} to q_{final} then there exists*

- *a collision-free finite sequence of free, manipulation, and locomotion-while-manipulation paths that links q_{initial} and q_{final}.*
- *a collision-free finite sequence of free, locomotion, and locomotion-while-manipulation paths that links q_{initial} and q_{final}.*
- *a collision-free finite sequence of free, locomotion, manipulation, and locomotion-while-manipulation paths that links q_{initial} and q_{final}.*

1.4. Kinematic Control-Theoretic Approach

In the previous section we have seen the underactuation of the robots as foliations in the C-space along which we need to cruise in order to reach our goal. In this section we will rather see this underactuation as a non-spanning distribution of control vector fields, our robots being considered as driftless stratified kinematic control systems. We strongly advice the reader to get to the two main references [**GB01**] and [**GB02**] since all what follows builds on their result. The references [**LS93, Lau98, BL00**] can also prove useful for the reader unfamiliar with nonholonomic motion planning.

1.4.1. Locomotion robot. First let us consider the robot R_1.

The aim here is to generate a *trajectory* (as opposed to *path* produced in the previous approach) using nonholonomic control techniques but without explicitly taking into account the obstacles. However, the philosophy remains the same: planning a sequence of transfer and transit trajectories in $\mathcal{S}_0 \cap \mathcal{S}_1$.

For this we first need to model R_1 as a kinematic control system. Our kinematic control inputs are $u_1 = \dot{\theta}_1$ and $u_2 = \dot{\theta}_2$. No control input controls directly ξ.

The system is stratified in the sense defined in [**GB01**]. If we denote by $\Phi \in \mathscr{C}^\infty(\mathcal{C})$ the function that maps every configuration $q \in \mathcal{C}$ to the height of the end effector $h = \Phi(q) = y(q)$, then we can redefine $\mathcal{S}_0 = \mathcal{C}$ as the top stratum and $\mathcal{S}_1 = \Phi^{-1}(\{0\})$ as the bottom stratum. We have the trivial inclusion chain $\mathcal{S}_1 \subset \mathcal{S}_0$.

Two different equations of motion are acting on the two strata:

- On $\mathcal{S}_0$, the base is fixed and we can write

$$\frac{d}{dt}\begin{pmatrix}\xi\\ \theta_1\\ \theta_2\end{pmatrix} = \begin{pmatrix}0\\ 1\\ 0\end{pmatrix} u_1 + \begin{pmatrix}0\\ 0\\ 1\end{pmatrix} u_2 \tag{1.27}$$

- On $\mathcal{S}_1$, the end effector is fixed as we consider a non-sliding contact, and thus the equation of motion is written

$$\frac{d}{dt}\begin{pmatrix}\xi\\ \theta_1\\ \theta_2\end{pmatrix} = \begin{pmatrix}2l\sin(\theta_1)\\ 1\\ -2\end{pmatrix} u_1 \tag{1.28}$$

We can rewrite those two equations using the formalism of driftless control theory. Let $x = (\xi, \theta_1, \theta_2)^T$ denote the state of our kinematic system (Note: for the remaining of this section x denotes the state of the system as usual in control theory and not the x-coordinate of the end-effector). Let $g_{0,1}(x) = \frac{\partial}{\partial\theta_1}$, $g_{0,2}(x) = \frac{\partial}{\partial\theta_2}$ be the two control fields acting on $\mathcal{S}_0$ and $g_{1,1}(x) = 2l\sin(\theta_1)\frac{\partial}{\partial\xi} + \frac{\partial}{\partial\theta_1} - 2\frac{\partial}{\partial\theta_2}$. Then our stratified driftless system is modelled by the two equations:

$$\begin{aligned}\dot{x} &= g_{0,1}(x)u_1 + g_{0,2}(x)u_2 \quad , \ x \in \mathcal{S}_0\\ \dot{x} &= g_{1,1}(x)u_1 \qquad\qquad\quad , \ x \in \mathcal{S}_1\end{aligned} \tag{1.29}$$

Let us study the controllability of our system.

PROPOSITION 1.4.1. *The underactuated kinematic control system R_1 is small time locally controllable in* $\mathrm{int}(\mathcal{C}_{\mathrm{free}})$

PROOF. We consider $x_0 \in \mathcal{S}_1$ an element from the bottom stratum. Let

$$\begin{aligned}\Delta_{\mathcal{S}_0}|_{x_0} &= \mathrm{span}\{g_{0,1}(x_0), g_{0,2}(x_0)\}\\ \Delta_{\mathcal{S}_1}|_{x_0} &= \mathrm{span}\{g_{1,1}(x_0)\}\end{aligned} \tag{1.30}$$

be the distributions associated with the control fields of each stratum and $\bar{\Delta}_{\mathcal{S}_0}|_{x_0}$ and $\bar{\Delta}_{\mathcal{S}_1}|_{x_0}$ be their involutive closure under Lie Bracketting. Since $[g_{0,1}, g_{0,2}] = 0$ we have

$$\begin{aligned}\bar{\Delta}_{\mathcal{S}_0}|_{x_0} &= \mathrm{span}\{g_{0,1}(x_0), g_{0,2}(x_0)\}\\ \bar{\Delta}_{\mathcal{S}_1}|_{x_0} &= \mathrm{span}\{g_{1,1}(x_0)\}\end{aligned} \tag{1.31}$$

we clearly have, for each $x_0 \in \mathcal{S}_1$ such that $\theta_1 \neq k\pi$

$$\bar{\Delta}_{\mathcal{S}_0}|_{x_0} + \bar{\Delta}_{\mathcal{S}_1}|_{x_0} = \mathrm{span}\{g_{0,1}(x_0), g_{0,2}(x_0), g_{1,1}(x_0)\} = T_{x_0}\mathcal{C} \tag{1.32}$$

and thus following the controllability theorem of [**GB01**] the system is small time locally controllable from x_0. □

Now let us address the issue of gait controllability. We consider the cyclic gait

$$\mathcal{G} = (\mathcal{S}_1, \mathcal{S}_0, \mathcal{S}_1) \tag{1.33}$$

in which the robot alternatively lifts its end-effector off the ground and then put it back on the ground.

PROPOSITION 1.4.2. *R_1 is gait-controllable with the gait $\mathcal{G}$.*

PROOF. We construct the gait distribution as follows:

$$\begin{aligned} \mathcal{D}_1 &= \bar{\Delta}_{\mathcal{S}_1}|_{x_0} \\ \mathcal{D}_2 &= \mathcal{D}_1 + \bar{\Delta}_{\mathcal{S}_0}|_{x_0} = T_{x_0}\mathcal{C} \\ \mathcal{D}_3 &= (\mathcal{D}_2 \cap T_{x_0}\mathcal{S}_1) + \bar{\Delta}_{\mathcal{S}_1}|_{x_0} \end{aligned} \tag{1.34}$$

we can parametrize $\mathcal{S}_1$ by the equations

$$\mathcal{S}_1 : \begin{cases} \xi &= \xi \\ \theta_1 &= \theta_1 \\ \theta_2 &= -2\theta_1 \end{cases} \tag{1.35}$$

which allows us to write

$$T_{x_0}\mathcal{S}_1 = \text{span}\left\{\frac{\partial}{\partial\xi}, \frac{\partial}{\partial\theta_1} - 2\frac{\partial}{\partial\theta_2}\right\} \tag{1.36}$$

we can see that $g_{1,1}(x_0) \in T_{x_0}\mathcal{S}_1$ and thus $\mathcal{D}_3 = T_{x_0}\mathcal{S}_1$ meaning that $\dim(\mathcal{D}_3) = \dim(T_{x_0}\mathcal{S}_1)$, which proves, following [**GB01**]'s result, the gait controllability of $\mathcal{G}$. □

We want now to plan a motion from an initial state $q_{\text{initial}} = (\xi_i, \theta_{1_i}, -2\theta_{1_i})^T \in \mathcal{S}_1$ to a goal state $q_{\text{final}} = (\xi_f, \theta_{1_f}, -2\theta_{1_f})^T \in \mathcal{S}_1$. To do so we first construct a stratified extended system on $\mathcal{S}_1$ by constructing a vector field from $\Delta_{\mathcal{S}_0}$ that is tangent to $\mathcal{S}_1$. The vector field we consider here is $g_{1,2} = g_{0,1} - 2g_{0,2} = \frac{\partial}{\partial\theta_1} - 2\frac{\partial}{\partial\theta_2}$, so that our system becomes, on the bottom stratum $\mathcal{S}_1$:

$$\dot{x} = g_{1,1}(x)u_1 + g_{1,2}(x)u_2 \tag{1.37}$$

then we extend the system by adding a vector field from the Lie Algebra of the two control fields we now have on $\mathcal{S}_1$ to better condition the system. We get the following stratified system on $\mathcal{S}_1$:

$$\dot{x} = b_1 v_1 + b_2 v_2 + b_3 v_3 \tag{1.38}$$

where

$$\begin{aligned} b_1 &= g_{1,1} \\ b_2 &= g_{1,2} \in \Delta_{\mathcal{S}_0} \cap T\mathcal{S}_1 \\ b_3 &= [b_1, b_2] = 2l\cos(\theta_1)\tfrac{\partial}{\partial\xi} \end{aligned} \tag{1.39}$$

then we solve this system for the fictitious inputs v_1, v_2, v_3 given a straight line trajectory linking q_{initial} and q_{final}:

$$\gamma(t) = (\gamma_\xi(t), \gamma_{\theta_1}(t), -2\gamma_{\theta_1}(t))^T \tag{1.40}$$

where

$$\begin{aligned} \gamma_\xi(t) &= \xi_i + \Delta\xi.t \\ \gamma_{\theta_1}(t) &= \theta_{1_i} + \Delta\theta_1.t \\ \Delta\xi &= \xi_f - \xi_i \\ \Delta\theta_1 &= \theta_{1_f} - \theta_{1_i} \end{aligned} \tag{1.41}$$

meaning that we solve

$$\dot{\gamma}(t) = b_1(\gamma(t))v_1 + b_2(\gamma(t))v_2 + b_3(\gamma(t))v_3 \tag{1.42}$$

which requires pseudo inverting a matrix

$$\begin{pmatrix} \Delta\xi \\ \Delta\theta_1 \\ -2\Delta\theta_1 \end{pmatrix} = \begin{pmatrix} 2l\sin(\gamma_{\theta_1}(t)) & 0 & 2l\cos(\gamma_{\theta_1}(t)) \\ 1 & 1 & 0 \\ -2 & -2 & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} \tag{1.43}$$

we choose the simplest solution of this system

$$\begin{pmatrix} v_1(t) \\ v_2(t) \\ v_3(t) \end{pmatrix} = \begin{pmatrix} 0 \\ \Delta\theta_1 \\ \frac{\Delta\xi}{2l\cos(\gamma_{\theta_1}(t))} \end{pmatrix} \tag{1.44}$$

given these inputs we solve the formal ordinary differential equation in a backward Philipp Hall basis of the Lie Algebra generated by b_1, b_2, b_3 (which happens to be (b_1, b_2, b_3))

$$\dot{S}(t) = S(t)(b_1v_1 + b_2v_2 + b_3v_3) \tag{1.45}$$

for which we search for a solution of the form

$$S(t) = e^{h_3(t)b_3}e^{h_2(t)b_2}e^{h_1(t)b_1} \tag{1.46}$$

by developing the formal exponentials to second order, we get the set of equations for the h_i functions:

$$\begin{cases} \dot{h}_1(t) = v_1 \\ \dot{h}_2(t) = v_2 \\ \dot{h}_3(t) = h_1(t)v_2 + v_3 \end{cases} \tag{1.47}$$

with the initial conditions $h_i(0) = 0$ for $i = 1, 2, 3$.

Integrating those equations gives us the "durations" for following each flow of the control field:

$$\begin{aligned} h_1(1) &= 0 \\ h_2(1) &= \Delta\theta_1 \\ h_3(1) &= \tfrac{\Delta\xi}{2l\Delta\theta_1}\ln\left|\tfrac{1}{\cos(\theta_{1_f})} + \tan(\theta_{1_f})\right| \end{aligned} \tag{1.48}$$

if $\Delta\theta_1 \neq 0$, or

$$\begin{aligned} h_1(1) &= 0 \\ h_2(1) &= 0 \\ h_3(1) &= \tfrac{\Delta\xi}{2l} \end{aligned} \tag{1.49}$$

if $\Delta\theta_1 = 0$.

Let's consider the case $\Delta\theta_1 \neq 0$.

If we denote $\phi_t^{b_i}$ the flow associated with the field b_i, the solution should thus be: follow $\phi_t^{b_1}$ for $t = 0s$, then follow $\phi_t^{b_2}$ for $t = \Delta\theta_1 s$, then follow $\phi_t^{b_3}$ for $t = \frac{\Delta\xi}{2l\Delta\theta_1}\ln|\frac{1}{\cos(\theta_{1_f})} + \tan(\theta_{1_f})|s$. However, the flow associated with $b_3 = [b_1, b_2]$ starting from x_0 could be rewritten, for $t > 0$,

$$\phi_t^{[b_1,b_2]}(x_0) = \phi_{\sqrt{t}}^{-b_2} \circ \phi_{\sqrt{t}}^{-b_1} \circ \phi_{\sqrt{t}}^{b_2} \circ \phi_{\sqrt{t}}^{b_1}(x_0) + \mathcal{O}(t) \tag{1.50}$$

Finally, denoting αu_i the command consisting in letting $u_i = 1$ for α seconds if $\alpha > 0$ and $u_i = -1$ for $-\alpha$ seconds if $\alpha < 0$ and denoting two successive controls by the

concatenation operation $\sharp$ we get our final sequence of commands (supposing for example that $\Delta\xi \geq 0$):

$$0u_1\sharp\Delta\theta_1 u_2\sharp\sqrt{\frac{\Delta\xi}{2l\Delta\theta_1}\ln\left|\frac{1}{\cos(\theta_{1_f})}+\tan(\theta_{1_f})\right|}(u_1\sharp u_2\sharp - u_1\sharp - u_2) \tag{1.51}$$

applied to the flows

$$\phi_t^{b_1}(x_0) = \begin{pmatrix} 2(\frac{\xi_0}{2}+\cos(\theta_{1_0})-\cos(t+\theta_{1_0})) \\ t+\theta_{1_0} \\ -2(t+\theta_{1_0}) \end{pmatrix} \tag{1.52}$$

and

$$\phi_t^{b_2}(x_0) = \begin{pmatrix} \xi_0 \\ t+\theta_{1_0} \\ -2(t+\theta_{1_0}) \end{pmatrix} \tag{1.53}$$

In the case $\Delta\theta_1 = 0$ the solution is simply

$$\sqrt{\frac{\Delta\xi}{2l}}(u_1\sharp u_2\sharp - u_1\sharp - u_2) \tag{1.54}$$

The solution is pictured in Fig. 1.16 in which the red curve represents the final output for an initial trajectory that is the black vertical line from 0 to 10.

Note that we do not reach the goal exactly, but with a bounded error [**LS93, GB02**]. The bound on the error allows us to reiterate this algorithm from the reached state as a new initial state until we reach the goal with a desired precision.

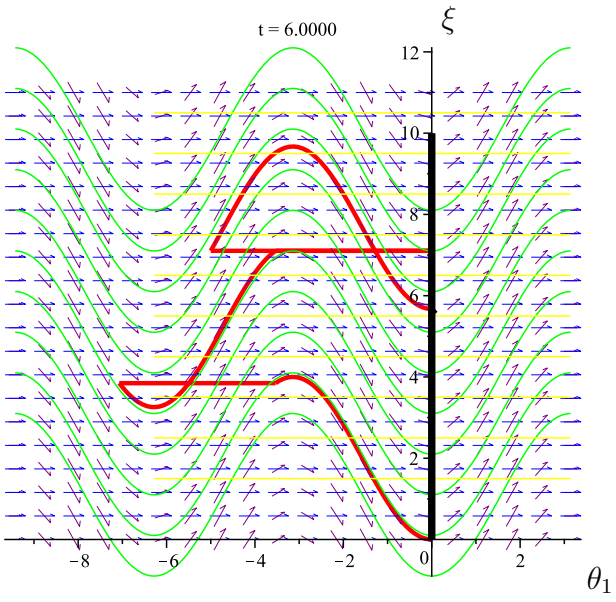


FIGURE 1.16. Solution in the (ξ, θ_1) plan. The initial trajectory is the thick black vertical segment drawn on ξ axis. The resulting solution is the red trajectory. In blue the $g_{1,2}$ control field, with its integral curves in yellow. In purple the $g_{1,1}$ control field, with its integral curves in green.

1.4.2. Manipulation robot. For R_2 we have similar properties to R_1. We just need to replace the variable ξ by the variable α. So let us consider the coordinate chart $(\alpha, \theta_1, \theta_2)$ in our configuration space manifold.

The equations of motion that are acting on the two strata are:

- On $\mathcal{S}_0$, the platform is fixed and we can write

$$\frac{d}{dt}\begin{pmatrix}\alpha\\ \theta_1\\ \theta_2\end{pmatrix} = \begin{pmatrix}0\\ 1\\ 0\end{pmatrix} u_1 + \begin{pmatrix}0\\ 0\\ 1\end{pmatrix} u_2 \tag{1.55}$$

- On $\mathcal{S}_1$, the end-effector is fixed in the platform's inertial frame as we consider a non-sliding contact, and thus the equation of motion is written

$$\frac{d}{dt}\begin{pmatrix}\alpha\\ \theta_1\\ \theta_2\end{pmatrix} = \begin{pmatrix}-2l\sin(\theta_1)\\ 1\\ -2\end{pmatrix} u_1 \tag{1.56}$$

The stratified driftless system is modelled by the two equations:

$$\begin{aligned}\dot{x} &= g_{0,1}(x)u_1 + g_{0,2}(x)u_2 \quad , \ x \in \mathcal{S}_0\\ \dot{x} &= g_{1,1}(x)u_1 \qquad\qquad\qquad\quad , \ x \in \mathcal{S}_1\end{aligned} \tag{1.57}$$

where

$$\begin{aligned}g_{0,1}(x) &= \tfrac{\partial}{\partial\theta_1}\\ g_{0,2}(x) &= \tfrac{\partial}{\partial\theta_2}\\ g_{1,1}(x) &= -2l\sin(\theta_1)\tfrac{\partial}{\partial\xi} + \tfrac{\partial}{\partial\theta_1} - 2\tfrac{\partial}{\partial\theta_2}\end{aligned} \tag{1.58}$$

PROPOSITION 1.4.3. *The underactuated kinematic control system R_2 is small time locally controllable in* $\mathrm{int}(\mathcal{C}_{\mathrm{free}})$

PROOF. See proof of proposition 1.4.1. □

Let's consider the gait $\mathcal{G} = (\mathcal{S}_1, \mathcal{S}_0, \mathcal{S}_1)$

PROPOSITION 1.4.4. *R_2 is gait-controllable with the gait $\mathcal{G}$.*

PROOF. See proof of proposition 1.4.2. □

We want now to plan a motion from a given $q_{\mathrm{initial}} = (\alpha_i, \theta_{1_i}, -2\theta_{1_i})^T$ to a given $q_{\mathrm{final}} = (\alpha_f, \theta_{1_f}, -2\theta_{1_f})$ in $\mathcal{C}_{\mathrm{free}}$.

Using the exact same method as for R_1, for $\Delta\theta_1 \neq 0$ and supposing for example that $\Delta\alpha \geq 0$, we get the solution:

$$0u_1\sharp\Delta\theta_1 u_2\sharp\sqrt{\frac{\Delta\alpha}{2l\Delta\theta_1}\ln\left|\frac{1}{\cos(\theta_{1_f})} + \tan(\theta_{1_f})\right|}(u_2\sharp u_1\sharp - u_2\sharp - u_1) \tag{1.59}$$

applied to the flows

$$\phi_t^{b_1}(x_0) = \begin{pmatrix}2(\frac{\alpha_0}{2} + \cos(t+\theta_{1_0}) - \cos(\theta_{1_0}))\\ t + \theta_{1_0}\\ -2(t+\theta_{1_0})\end{pmatrix} \tag{1.60}$$

and

$$\phi_t^{b_2}(x_0) = \begin{pmatrix} \xi_0 \\ t + \theta_{1_0} \\ -2(t + \theta_{1_0}) \end{pmatrix} \tag{1.61}$$

For $\Delta\theta_1 = 0$ we get:

$$\sqrt{\frac{\Delta\alpha}{2l}}(u_2 \sharp u_1 \sharp - u_2 \sharp - u_1) \tag{1.62}$$

1.4.3. L & M robot. R_3, with the switching modes control strategy introduced in section 1.3, can also be modelled as a stratified system.

Let us first see why R_3 cannot be directly modelled as a driftless stratified system if we do not consider this switching strategy. In this case when the rubber end-effector is in contact at fixed location in the platform's frame $\beta =$ constant then the system evolves in the submanifold defined by the implicit equation:

$$\xi + 2l\cos(\theta_1) + \beta = \alpha \tag{1.63}$$

the derivation with respect to time t leads:

$$\dot{\xi} - 2l\sin(\theta_1)\dot{\theta}_1 = \dot{\alpha} \tag{1.64}$$

writing $\dot{\theta}_1 = u_1$ we get a system of the form

$$A\dot{x} = \sum_i g_i(x)u_i \tag{1.65}$$

where A is a non invertible matrix and thus cannot be written in the desired form

$$\dot{x} = \sum_i g_i(x)u_i \tag{1.66}$$

Now back to the switching control strategy. The equations of motions acting on the two strata are:

- on $\mathcal{S}_0$:

$$\frac{d}{dt}\begin{pmatrix} \xi \\ \theta_1 \\ \theta_2 \\ \alpha \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} u_1 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} u_2 \tag{1.67}$$

- on $\mathcal{S}_1$, in manipulation state:

$$\frac{d}{dt}\begin{pmatrix} \xi \\ \theta_1 \\ \theta_2 \\ \alpha \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ -2 \\ -2l\sin(\theta_1) \end{pmatrix} u_1 \tag{1.68}$$

- on $\mathcal{S}_1$, in locomotion state:

$$\frac{d}{dt}\begin{pmatrix} \xi \\ \theta_1 \\ \theta_2 \\ \alpha \end{pmatrix} = \begin{pmatrix} 2l\sin(\theta_1) \\ 1 \\ -2 \\ 0 \end{pmatrix} u_1 \tag{1.69}$$

As we can see, two different equations of motion are acting on the bottom stratum $\mathcal{S}_1$. They correspond to two control vector fields defined on $\mathcal{S}_1$. Since the solution produced by the method of [**GB02**] consists in following the vector fields sequentially and never a linear combination of the vector fields, we can use it for R_3 to produce the control sequence with the state-switching control nested in the solution.

So we want to steer the system from a given $q_{\text{initial}} = (\xi_i, \theta_{1_i}, -2\theta_{1_i}, \alpha_i)^T$ to a given $q_{\text{final}} = (\xi_f, \theta_{1_f}, -2\theta_{1_f}, \alpha_f)^T$. We first derive equation the stratified driftless system on the bottom stratum:

$$\dot{x} = g_1(x)u_1 + g_2(x)u_2 + g_3(x)u_3 \tag{1.70}$$

with

$$\begin{aligned} g_1(x) &= \begin{pmatrix} 0 \\ 1 \\ -2 \\ 0 \end{pmatrix}, g_2(x) = \begin{pmatrix} 2l\sin(\theta_1) \\ 1 \\ -2 \\ 0 \end{pmatrix} \\ g_3(x) &= \begin{pmatrix} 0 \\ 1 \\ -2 \\ -2l\sin(\theta_1) \end{pmatrix} \end{aligned} \tag{1.71}$$

we then extend the system by adding vector fields from $\text{Lie}(g_1, g_2, g_3)$:

$$\dot{x} = b_1 v_1 + b_2 v_2 + b_3 v_3 + b_4 v_4 + b_5 v5 \tag{1.72}$$

where

$$\begin{aligned} b_1 &= g_1 \\ b_2 &= g_2 \\ b_3 &= g_3 \\ b_4 &= [g_1, g_2] = \begin{pmatrix} 2l\cos(\theta_1) \\ 0 \\ 0 \\ 0 \end{pmatrix} \\ b_5 &= [g_1, g_3] = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -2l\cos(\theta_1) \end{pmatrix} \end{aligned} \tag{1.73}$$

(Note: we stop to second order and we do not need to add $[g_2, g_3] = b_5 - b_4$), then we solve this system for the fictitious inputs v_1, v_2, v_3, v_4, v_5 given a straight line trajectory linking q_{initial} and q_{final}:

$$\gamma(t) = (\gamma_\xi(t), \gamma_{\theta_1}(t), -2\gamma_{\theta_1}(t), \gamma_\alpha(t))^T \tag{1.74}$$

where

$$\begin{aligned}
&\gamma_\xi(t) = \xi_i + \Delta\xi.t \\
&\gamma_\alpha(t) = \alpha_i + \Delta\alpha.t \\
&\gamma_{\theta_1}(t) = \theta_{1_i} + \Delta\theta_1.t \\
&\Delta\xi = \xi_f - \xi_i \\
&\Delta\alpha = \alpha_f - \alpha_i \\
&\Delta\theta_1 = \theta_{1_f} - \theta_{1_i}
\end{aligned} \tag{1.75}$$

so we solve

$$\dot{\gamma}(t) = b_1(\gamma(t))v_1 + b_2(\gamma(t))v_2 + b_3(\gamma(t))v_3 + b_4(\gamma(t))v_4 + b_5(\gamma(t))v_5 \tag{1.76}$$

which requires pseudo inverting the matrix

$$\begin{pmatrix} \Delta\xi \\ \Delta\theta_1 \\ -2\Delta\theta_1 \\ \Delta\alpha \end{pmatrix} = \begin{pmatrix} 0 & 2l\sin(\gamma_{\theta_1}) & 0 & 2l\cos(\gamma_{\theta_1}) & 0 \\ 1 & 1 & 1 & 0 & 0 \\ -2 & -2 & -2 & 0 & 0 \\ 0 & 0 & -2l\sin(\gamma_{\theta_1}) & 0 & -2l\cos(\gamma_{\theta_1}) \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{pmatrix} \tag{1.77}$$

once again we choose the simplest solution of this system

$$\begin{pmatrix} v_1(t) \\ v_2(t) \\ v_3(t) \\ v_4(t) \\ v_5(t) \end{pmatrix} = \begin{pmatrix} \Delta\theta_1 \\ 0 \\ 0 \\ \frac{\Delta\xi}{2l\cos(\gamma_{\theta_1}(t))} \\ -\frac{\Delta\alpha}{2l\cos(\gamma_{\theta_1}(t))} \end{pmatrix} \tag{1.78}$$

given these inputs we solve the formal ordinary differential equation in a backward Philipp Hall basis of the Lie Algebra generated by b_1, b_2, b_3, b_4, b_5 which is also $(b_1, b_2, b_3, b_4, b_5)$

$$\dot{S}(t) = S(t)(b_1v_1 + b_2v_2 + b_3v_3 + b_4v_4 + b_5v_5) \tag{1.79}$$

for which we search for a solution of the form

$$S(t) = e^{h_5(t)b_5}e^{h_4(t)b_4}e^{h_3(t)b_3}e^{h_2(t)b_2}e^{h_1(t)b_1} \tag{1.80}$$

by developing the formal exponentials to second order, we get the set of equations for the h_i functions:

$$\begin{cases} \dot{h}_1 = v_1 \\ \dot{h}_2 = v_2 \\ \dot{h}_3 = v_3 \\ -\dot{h}_2h_1 + \dot{h}_3h_2 + \dot{h}_4 = v_4 \\ -\dot{h}_3h_1 - \dot{h}_3h_2 + \dot{h}_5 = v_5 \end{cases} \tag{1.81}$$

with the initial conditions $h_i(0) = 0$ for $i = 1, 2, 3, 4, 5$.

Integrating those equations gives us the "durations" for following each flow of the control field:

$$\begin{aligned} h_1(1) &= \Delta\theta_1 \\ h_2(1) &= 0 \\ h_3(1) &= 0 \\ h_4(1) &= \tfrac{\Delta\xi}{2l\Delta\theta_1} \ln \left| \tfrac{1}{\cos(\theta_{1_f})} + \tan(\theta_{1_f}) \right| \\ h_5(1) &= -\tfrac{\Delta\alpha}{2l\Delta\theta_1} \ln \left| \tfrac{1}{\cos(\theta_{1_f})} + \tan(\theta_{1_f}) \right| \end{aligned} \tag{1.82}$$

if $\Delta\theta_1 \neq 0$, or

$$\begin{aligned} h_1(1) &= 0 \\ h_2(1) &= 0 \\ h_3(1) &= 0 \\ h_4(1) &= \tfrac{\Delta\xi}{2l} \\ h_5(1) &= \tfrac{\Delta\alpha}{2l} \end{aligned} \tag{1.83}$$

if $\Delta\theta_1 = 0$.

Finally, for $\Delta\theta_1 \neq 0$ and supposing for example that $\Delta\xi \geq 0$ and $\Delta\alpha \geq 0$, we get the solution:

$$\begin{aligned} &\Delta\theta_1 u_1 \sharp 0 u_2 \sharp 0 u_3 \\ &\sharp \sqrt{\frac{\Delta\xi}{2l\Delta\theta_1} \ln |\frac{1}{\cos(\theta_{1_f})} + \tan(\theta_{1_f})|} (u_1 \sharp u_2 \sharp - u_1 \sharp - u_2) \\ &\sharp \sqrt{\frac{\Delta\alpha}{2l\Delta\theta_1} \ln |\frac{1}{\cos(\theta_{1_f})} + \tan(\theta_{1_f})|} (u_3 \sharp u_1 \sharp - u_3 \sharp - u_1) \end{aligned} \tag{1.84}$$

applied to the flows

$$\begin{aligned} \phi_t^{b_1}(x_0) &= \begin{pmatrix} \xi_0 \\ t + \theta_{1_0} \\ -2(t + \theta_{1_0}) \\ \alpha_0 \end{pmatrix} \\ \phi_t^{b_2}(x_0) &= \begin{pmatrix} 2(\frac{\xi_0}{2} - \cos(t + \theta_{1_0}) + \cos(\theta_{1_0})) \\ t + \theta_{1_0} \\ -2(t + \theta_{1_0}) \\ \alpha_0 \end{pmatrix} \\ \phi_t^{b_3}(x_0) &= \begin{pmatrix} \xi_0 \\ t + \theta_{1_0} \\ -2(t + \theta_{1_0}) \\ 2(\frac{\alpha_0}{2} + \cos(t + \theta_{1_0}) - \cos(\theta_{1_0})) \end{pmatrix} \end{aligned} \tag{1.85}$$

For $\Delta\theta_1 = 0$ we get:

$$\sqrt{\frac{\Delta\xi}{2l}}(u_1 \sharp u_2 \sharp - u_1 \sharp - u_2) \sharp \sqrt{\frac{\Delta\alpha}{2l}}(u_3 \sharp u_1 \sharp - u_3 \sharp - u_1) \tag{1.86}$$

The solution is pictured in Figs. 1.17 and 1.18.

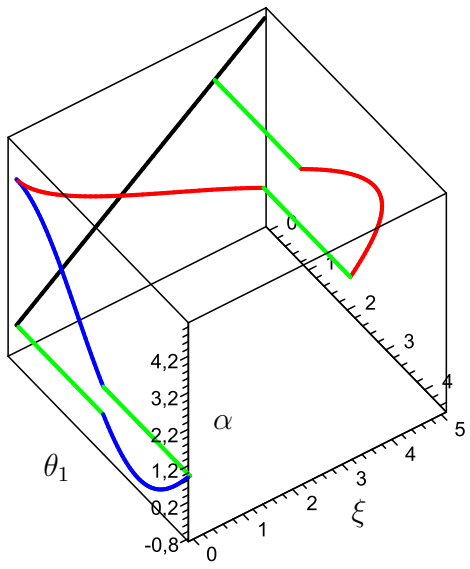


FIGURE 1.17. Trajectory planning for R_3. The colors used are the same as in Fig.1.15. The initial trajectory, which violates the foliation, is the black diagonal segment on the left-back face of the cube, the resulting trajectory is the red-blue-green trajectory that follows the foliations.

1.5. Dynamic Trajectory Planning Approach

In the previous sections we were primarily concerned by geometric path planning, even though section 1.4 tackled the problem from a kinematic trajectory planning perspective. In this section the objective is to generate torque-driven dynamically valid trajectories in the state space $T\mathcal{C}$ (the tangent bundle of the smooth manifold $\mathcal{C}$).

1.5.1. Locomotion robot. First let us study the case of the robot R_1. We would like to generate dynamically valid trajectories (open-loop control laws) for both the transfer and the transit paths.

PROBLEM 1.5.1. *Given* $(q_{\text{initial}}, \dot{q}_{\text{initial}}), (q_{\text{final}}, \dot{q}_{\text{final}}) \in T\mathcal{C}_{\text{free}}$ *and a geometric path* $p : [0,1] \to \mathcal{C}_{\text{free}}$ *such that* $p(0) = q_{\text{initial}}$ *and* $p(1) = q_{\text{final}}$, *find* $t_f \in \mathbb{R}$ *and a re-parametrization of* $\text{Tr}(p)$ $\gamma : [0, t_f] \to \mathcal{C}_{\text{free}}$ *such that* γ *realizes the dynamics equations of motion of* R_1 *along the path, under a Coulomb friction model hypothesis.*

The efforts applied on R_1 in each of the two strata representing the two contact modes are portrayed in Fig. 1.19.

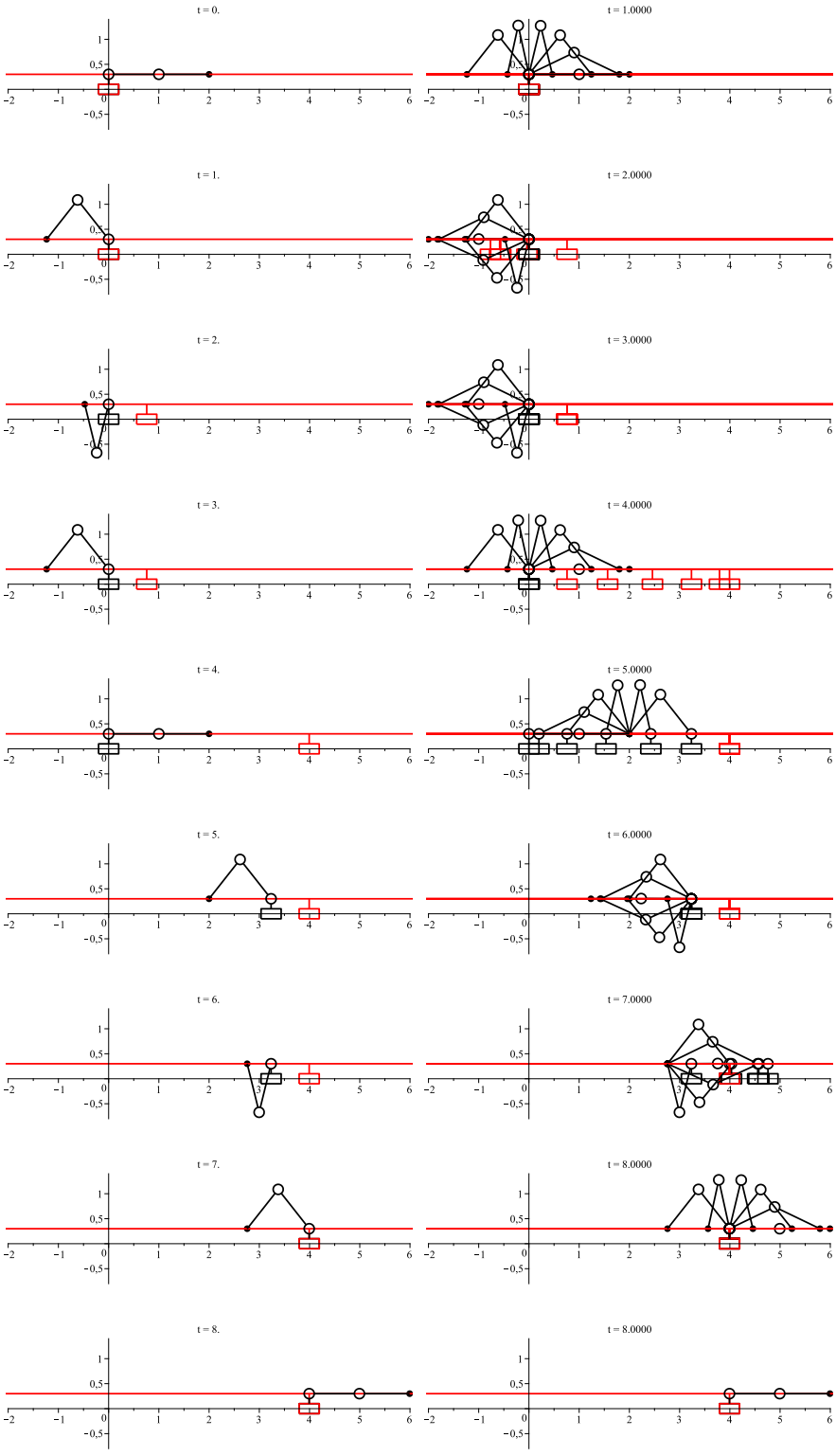
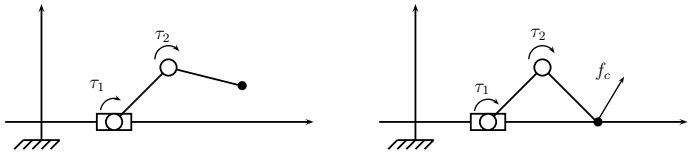


FIGURE 1.18. Solution of the trajectory planning for the R_3 system. The sliding of the black rectangle and the red rectangle along the horizontal axis illustrate respectively the locomotion and the manipulation components of the motion. The first column displays snapshots of the motion taken at times of change of control fields (points where the curve in Fig. 1.17 changes color). The second column represents the transition motions between two successive snapshots.



(a) Free mode, stratum $\mathcal{S}_0$ (b) Contact mode, stratum $\mathcal{S}_1$

FIGURE 1.19. Forces and torques in the two modes

Using the Lagrangian approach, the dynamics of the system can be written as

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + N(q,\dot{q}) - J(q)^T f_c = \begin{pmatrix} 0 \\ \tau_1 \\ \tau_2 \end{pmatrix} \tag{1.87}$$

which is in $\mathcal{S}_1$ (contact mode) when $f_c \neq 0$ and in $\mathcal{S}_0$ (free mode) when $f_c = 0$; M, C, N, J denote respectively the inertia matrix, the Coriolis and centrifugal effects, the external efforts (gravity, joint friction) vector, and the Jacobian matrix of the robot.

In the free mode, we can notice that $\ddot{\xi} \neq 0$ provided that the inertial effects of moving the links will cause a dynamic reaction on the base. In the following we will consider these links dynamics effects as perturbations and neglect them, which means that on the free mode $\ddot{\xi} = 0$.

Let us now focus on the contact mode, which is our main concern in this study; f_c is the Lagrange multiplier associated with the Lagrangian model of the system under the Pfaffian constraint $J(q)\dot{q} = 0$. Solving the dynamic and the Pfaffian constraint equations for f_c and $\ddot{q}$ leads

$$f_c = -(JM^{-1}J^T)^{-1}(JM^{-1}(\tau - C\dot{q} - N) + \dot{J}\dot{q}) \tag{1.88}$$

where $\tau = (0, \tau_1, \tau_2)^T$.

To avoid sliding, f_c has to lie within the Coulomb friction cone $\mathcal{F}$:

$$f_c \in \mathcal{F} \tag{1.89}$$

and

$$\mathcal{F} = \{(f_x, f_y) \in \mathbb{R}^2 \mid f_y \geq 0 \text{ and } |f_x| \leq \mu f_y\} \tag{1.90}$$

Now, we derive an open-loop control law $t \mapsto (\tau_1(t), \tau_2(t))$ which steers the system from an initial contact state $(q_i, \dot{q}_i)$ to a final state $(q_f, \dot{q}_f)$ maintaining a non-sliding contact with the ground. To do so we will adapt some of the ideas that were introduced in [**SEM05**].

To make the derivations easier we will neglect the masses of the links and consider only the mass of the sliding base m_0. The dynamics equations become:

$$\begin{cases} m_0\ddot{\xi} = f_x \\ f_x.(\sin(\theta_1) + l\sin(\theta_1+\theta_2)) - f_y.(\cos(\theta_1) + \cos(\theta_1+\theta_2)) = \tau_1/l \\ f_x.\sin(\theta_1+\theta_2) - f_y.\cos(\theta_1+\theta_2) = \tau_2/l \end{cases} \tag{1.91}$$

In a given contact mode, the system evolves in a one-dimensional submanifold of the configuration space, a leaf of the stratum $\mathcal{S}_1$, that we will parametrize with ξ. For

example, if the contact is fixed at the abscissa 0 then $\xi = -2l\cos(\theta_1)$ and $\theta_2 = -2\theta_1$. Solving equation (1.91) for f_x and f_y gives us

$$\begin{cases} f_x = \frac{\tau_1 - 2\tau_2}{\sqrt{4l^2 - \xi^2}} \\ f_y = \frac{\tau_1}{\xi} \end{cases} \tag{1.92}$$

and the friction cone condition $f_c \in \mathcal{F}$, together with the maximum torques conditions $|\tau_1| \leq \tau_{\max}$ and $|\tau_2| \leq \tau_{\max}$ yields the following torque cone condition

$$A_\xi \begin{pmatrix} \tau_1 \\ \tau_2 \end{pmatrix} \leq b_\xi \tag{1.93}$$

where

$$A_\xi = \begin{pmatrix} \frac{1}{\sqrt{4l^2-\xi^2}} - \frac{\mu}{\xi} & -\frac{2}{\sqrt{4l^2-\xi^2}} \\ -\frac{1}{\sqrt{4l^2-\xi^2}} - \frac{\mu}{\xi} & \frac{2}{\sqrt{4l^2-\xi^2}} \\ -1 & 0 \\ 1 & 0 \\ 0 & -1 \\ 0 & 1 \end{pmatrix} \tag{1.94}$$

and

$$b_\xi = \begin{cases} (0,0,0,\tau_{\max},\tau_{\max},\tau_{\max})^T & ,\xi > 0 \\ (0,0,\tau_{\max},0,\tau_{\max},\tau_{\max})^T & ,\xi < 0 \end{cases} \tag{1.95}$$

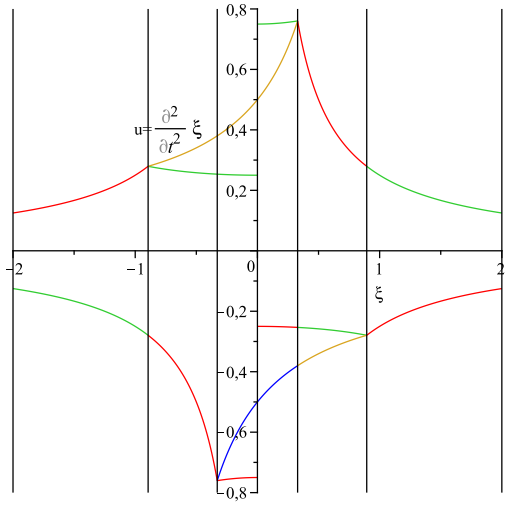
Finally, the open-loop dynamic trajectory planning reduces to

$$\begin{aligned} \ddot{\xi} = f(\xi,\tau_1,\tau_2) &= \frac{\tau_1 - 2\,\tau_2}{m_0\sqrt{4l^2-\xi^2}} \\ \text{under the constraint} \quad A_\xi &\begin{pmatrix} \tau_1 \\ \tau_2 \end{pmatrix} \leq b_\xi \end{aligned} \tag{1.96}$$

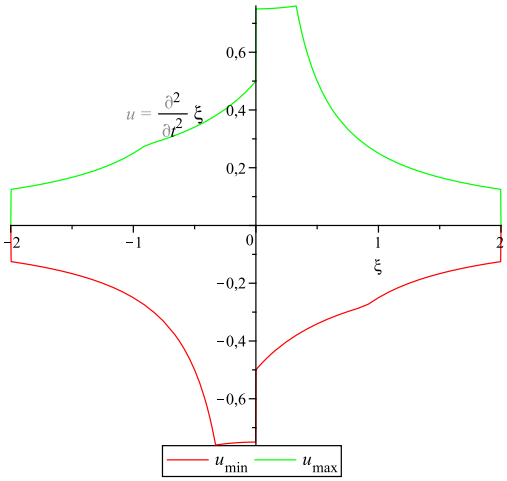
or, putting C_ξ the line matrix $C_\xi = \frac{1}{m_0\sqrt{4l^2-\xi^2}}\begin{pmatrix} 1 & -2 \end{pmatrix}$,

$$\ddot{\xi} = C_\xi \begin{pmatrix} \tau_1 \\ \tau_2 \end{pmatrix} \text{ under the constraint } \quad A_\xi \begin{pmatrix} \tau_1 \\ \tau_2 \end{pmatrix} \leq b_\xi \tag{1.97}$$

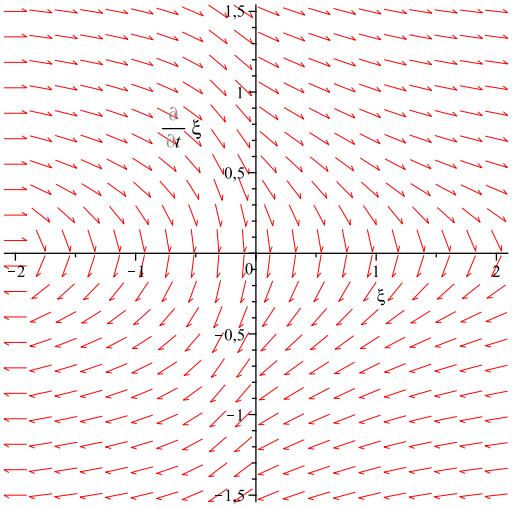
projecting in the space of task freedom [**SEM05**], using the change of control input $u = C_\xi \begin{pmatrix} \tau_1 \\ \tau_2 \end{pmatrix}$, we get the simple double integrator $\ddot{\xi} = u$ where the torque cone condition translates into bounds on acceleration $u_{\min}(\xi) \leq u \leq u_{\max}(\xi)$. The time-optimal solution for this problem is known as the "bang-bang" control law [**BDG85**], which consists in applying maximal acceleration forward from the initial state, maximal deceleration (*ie.* minimal acceleration) backward from the final state, and switching between those two commands at the intersection point of the two trajectories obtained, see Fig. 1.20.



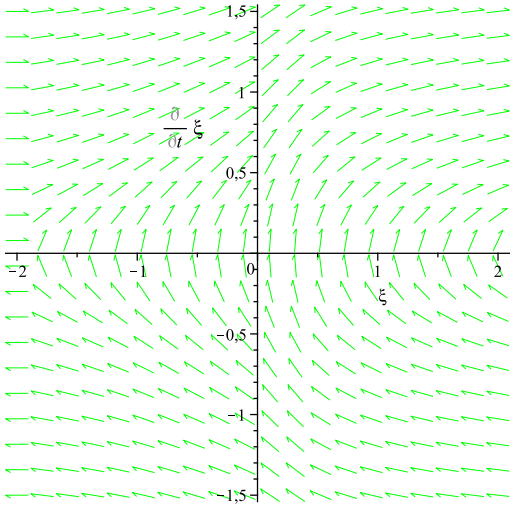
(a) Projection of torque cone in the $\ddot{\xi}$ space



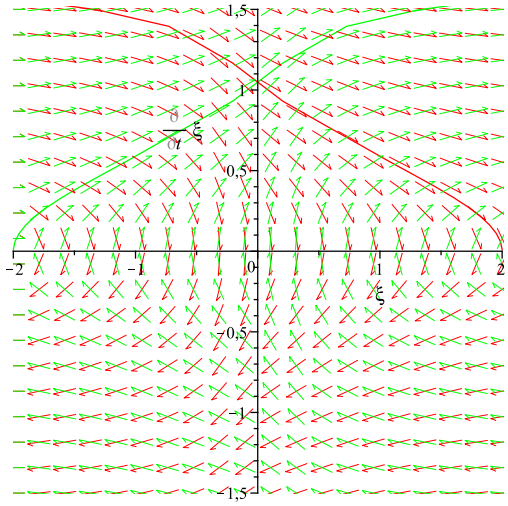
(b) Maximum acceleration and deceleration



(c) Control field associated with $u_{\min}$



(d) Control field associated with $u_{\max}$



(e) Numerical integration and final solution

FIGURE 1.20. Bang-bang control law synthesis

1.5.2. Manipulation robot. Similarly to R_1, we now consider the robot R_2 in the contact stratum $\mathcal{S}_1$. Let now m_0 denote the mass of the sliding platform.

Let $f_c = (f_x, f_y) \in \mathbb{R}^2$ be the contact force applied by the sliding platform on the end-effector of the manipulator.

Instead of writing the Newton's second law of motion applied to the platform and concatenate it with the Lagrangian dynamics of the manipulator coupled with the non-sliding contact point kinematic condition, we exhibit the equivalence with R_1 by directly writing the constrained Lagrangian problem for the whole system defined by its generalized coordinates $q = (\alpha, \theta_1, \theta_2)^T$. We get a similar equation to (1.87) where f_c must be seen as the Lagrangian multiplier associated with $J(q)\dot{q} = 0$, that we derived by differentiating the holonomic contact constraint

$$h(q) = \alpha - 2\cos(\theta_1) + \text{constant} = 0 \tag{1.98}$$

Thus, the following derivations follow the exact same scheme as for R_1, and we can control the variable α using the bang-bang control law.

1.5.3. L & M robot. The dynamics and control of R_3 with the switching states control strategy considered in section 1.3 simply reduce to the dynamics and control of R_1 and R_2 separately in each of the states of the system, respectively the locomotion and manipulation mode.

Here we will rather derive the dynamics of R_3 in contact mode (stratum $\mathcal{S}_1$) without the switching control input strategy. This can be seen as the dynamics of motion realized by taking a contact support on a mobile piece of the environment by the robot R_1, performing manipulation and locomotion at the same time.

Once again, instead of writing separately the dynamics of the subsystems made of the locomotor/manipulator coupling them with the non-sliding contact point kinematic constraint, we directly consider the whole system $q = (\xi, \theta_1, \theta_2, \alpha)^T$ and write its Lagrangian equation of motion under the Pfaffian constraint which derives from the holonomic constraint

$$h(q) = \begin{pmatrix} \xi + l\cos(\theta_1) + l\cos(\theta_1 + \theta_2) - \alpha + \text{constant} \\ l\sin(\theta_1) + l\sin(\theta_1 + \theta_2) = 0 \end{pmatrix} = 0 \tag{1.99}$$

So we get the following equation

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + N(q, \dot{q}) - J(q)^T f_c = \begin{pmatrix} 0 \\ \tau_1 \\ \tau_2 \\ 0 \end{pmatrix} \tag{1.100}$$

Let us now generate a control law for R_3 in contact mode under Coulomb friction hypothesis. For the sake of clarity and without loss of generality we suppose that the end effector is fixed at the location $\beta = 0$. Neglecting the masses of the link gives us the following dynamics equation:

$$\begin{pmatrix} m_\xi & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \vdots \\ \vdots & \ddots & 0 & 0 \\ 0 & \cdots & 0 & m_\alpha \end{pmatrix} \begin{pmatrix} \ddot{\xi} \\ \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{\alpha} \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & \alpha - \xi \\ \sqrt{1 - (\frac{\alpha - \xi}{2})^2} & \frac{\alpha - \xi}{2} \\ -1 & 0 \end{pmatrix} \begin{pmatrix} f_x \\ f_y \end{pmatrix} = \begin{pmatrix} 0 \\ \tau_1 \\ \tau_2 \\ 0 \end{pmatrix} \tag{1.101}$$

If we introduce the new variable $\delta = \xi - \alpha$ which expresses the displacement of the locomotor in the platform's inertial frame, we can rewrite the equation as

$$\begin{cases} \frac{m_\xi - m_\alpha}{2}\ddot{\delta} + f_x = 0 \\ -\delta f_y = \tau_1 \\ \sqrt{1 - \frac{\delta^2}{4}} f_x - \frac{\delta}{2} f_y = \tau_2 \end{cases} \tag{1.102}$$

which are the same equations as the ones we got respectively for ξ and α in R_1 and R_2, with a virtual mass $m_0 = \frac{m_\xi - m_\alpha}{2}$. We can thus control the variable δ with the same control law, which is not any more valid for ξ and α separately.

1.6. Conclusion

The motion planning and control problems for the systems R_1 (locomotion-only), R_2 (manipulation-only), and R_3 (simultaneous locomotion and manipulation) are solvable with the same tools. We proved the reduction property for all three systems. This property reduces the path planning problem in a foliated configuration space to a classical path planning problem in a non-foliated space. The formulation of the motion planning problem as a BVP was written for the three systems, and non-holonomic trajectory planning techniques were used for solving this problem. Dynamics derivations appeared also to be equivalent for the three robots. The Lagrangian approach acting on one global system with generalized coordinates instead of different subsystems as traditionally considered turned out to be a powerful unification tool for making these derivations.

We thus successfully applied a set of three motion planning methods for the three systems R_1, R_2, and R_3. By doing so, we showed that our initial paradigm, the simultaneous non-decoupled locomotion and manipulation planning, holds using any of these methods.

Though being theory-oriented, the study was primarily motivated by the practical humanoid robot motion planning issues, given that a humanoid robot is a platform with both locomotion and manipulation capabilities. However, the particular derivations presented here require low dimensionality that is not found is such humanoid systems. Nevertheless, scaling this theoretical paradigm to practical real robots can be achieved by using adequate complexity-reduction strategies (for example the planning method of [**SLCS04**] is applied to multiple-dof manipulators using a sampling-based strategy). As for humanoids, by adapting our paradigm to the contact-before-motion strategy [**Bre06**], it becomes conceivable to integrate and fuse works done on humanoid locomotion planning [**EKM06, HBL$^+$08**] and dexterous manipulation planning [**YSY03, SSEKP07, XKL07**], or any other type of whole-body manipulation planning [**YKH04, EAPL06**], within one non-decoupled planning framework. This is the purpose of the next chapter (Chapter 2). As an example, Fig. 1.21 illustrates the non-decoupled locomotion and manipulation planning for a humanoid robot as presented in this chapter. In this scenario, the goal configuration is specified in terms of a goal global position of the humanoid robot and a goal orientation for the manipulated box. Our non-decoupled planner plans a sequence of contacts between the three interacting entities (humanoid robot, box, environment) in order to reach the goal.

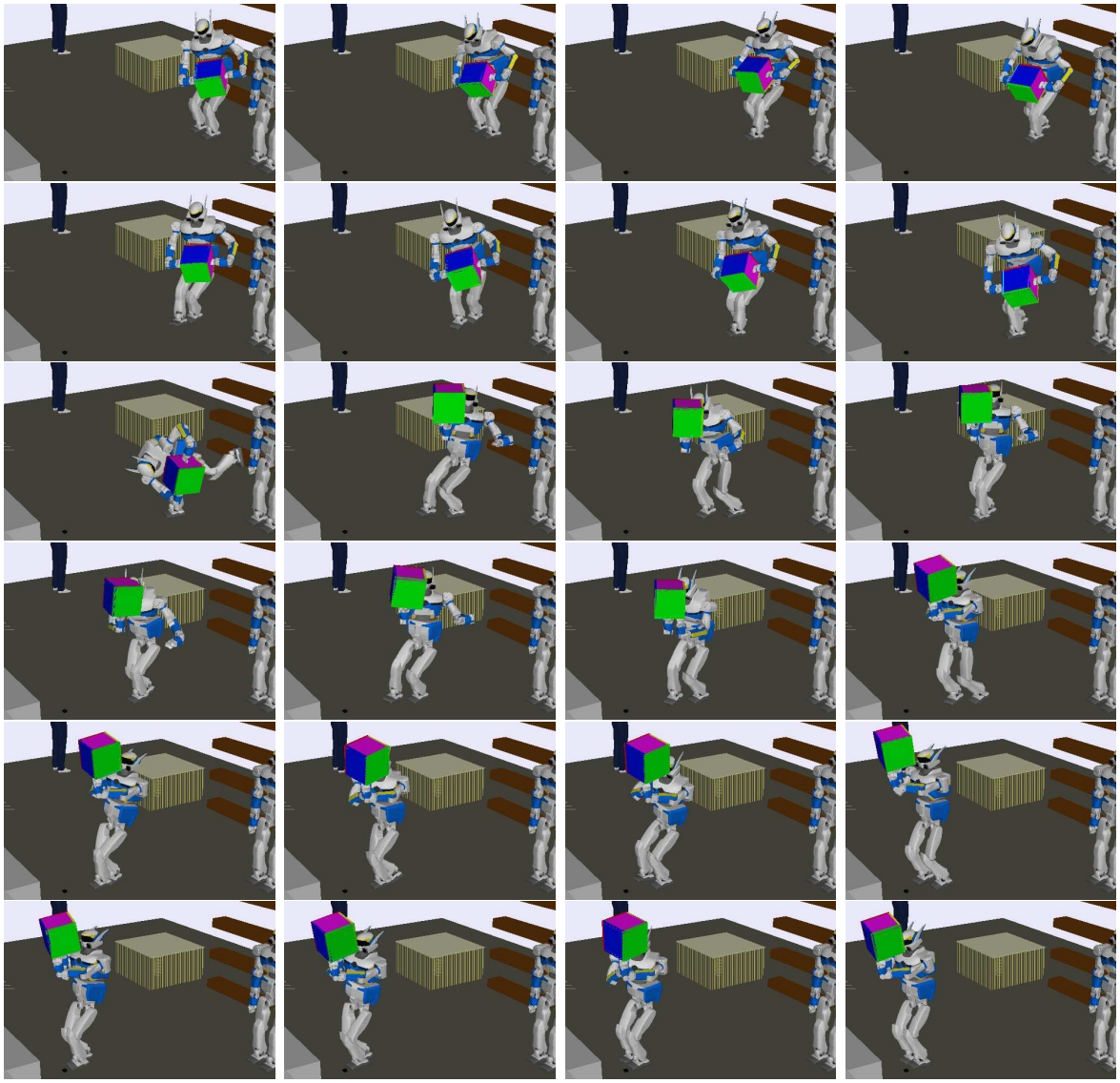


FIGURE 1.21. The objective is for the HRP-2 robot to advance 2 m forward while simultaneously performing half rotation of the 5 kg box, bringing the purple face up. Friction coefficients between the hands and the box are set to $\mu = 1$. The robot autonomously re-grasps and rotates the box while walking.

CHAPTER 2

Multi-Contact Stances Planning for Multiple Agents

We move to the full-scale systems from here on, and we propose in this second chapter a generalized framework together with an algorithm to plan a discrete sequence of multi-contact stances that brings a set of collaborating robots and manipulated objects from a specified initial configuration to a desired goal through non-gaited acyclic contacts with their environment or among each other. The broad range of applications of this generic algorithm includes legged locomotion planning, whole-body manipulation planning, dexterous manipulation planning, as well as any multi-contact-based motion planning problem that might combine several of these sub-problems, thus bridging the gap with the non-decoupled planning for low-dimensional systems as introduced in Chapter 1. We demonstrate the versatility of our planner through example scenarios taken from the aforementioned classes of problems in virtual environments.

2.1. Introduction

Recent works [**Esc08, Hau08**] started tackling the acyclic motion planning problem for humanoid and/or legged robots taking a *contacts-before-motion* planning approach. As opposed to the footstep planning problem [**KNK$^+$01, CKNK03, CLK$^+$05, TvdP98, KMB95**], they target higher level "intelligence" of the robots by erasing the knowledge-based specification of a bipedal or quadrupedal gait. The approach is based on planning a feasible sequence of stances from an initial configuration to a goal configuration, before planning the subsequent continuous motion that goes through this planned sequence of stances. This chapter is concerned only with the first part of the problem, i.e. the discrete stances sequence planning sub-problem. Such a decoupling scheme of the two components of the problem, though less theoretically founded in terms of completeness than the interleaved approach of *multi-modal planning* [**HL08**], enables us to reduce the complexity of the problem and yet still manages to solve highly constrained situations as demonstrated on practical real-life humanoid robot experiments [**EKM06, EKMG08, EK09**].

The core algorithm we are using here was first introduced in the works of Escande et al. [**EKM06**]. In its most reduced form, it is a *Best-First Planning* (BFP) algorithm [**Lat91, LaV06**] that explores the continuum of the workspace for finding best contact locations, as opposed to the main other method first introduced in the works of Hauser et al. [**HBL05**] requiring prior discretization of possible contact locations on the environment. A major drawback of this latter approach resides in the difficult trade-off between the possible combinatorial issues that would be raised by too many pre-discretized locations, versus the possible misses of solutions induced by too few pre-discretized locations (The "continuous modes" problem as raised in [**HL10**] and [**HNTH11**]).

In this chapter we build on this BFP-based algorithm and propose a novel framework that makes it possible, with one unique planner, to solve different classes of robotics contacts planning problems, beyond the initially targeted "legged locomotion for a single robot" problem. Such a planner can solve, for example, the non-gaited dexterous manipulation problem, some example approaches of which can be found in the past few years' literature [**SSEKP07, XKL07, YSY03, MA04**]. A more original contribution is to solve the contacts planning problem for collaborative robots manipulating objects [**EAPL06**]. The needed synchronization of contacts planning for the cooperative carrying of a heavy object by two humanoid robots is one example of the results of the planner. Additionally, by unifying locomotion and manipulation, the planner can also solve contact planning problems for situations interleaving both, which can prove useful for platforms such as humanoid robots that are designed to execute both locomotion and manipulation tasks.

These contributions (extension to multiple agents, generalization to any robotic platform, and non-decoupling of locomotion and manipulation) are made possible thanks to a formulation of the problem that reaches a higher level of abstraction, necessary in order to achieve the desired generalization. It allows us to make the extensions listed above with little rewriting effort of the existing algorithms. In other words, the algorithms here are the same as their original form [**Esc08, Hau08**]; by generalizing the formalism and the framework, we extend their capabilities to a wider range of applications. This is our main contribution. The approach retained for the multi-robot systems is a centralized one. Decentralized strategies such as prioritized planning [**ELP87, vdBO05**] or fixed-path coordination [**SLL02**] are not directly applicable since the nature of our planning is different and does not occur in the configuration space but rather in a different-nature stances set.

The rest of this chapter is organized as follows. We first propose a formulation of the problem using the language and formalism of basic set theory (Section 2.2). We then write our algorithm in this synthetic language and compare it with the other existing method (Section 2.3). Last we demonstrate some results obtained by our planner on different classes of problems (Section 2.4).

2.2. Preliminary Notations and Definitions

In this section we will introduce the set-theoretic formalism that will make the extensions and the locomotion-manipulation unification process easier to write. The abstraction effort invested in this section will later be rewarded in the algorithms writing section (Section 2.3). It will allow us to write the algorithms in a very generic, synthetic, and rigorous style. It might be useful to recall beforehand that, within this formalism, for any two sets A and B, $p : A \to B$ denotes a mapping from A to B, $\mathscr{P}(A)$ denotes the power set of A (set of subsets of A), $\text{card}(A)$ the cardinality of A, and for any two subsets $A' \in \mathscr{P}(A)$ and $B' \in \mathscr{P}(B)$, $p(A')$ and $p^{-1}(B')$ denote respectively the direct and inverse images of A' and B' under the mapping p. We use the symbol $A_1 \setminus A_2$ to denote the difference of two subsets $A_1, A_2 \in \mathscr{P}(A)$.

So let us suppose we have a system of N robots indexed in the set $\{1, \ldots, N\}$. A "robot" here is either a fully- or under-actuated articulated mechanism or a non-actuated manipulated object. The environment can also be considered as a special case of "robot", indexed with 0. Thus the index set $\{0, \ldots, N\}$ contains all the articulated mechanisms, the manipulated objects, and the environment.

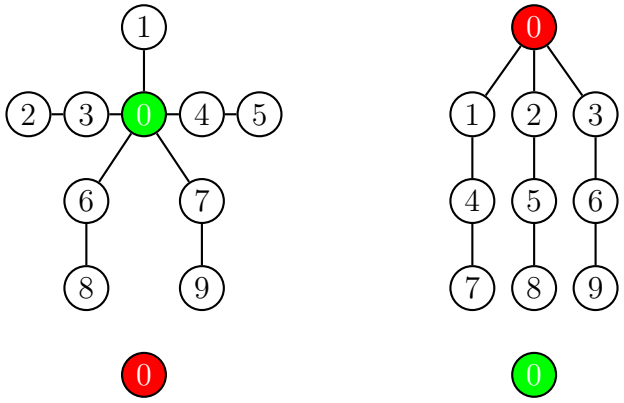


FIGURE 2.1. Examples of the 4 types of kinematic trees yielding configuration space. Top left: a humanoid robot. Top right: a dexterous hand. Bottom left: the environment. Bottom right: a manipulated object. In red: fixed base. In green: free-flying base. A system of robots consists of an arbitrary number of any of those 4 types of kinematics trees.

Each robot $r \in \{0, \dots, N\}$ can be represented as a kinematic tree made of b_r bodies (nodes of the tree) indexed in $\{0, \dots, b_r - 1\}$, linked by j_r actuated joints (edges of the tree) indexed in $\{1, \dots, j_r\}$ (or $\varnothing$ if $j_r = 0$). See Fig. 2.1.

- $b_r = 1$ and $j_r = 0$ if r refers to the environment or to a manipulated object.
- The index 0 in the set $\{0, \dots, b_r - 1\}$ refers to the root body of the kinematic tree representing the robot r.

The configuration q of the system is an element of $\mathscr{C} = \prod_{r=1}^{N} \mathscr{C}_r$, the Cartesian product of the configuration spaces of the individual robots of the system. Hence

$$q = (q_1, \dots, q_N)\,, \tag{2.1}$$

where, for $r \in \{1, \dots, N\}$,

- $q_r = (x_r, y_r, z_r,\ \alpha_r, \beta_r, \gamma_r, \delta_r,\ \theta_{r,1}, \dots, \theta_{r,j_r})$, if r refers to a free-base articulated mechanism such as a humanoid robot for example, the first seven components representing the 3D position and the unit quaternion-parametrized orientation of its root body indexed by 0.
- $q_r = (x_r, y_r, z_r,\ \alpha_r, \beta_r, \gamma_r, \delta_r)$, if r refers to a rigid non-articulated manipulated object.
- $q_r = (\theta_{r,1}, \dots, \theta_{r,j_r})$, if r refers to a fixed-base manipulator such as the finger of a multi-fingered dexterous hand for example.
- q_r is not defined for $r = 0$ (the environment). It could be if we were considering deformable environment for example.

On each robot $r \in \{0, \dots, N\}$ we further specify a set of m_r planar surface patches indexed in $\{1, \dots, m_r\}$. A pair $(r, s) \in \{0, \dots, N\} \times \{1, \dots, m_r\}$, which characterizes the surface, refers to an element $(b'_{r,s}, T_{r,s})$ of $\{0, \dots, b_r - 1\} \times SE(3)$, where $b'_{r,s}$ denotes the body to which the surface (r, s) is rigidly attached and $T_{r,s} = (o_{r,s}, \vec{x}_{r,s}, \vec{y}_{r,s}, \vec{z}_{r,s})$ denotes a frame attached to the body $b'_{r,s}$, such that the point $o_{r,s}$ belongs to the surface, the vector $\vec{z}_{r,s}$ is the inwards normal to the surface, and the vectors $\vec{x}_{r,s}, \vec{y}_{r,s}$ are tangential to the

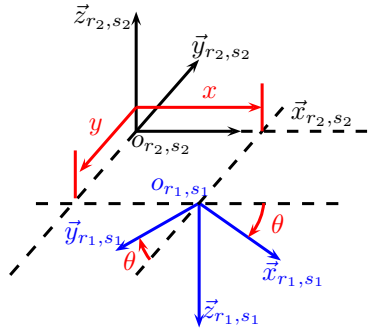


FIGURE 2.2. Geometric illustration of a contact $(r_1, s_1, r_2, s_2,\ x, y, \theta)$.

surface. More general (i.e. non-planar) surface patches can be handled by considering *normalized Gauss frames* [**Mon88**].

A contact is given by the specification of the two surfaces in contact (r_1, s_1) and (r_2, s_2) (i.e. a 4-tuple (r_1, s_1, r_2, s_2)) as well as their relative position/orientation (x, y, θ). More precision is found in the following definition:

DEFINITION 2.2.1 (contact, set of contacts E). *A* contact *is a 7-tuple* $(r_1, s_1, r_2, s_2,\ x, y, \theta)$, *such that* $r_1 \in \{1, \dots, N\}$, $r_2 \in \{0, \dots, N\}$, $r_2 \leq r_1$, $s_1 \in \{1, \dots, m_{r_1}\}$, $s_2 \in \{1, \dots, m_{r_2}\}$, $s_2 < s_1$ *if* $r_1 = r_2$, $b'_{r_1,s_1} \neq b'_{r_2,s_2}$ *if* $r_1 = r_2$, *and* $(x, y, \theta) \in \mathbb{R}^2 \times \mathbb{S}^1$. *We define the* set of contacts *E as the subset of* $\mathbb{N}^4 \times \mathbb{R}^2 \times \mathbb{S}^1$ *made of such 7-tuples.*

REMARK 2.2.2. *We can notice that this very generic definition only excludes environment-environment contacts* ($r_1 \neq 0$), *all other contact situations are possible, including a contact between two different bodies of the same robot (case* $r_1 = r_2$). *The ordering imposed on* (r_1, r_2) *and on* (s_1, s_2) *if* $r_1 = r_2$ *is required to avoid representing twice the same contact situation.*

A contact $(r_1, s_1, r_2, s_2,\ x, y, \theta)$ geometrically corresponds to setting

$$\vec{z}_{r_1,s_1}(q) = -\vec{z}_{r_2,s_2}(q)\,, \tag{2.2}$$

$$\vec{x}_{r_1,s_1}(q) = \cos(\theta)\vec{x}_{r_2,s_2}(q) + \sin(\theta)\vec{y}_{r_2,s_2}(q)\,, \tag{2.3}$$

$$\vec{y}_{r_1,s_1}(q) = \sin(\theta)\vec{x}_{r_2,s_2}(q) - \cos(\theta)\vec{y}_{r_2,s_2}(q)\,, \tag{2.4}$$

$$o_{r_1,s_1}(q) = o_{r_2,s_2}(q) + x\,\vec{x}_{r_2,s_2}(q) + y\,\vec{y}_{r_2,s_2}(q)\,. \tag{2.5}$$

We call these equations the *contact equations*. They are illustrated in Fig. 2.2. Once again, for simplicity these are restricted to the planar surfaces case; for surfaces modeled as manifolds, the more general contact equations [**Mon88**] should be considered (see Section 2.4 for our practical handling of non-planar surfaces).

On $\mathbb{N}^4 \times \mathbb{R}^2 \times \mathbb{S}^1$ we consider the projection map $p_{\mathbb{N}^4} : (r_1, s_1, r_2, s_2,\ x, y, \theta) \mapsto (r_1, s_1, r_2, s_2)$ which keeps only the first 4 components of the 7-tuple. $p_{\mathbb{N}^4}$ maps a contact to the pair of surfaces that constitute that contact.

DEFINITION 2.2.3 (stance, set of stances Σ). *A* stance σ *is a subset of the set of contacts* E *such that every pair of surfaces appears at most once. The set of all stances is denoted* Σ,

$$\Sigma = \left\{\sigma \in \mathscr{P}(E) \mid \forall c_1, c_2 \in \sigma : c_1 \neq c_2 \Rightarrow p_{\mathbb{N}^4}(c_1) \neq p_{\mathbb{N}^4}(c_2)\right\}. \tag{2.6}$$

REMARK 2.2.4. *A stance* σ *is necessarily a finite subset of* E*, given that*

$$\text{card}(\sigma) \leq \text{card}(p_{\mathbb{N}^4}(E)) \leq N\,(N+1)\,(\max_{r\in\{0,\dots,N\}} m_r)^2\,. \tag{2.7}$$

Every configuration of the system of robots defines a unique stance made of all the contacts for the robots in that configuration. Let us then denote $p_\Sigma : \mathscr{C} \to \Sigma$ the "forward kinematics" mapping that maps every configuration q to its stance σ. Inversely, each stance σ defines an "inverse kinematics" submanifold $\mathscr{Q}_\sigma$ of the configuration space containing all the configurations that satisfy the contact equations (2.2), (2.3), (2.4), and (2.5) for all the contacts in the stance,

$$\mathscr{Q}_\sigma = p_\Sigma^{-1}(\{\sigma\})\,. \tag{2.8}$$

On this submanifold we isolate a special subspace of same dimensionality but less volume $\mathscr{F}_\sigma$ in which the configurations are physically valid static configurations (i.e. configurations that are in static equilibrium, collision-free, for which the joint angles and torques are within their prescribed bounds).

The planning we will perform will be made on the set of stances Σ, rather than on the configuration space $\mathscr{C}$ as it is the case in usual motion planning. We thus need to define an adjacency relation between stances. Two stances will be considered adjacent if they differ by exactly one contact. To formalize this we define the binary relation "have one contact less than", that we denote $\sqsubset$, as

$$\sigma_1 \sqsubset \sigma_2 \quad \text{if} \quad \sigma_1 \subset \sigma_2 \text{ and } \text{card}(\sigma_2) = \text{card}(\sigma_1) + 1\,. \tag{2.9}$$

DEFINITION 2.2.5 (adjacency). *Two stances* σ_1 *and* σ_2 *are said to be* adjacent *if* $\sigma_1 \sqsubset \sigma_2$ *or* $\sigma_2 \sqsubset \sigma_1$*. Given a stance* σ *we define the three following* adjacency sets*:* $\text{Adj}^+(\sigma)$ *the set of stances that add one contact to* σ*,* $\text{Adj}^-(\sigma)$ *the set of stances that remove one contact from* σ*, and* $\text{Adj}(\sigma)$ *the set of stances that are adjacent to* σ *(add or remove one contact). Formally:*

$$\text{Adj}^+(\sigma) = \{\sigma' \in \Sigma \mid \sigma \sqsubset \sigma'\}\,, \tag{2.10}$$

$$\text{Adj}^-(\sigma) = \{\sigma' \in \Sigma \mid \sigma' \sqsubset \sigma\}\,, \tag{2.11}$$

$$\text{Adj}(\sigma) = \text{Adj}^+(\sigma) \cup \text{Adj}^-(\sigma)\,. \tag{2.12}$$

A step in the plan will be a transition from one stance to an adjacent stance. Such a step will be feasible if there exists a common transition configuration that realizes both stances at the same time, i.e. if the intersection of the respective feasible spaces of the two stances is non-empty. This motivates the following definition:

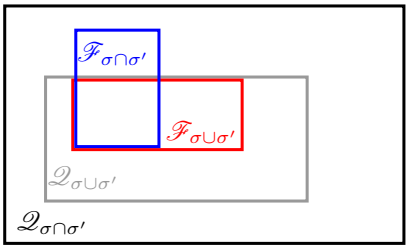


FIGURE 2.3. Venn diagrams illustrating Proposition 2.2.9.

DEFINITION 2.2.6 (feasible sequence of stances). *A sequence of stances* $(\sigma_1, \dots, \sigma_n) \in \Sigma^n$, $n \geq 2$, *is said to be* feasible *if it is made of a succession of adjacent stances with common transition configurations between two successive stances*

$$\forall\, i \in \{1, \dots, n-1\} \quad \sigma_{i+1} \in \mathrm{Adj}(\sigma_i) \text{ and } \mathscr{F}_{\sigma_i} \cap \mathscr{F}_{\sigma_{i+1}} \neq \varnothing\,. \tag{2.13}$$

We can now formulate the problem we want to solve:

PROBLEM 2.2.7 (non-gaited stances planning problem). *Given two stances* σ_{start} *and* σ_{goal} *in* Σ, *find a feasible sequence of stances* $(\sigma_1, \dots, \sigma_n)$ *such that* $\sigma_1 = \sigma_{\text{start}}$ *and* $\sigma_n = \sigma_{\text{goal}}$.

The ability to solve Problem 2.2.7 rather than cyclic gaited steps planning problems makes the robots more autonomous in handling unexpectedly structured environment. Note, however, that in many simple cases, gaited sequences emerge automatically ("naturally") in our results from solving Problem 2.2.7 (cf. Section 2.4).

REMARK 2.2.8. *We can also specify the goal to reach in terms of a configuration* q_{goal} *rather than a stance* σ_{goal}. *In this case we get the same formulation as Problem 2.2.7 where* σ_{goal} *simply denotes* $p_\Sigma(q_{\text{goal}})$. *These are actually the kind of queries we are addressing in Section 2.4.*

Solving Problem 2.2.7 in a greedy algorithmic way amounts to exploring $\mathrm{Adj}(\sigma)$ for a given σ, choosing $\sigma' \in \mathrm{Adj}(\sigma)$, finding a configuration q in $\mathscr{F}_\sigma \cap \mathscr{F}_{\sigma'}$ to validate the choice of σ', and iterating on σ'. Let us then analyse more closely the structure of $\mathrm{Adj}(\sigma)$ for a given $\sigma \in \Sigma$. First, we should rewrite constructive expressions of the adjacency sets. From Definition 2.2.5 it follows that

$$\mathrm{Adj}^+(\sigma) \;=\; \Big\{\sigma \cup \{c\} \mid c \in p_{\mathbb{N}^4}^{-1}\Big(p_{\mathbb{N}^4}(E) \setminus p_{\mathbb{N}^4}(\sigma)\Big)\Big\}\,, \tag{2.14}$$

$$\mathrm{Adj}^-(\sigma) \;=\; \Big\{\sigma \setminus \{c\} \mid c \in \sigma\Big\}\,. \tag{2.15}$$

The removing-one-contact set $\mathrm{Adj}^-(\sigma)$ is thereby a finite set, with $\mathrm{card}\big(\mathrm{Adj}^-(\sigma)\big) = \mathrm{card}(\sigma)$. The adding-one-contact set $\mathrm{Adj}^+(\sigma)$, however, needs to be more finely structured. When adding a contact $(r_1, s_1, r_2, s_2, x, y, \theta)$, we first choose the two surfaces (r_1, s_1) and (r_2, s_2) that we want to add as a contact, then we decide what their relative

position/orientation (x, y, θ) will be. A nice way to formalize this is through equivalence classes. Let us define on $\mathrm{Adj}^+(\sigma)$ the following equivalence relation

$$\sigma_1 \sim_\sigma \sigma_2 \text{ if } \sigma_1 = \sigma \cup \{c_1\} \text{ and } \sigma_2 = \sigma \cup \{c_2\} \text{ and } p_{\mathbb{N}^4}(c_1) = p_{\mathbb{N}^4}(c_2)\,. \tag{2.16}$$

This equivalence relation only makes distinction between the surface pairs in the added contacts with no consideration for the positions (x, y, θ). The quotient set $\mathrm{Adj}^+(\sigma)_{/\sim_\sigma}$, containing all the possible surface pairs that we can add to the stance, is in canonical bijection with $p_{\mathbb{N}^4}(E) \setminus p_{\mathbb{N}^4}(\sigma)$, i.e. the set of surface pairs that are not already present in the stance. So for each 4-tuple $(r_1, s_1, r_2, s_2) \in p_{\mathbb{N}^4}(E) \setminus p_{\mathbb{N}^4}(\sigma)$ we denote $\mathrm{cl}_\sigma(r_1, s_1, r_2, s_2)$ the corresponding equivalence class, which contains all the possible positions (x, y, θ) when we want to add the surface pair (r_1, s_1, r_2, s_2) as a contact (this equivalence class is thus homeomorphic to $\mathbb{R}^2 \times \mathbb{S}^1$)

$$\mathrm{cl}_\sigma(r_1, s_1, r_2, s_2) = \Big\{ \sigma \cup \big\{(r_1, s_1, r_2, s_2, x, y, \theta)\big\} \mid (x, y, \theta) \in \mathbb{R}^2 \times \mathbb{S}^1 \Big\}\,. \tag{2.17}$$

We now have all the ingredients to write an algorithm that tries to solve Problem 2.2.7: exploring $\mathrm{Adj}^-(\sigma)$ is straightforward; for $\mathrm{Adj}^+(\sigma)$, the algorithm explores every equivalence class from $\mathrm{Adj}^+(\sigma)_{/\sim_\sigma}$ by solving an optimization problem on (x, y, θ).

Before concluding this section, we will state a last useful property related to feasible transitions between two adjacent stances. For two adjacent stances σ and σ', a configuration in $\mathscr{F}_\sigma \cap \mathscr{F}_{\sigma'}$ is a configuration that realizes the geometric closure condition for the larger stance of the two ($\mathscr{Q}_{\sigma\cup\sigma'}$) and the feasibility condition for the smaller stance of the two ($\mathscr{F}_{\sigma\cap\sigma'}$). We can formalize this through the following property, illustrated in Fig. 2.3:

Proposition 2.2.9. *Let $\sigma \in \Sigma$ and $\sigma' \in \mathrm{Adj}(\sigma)$. Then we have*

$$\mathscr{F}_\sigma \cap \mathscr{F}_{\sigma'} = \mathscr{Q}_{\sigma\cup\sigma'} \cap \mathscr{F}_{\sigma\cap\sigma'}\,. \tag{2.18}$$

Proof. $\mathscr{F}_\sigma \cap \mathscr{F}_{\sigma'} \subset \mathscr{Q}_{\sigma\cup\sigma'} \cap \mathscr{F}_{\sigma\cap\sigma'}$ is trivial. Inversely, let $q \in \mathscr{Q}_{\sigma\cup\sigma'} \cap \mathscr{F}_{\sigma\cap\sigma'}$. This implies that q belongs to $\mathscr{Q}_{\sigma\cup\sigma'}$ and is a physically valid static configuration, hence $q \in \mathscr{F}_{\sigma\cup\sigma'}$ and subsequently $q \in \mathscr{F}_{\sigma\cap\sigma'} \cap \mathscr{F}_{\sigma\cup\sigma'} = \mathscr{F}_\sigma \cap \mathscr{F}_{\sigma'}$. □

Corollary 2.2.10. *Let $\sigma \in \Sigma$ and $\mathrm{cl}_\sigma(r_1, s_1, r_2, s_2) \in \mathrm{Adj}^+(\sigma)_{/\sim_\sigma}$. Then we have*

$$\mathscr{F}_\sigma \cap \Big(\bigcup_{(x,y,\theta)} \mathscr{F}_{\sigma\cup\{(r_1,s_1,r_2,s_2,x,y,\theta)\}} \Big) = \mathscr{F}_\sigma \cap \Big(\bigcup_{(x,y,\theta)} \mathscr{Q}_{\sigma\cup\{(r_1,s_1,r_2,s_2,x,y,\theta)\}} \Big)\,. \tag{2.19}$$

The role of Proposition 2.2.9 is to release redundant constraints in Definition 2.2.6, while Corollary 2.2.10 will prove useful later in the course of this chapter (Section 2.3.2).

Remark 2.2.11. *In some works* [**ALS95, SSEKP07, SLCS04**] *a path through $\mathscr{F}_\sigma$ from $q \in \mathscr{F}_\sigma$ to $q' \in \mathscr{F}_\sigma \cap \mathscr{Q}_{\sigma'}$ for $\sigma' \in \mathrm{Adj}^+(\sigma)$ would be called a* transit path*, and a path through $\mathscr{F}_\sigma$ from $q \in \mathscr{F}_\sigma$ to $q' \in \mathscr{Q}_\sigma \cap \mathscr{F}_{\sigma'}$ for $\sigma' \in \mathrm{Adj}^-(\sigma)$ is called a* transfer path *(cf. Fig. 2.4).*

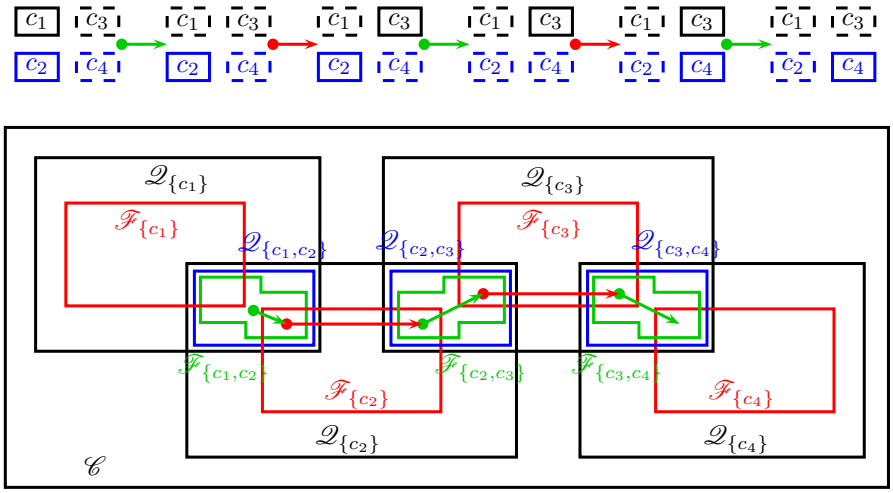


FIGURE 2.4. Transfer and transit paths for a biped feasible sequence of stances. In green transfer paths, in red transit paths. The top figure represents the footprints in Σ (right foot in blue, left foot in black), the bottom figure is a representation in $\mathscr{C}$ (for clarity $\mathscr{Q}_{\{c_1\}} \cap \mathscr{Q}_{\{c_4\}}$ is not represented).

2.3. Algorithm

Our objective now is to solve Problem 2.2.7 formulated in Section 2.2.

2.3.1. The Discrete Approach. In this section we discuss the approach proposed in the works of Hauser et al. and see how it fits in our generalized formalism for multiple agents. This approach is based on prior discretization of E. Let E_{finite} be a finite subset of E containing the start and goal stances,

$$(\sigma_{\text{start}} \cup \sigma_{\text{goal}}) \subset E_{\text{finite}} \subset E, \ \text{card}(E_{\text{finite}}) < \infty \,. \tag{2.20}$$

Let Σ_{finite} be the restriction of Σ to E_{finite},

$$\Sigma_{\text{finite}} = \{\sigma \in \Sigma \mid \sigma \subset E_{\text{finite}}\} \,. \tag{2.21}$$

Σ_{finite} is a finite set endowed with a finite undirected graph structure defined by the adjacency relation, as can be seen through the following constructions ("Trans" stands for *transitions* [**Hau08**])

$$\text{Adj}_{\text{finite}}(\sigma) = \text{Adj}(\sigma) \cap \Sigma_{\text{finite}} \,, \tag{2.22}$$

$$\text{Trans}(\sigma) = \{\sigma\} \times \text{Adj}_{\text{finite}}(\sigma) \,, \tag{2.23}$$

$$\mathscr{G} = \bigcup_{\sigma \in \Sigma_{\text{finite}}} \text{Trans}(\sigma) \tag{2.24}$$

$$= \{(\sigma_1, \sigma_2) \in \Sigma_{\text{finite}}^2 \mid \sigma_1 \sqsubset \sigma_2 \text{ or } \sigma_2 \sqsubset \sigma_1\} \,. \tag{2.25}$$

These constructions give us the finite graph structure that we wanted $(\Sigma_{\text{finite}}, \mathscr{G})$.

Hauser's algorithm explores this graph by growing a connected sub-graph $(\mathscr{V}, \mathscr{E})$, $\mathscr{V} \subset \Sigma_{\text{finite}}$, $\mathscr{E} \subset \mathscr{G}$, and maintaining a priority queue $Q \subset \mathscr{G} \times \mathbb{R}$. Let $f : \Sigma_{\text{finite}} \to \mathbb{R}$ be a cost function on the stances, this cost function is based on different heuristics such as the distance to goal, the distance to reference configurations, and the robustness of the static equilibrium. Algorithm 1 gives the outline of the planner (the *expansion* phase of

the multi-modal planner [**Hau08**]). $p_{\mathscr{G}} : \mathscr{G} \times \mathbb{R} \to \mathscr{G}$ denotes the canonical projection on $\mathscr{G}$.

Algorithm 1 FIND_SEQUENCE_OF_STANCES($\sigma_{\text{start}}, \sigma_{\text{goal}}$)

1: $\mathscr{V} \leftarrow \{\sigma_{\text{start}}\}$
2: $\mathscr{E} \leftarrow \varnothing$
3: $Q \leftarrow \varnothing$
4: **for all** $(\sigma_{\text{start}}, \sigma') \in \text{Trans}(\sigma_{\text{start}})$ **do**
5: $\quad Q \leftarrow Q \cup \{(\sigma_{\text{start}}, \sigma', f(\sigma'))\}$
6: **end for**
7: **repeat**
8: $\quad (\sigma_{\text{current}}, \sigma_{\text{adjacent}}, c) \leftarrow Q$.POP_LOWEST_COST()
9: $\quad q_{\text{adjacent}} \leftarrow$ SAMPLE_RANDOM($\mathscr{F}_{\sigma_{\text{current}} \cap \sigma_{\text{adjacent}}} \cap \mathscr{Q}_{\sigma_{\text{current}} \cup \sigma_{\text{adjacent}}}$)
10: $\quad$ **if** $q_{\text{adjacent}} \neq$ NULL **then**
11: $\quad\quad \mathscr{V} \leftarrow \mathscr{V} \cup \{\sigma_{\text{adjacent}}\}$
12: $\quad\quad \mathscr{E} \leftarrow \mathscr{E} \cup \{(\sigma_{\text{current}}, \sigma_{\text{adjacent}})\}$
13: $\quad\quad$ **for all** $(\sigma_{\text{adjacent}}, \sigma') \in \text{Trans}(\sigma_{\text{adjacent}}) \setminus p_{\mathscr{G}}(Q)$ **do**
14: $\quad\quad\quad Q \leftarrow Q \cup \{(\sigma_{\text{adjacent}}, \sigma', f(\sigma'))\}$
15: $\quad\quad$ **end for**
16: $\quad$ **else**
17: $\quad\quad Q \leftarrow Q \cup \{(\sigma_{\text{current}}, \sigma_{\text{adjacent}}, c + \text{COST\_INCREMENT})\}$
18: $\quad$ **end if**
19: **until** $\sigma_{\text{goal}} \in \mathscr{V}$ or c.IS_OUT_OF_RANGE()

Starting from σ_{start} the algorithm enqueues all the discretized stances that are adjacent to σ_{start} (lines 1 to 5), indifferently adding or removing a contact since they are all in finite number. Then it enters the main search loop (lines 7 to 19): first dequeuing the "most promising" pair of stances made of the currently explored stance along with one of its adjacent stances (line 8), and tries to sample a feasible transition configuration using Proposition 2.2.9 (line 9). In case of success (lines 10 to 15), the adjacent stance is added to the exploration graph (lines 11 and 12) and all the transitions from this adjacent stances (i.e. the stances that are adjacent to the adjacent stance) that are not already present in the queue are enqueued for future exploration (lines 13 to 15). In case of failure to sample a transition configuration, the considered pair is penalised by augmenting its cost and re-enqueued into Q (lines 16 and 17). As the exploration progresses and no solution is found, more time will be allocated to sampling reluctant transitions.

In the worst case, this algorithm will end up exploring all the stances in the connected component of $(\Sigma_{\text{finite}}, \mathscr{G})$ containing σ_{start}. However, if no solution is found then this does not necessarily mean that Problem 2.2.7 does not have a solution, but it could also be due to the fact that the discretization performed by choosing E_{finite} might not have been fine enough. This problem is not encountered in our proposed algorithm that we detail hereunder.

2.3.2. The Continuous Approach. In this approach we do not need to discretize Σ. We grow a tree $(\mathscr{V}, \mathscr{E})$, $\mathscr{V} \subset \Sigma$, $\mathscr{E} \subset \Sigma^2$, and we maintain on it a priority queue $Q \subset \Sigma \times \mathbb{R}$. Let $f : \mathscr{C} \to \mathbb{R}$ be a cost function on the configuration space. Algorithm 2 gives the outline of the planner, where ε and δ are two positive parameters, and DISTANCE is a heuristic "metric" on Σ.

Algorithm 2 FIND_SEQUENCE_OF_STANCES($\sigma_{\text{start}}, \sigma_{\text{goal}}$)

1: $q_{\text{start}} \leftarrow$ FIND_BEST_CONFIG($\mathscr{F}_{\sigma_{\text{start}}}$)
2: $\mathscr{V} \leftarrow \{\sigma_{\text{start}}\}$
3: $\mathscr{E} \leftarrow \varnothing$
4: $Q \leftarrow \{(\sigma_{\text{start}}, f(q_{\text{start}}))\}$
5: **repeat**
6: $(\sigma_{\text{current}}, c) \leftarrow Q$.POP_LOWEST_COST_STANCE()
7: **for all** $\sigma_{\text{adjacent}} \in \text{Adj}^-(\sigma_{\text{current}})$ **do**
8: $q_{\text{adjacent}} \leftarrow$ FIND_BEST_CONFIG$\big(\mathscr{Q}_{\sigma_{\text{current}}} \cap \mathscr{F}_{\sigma_{\text{adjacent}}}\big)$
9: **if** $q_{\text{adjacent}} \neq$ NULL and DISTANCE($\sigma_{\text{adjacent}}, \mathscr{V}$) $> \varepsilon$ **then**
10: $\mathscr{V} \leftarrow \mathscr{V} \cup \{\sigma_{\text{adjacent}}\}$
11: $\mathscr{E} \leftarrow \mathscr{E} \cup \{(\sigma_{\text{current}}, \sigma_{\text{adjacent}})\}$
12: $Q \leftarrow Q \cup \{(\sigma_{\text{adjacent}}, f(q_{\text{adjacent}}))\}$
13: **end if**
14: **end for**
15: **for all** $\text{cl}_{\sigma_{\text{current}}}(r_1, s_1, r_2, s_2) \in \text{Adj}^+(\sigma_{\text{current}})_{/\sim_{\sigma_{\text{current}}}}$ **do**
16: $q_{\text{adjacent}} \leftarrow$ FIND_BEST_CONFIG$\big(\mathscr{F}_{\sigma_{\text{current}}} \cap (\bigcup_{(x,y,\theta)\in\mathbb{R}^2\times\mathbb{S}^1} \mathscr{Q}_{\sigma_{\text{current}}\cup\{(r_1,s_1,r_2,s_2,x,y,\theta)\}})\big)$
17: $\sigma_{\text{adjacent}} \leftarrow p_\Sigma(q_{\text{adjacent}})$
18: **if** $q_{\text{adjacent}} \neq$ NULL and DISTANCE($\sigma_{\text{adjacent}}, \mathscr{V}$) $> \varepsilon$ **then**
19: $\mathscr{V} \leftarrow \mathscr{V} \cup \{\sigma_{\text{adjacent}}\}$
20: $\mathscr{E} \leftarrow \mathscr{E} \cup \{(\sigma_{\text{current}}, \sigma_{\text{adjacent}})\}$
21: $Q \leftarrow Q \cup \{(\sigma_{\text{adjacent}}, f(q_{\text{adjacent}}))\}$
22: **end if**
23: **end for**
24: **until** DISTANCE($\sigma_{\text{goal}}, \mathscr{V}$) $< \delta$ or $Q = \varnothing$

First, the algorithm enqueues σ_{start} (lines 1 to 4). Then it enters the main search loop (lines 5 to 24), which consists once again in dequeuing the "most promising" stance (line 6), and exploring the adjacent stances. This exploration is split into two stages: the adjacent stances by removing a contact (lines 7 to 14) and the adjacent stances by adding a contact (lines 15 to 23). The former adjacent stances are in finite number and for each of them the algorithm tries to sample a feasible transition configuration (line 8). In case of success, the adjacent stance, if not already in the exploration graph, is added to this exploration graph and enqueued (lines 9 to 13). The latter adjacent stances are explored via their equivalence classes, meaning that the algorithm picks up a pair of surfaces not already in the currently explored stance (line 15), and for every such pair it tries to find a transition configuration while simultaneously looking for the best relative position for the pair of surfaces (line 16), upon sucess the pair of surfaces is completed as a contact with the found relative position and added to the current stance (line 17) to form the adjacent stance that will be enqueued and added the exploration graph if not already present (lines 18 to 22).

The main added value of Algorithm 2 with regard to Algorithm 1 lies in line 16. Indeed, both Algs. 1 and 2 rely on an *inverse stance solver* that returns configurations from 3 types of queries:

- type 1 queries are made on spaces of the form $\mathscr{F}_\sigma$,

- type 2 queries are made on spaces of the form $\mathscr{Q}_\sigma \cap \mathscr{F}_{\sigma'}$ where $\sigma' \in \mathrm{Adj}^-(\sigma)$ (cf. Proposition 2.2.9),
- type 3 queries are made on spaces of the form $\mathscr{F}_\sigma \cap (\bigcup_{(x,y,\theta)} \mathscr{Q}_{\sigma \cup \{(r_1,s_1,r_2,s_2,x,y,\theta)\}})$ where $\mathrm{cl}_\sigma(r_1, s_1, r_2, s_2) \in \mathrm{Adj}^+(\sigma)_{/\sim_\sigma}$ (cf. Corollary 2.2.10).

In Algorithm 1 this inverse stance solver is called through SAMPLE_RANDOM and is based the *Iterative Constraint Enforcement* method described in [**HBL05**]. In Algorithm 2 the solver is called through FIND_BEST_CONFIG. It is based on a "black-box" non-linear optimization solver, detailed in the next chapter (Chapter 3). While type 2 queries are answered by both solvers, processing type 3 queries is a specificity of our solver, which, for $\sigma \in \Sigma$ and $\mathrm{cl}_\sigma(r_1, s_1, r_2, s_2) \in \mathrm{Adj}^+(\sigma)_{/\sim_\sigma}$, solves the following optimization problem

$$\min_{q,\lambda,(x,y,\theta)} \mathrm{obj}(q,\lambda) \quad \text{subject to} \begin{cases} (x,y,\theta) = p_{\mathbb{R}^2\times\mathbb{S}^1}(q) \\ q \in \mathscr{F}_\sigma \\ q \in \mathscr{Q}_{\sigma\cup(r_1,s_1,r_2,s_2,x,y,\theta)}\,, \end{cases} \tag{2.26}$$

where $p_{\mathbb{R}^2\times\mathbb{S}^1} : \mathscr{C} \to \mathbb{R}^2\times\mathbb{S}^1$ is the "forward kinematics" mapping which inverts for (x, y, θ) the contact equations (2.3), (2.4), and (2.5). λ is a parametrization of the contact forces.

Let us now write a more detailed expression of $\mathrm{obj}(q, \lambda)$. The guide path that was computed through collision-free path planning (see Appendix A) is made of a number M of *milestone configurations* $q_{\mathrm{ref},1}, \ldots, q_{\mathrm{ref},M}$ (the nodes of the tree or graph structure that was constructed in the PRM [**KScLO96**] or RRT [**LK01**] process and from which the collision-free path is extracted). Such milestones carry information about major global change in the configuration of the system q along the guide path. When executing the search Algorithm 2, we maintain a global integer variable ι that is initialized to 1 and incremented by 1 whenever $q_{\mathrm{ref},\iota}$ is reached (ie. when the search tree $\mathcal{T}$ contains a configuration close enough to $q_{\mathrm{ref},\iota}$). The search succeeds and Algorithm 2 stops when reaching $q_{\mathrm{ref},M} = q_{\mathrm{goal}}$. The objective function of the inverse stance solver will be composed of four weighted components:

- a global configuration component

$$\mathrm{obj}_1(q,\lambda) = \|q - q_{\mathrm{ref},\iota}\|^2\,, \tag{2.27}$$

- a force minimization component

$$\mathrm{obj}_2(q,\lambda) = \|\lambda\|^2\,, \tag{2.28}$$

- a torque minimization component

$$\mathrm{obj}_3(q,\lambda) = \|\tau(q,\lambda)\|^2\,, \tag{2.29}$$

- a last component $\mathrm{obj}_4(q, \lambda)$ used when adding a contact and that will steer the position of the non-fixed contact to the position of the corresponding links in the currently-targeted milestone $q_{\mathrm{ref},\iota}$, this is the component which "pulls" the foot to advance forward when walking for example.

$$\mathrm{obj}_4(q,\lambda) = \|o_{r_1,s_1}(q) - o_{r_1,s_1}(q_{\mathrm{ref},\iota})\|^2 + \|o_{r_2,s_2}(q) - o_{r_2,s_2}(q_{\mathrm{ref},\iota})\|^2\,. \tag{2.30}$$

If we denote w_1, w_2, w_3, w_4 the respective weights then the objective function is simply

$$\mathrm{obj}(q,\lambda) = \begin{cases} \sum_{i=1}^4 w_i \,\mathrm{obj}_i(q,\lambda) & \text{if adding contact,} \\ \sum_{i=1}^3 w_i \,\mathrm{obj}_i(q,\lambda) & \text{if removing contact.} \end{cases} \tag{2.31}$$

Algorithm 2 is a best-first search algorithm. As such, it is a greedy algorithm that suffers from the local minima problem. To avoid this, many heuristics can be added to the algorithmic blueprint defined by Algorithm 2 [**Esc08, EKM06, EKMG08**]. However, anecdotally, such problems were not encountered in the runs of the planner that we made in the experiments of Section 2.4. Although completeness and global optimality issues are not tackled in our work, the analysis here being only qualitative, the proposed algorithm proved to be practically efficient in solving the queries of Section 2.4.

2.4. Results

In this section we show results obtained by applying the generic algorithm Algorithm 2 to different classes of problems, cf. Figs. 2.5, 2.6, 2.7, 2.8, and 2.9. In all these figures, for the computed solution sequence of stances $(\sigma_1, \ldots, \sigma_n) \in \Sigma^n$ of the considered problem, we display a sequence of configurations $(q'_1, \ldots, q'_n) \in \mathscr{C}^n$ such that $q'_1 \in \mathscr{F}_{\sigma_1}$ and $\forall\, i \in \{2, \ldots, n\}$ $q'_i \in \mathscr{F}_{\sigma_i} \cap \mathscr{F}_{\sigma_{i-1}}$. It is very important to emphasize here that the pictures are not snapshots of a continuous motion. They are not merely representative of the result, they are the result. So it is important to keep this in mind in order not to over-estimate the results presented here.

In these scenarios we used three robots models:

- a model of the HRP-2 humanoid robot [**KKK$^+$04**] appearing in Figs. 2.5, 2.7, 2.8, and 2.9,
- rigid objects: the ball of Fig. 2.6, the table of Fig. 2.7, and the box of Fig. 2.8,
- fixed-base manipulators: the four fingers of Fig. 2.6.

Surface patches on the robots have been chosen as follows:

- one surface per foot of the HRP-2 robot in all the scenarios, one surface per hand in Figs. 2.7, 2.8, and 2.9,
- one surface per planar piece of the ground in all the scenarios,
- one surface per face of the cube in Fig. 2.8,
- one surface per handle of the table in Fig. 2.7,
- one surface per monkey bar in Fig. 2.9,
- one surface per fingertip in Fig. 2.6,
- 20 regularly distributed planar surfaces tangent to the ball in Fig. 2.6. Every such plane approximates the spherical surface around the tangent point. Contacts yielded on this tangent planes are then projected back onto the spherical surface.

In the modeling of the feasible spaces $\mathscr{F}_\sigma$ we considered the following constraints (see Chapter 3):

- static equilibrium for all the underactuated free-base robots (including objects) considered as individual systems, under the action of external contact forces, gravity force, and actuation torques,
- Newton's third law for all the internal contact forces on the system of robots and objects considered as a whole,
- Coulomb friction model for the unilateral contact forces (all the forces except the ones listed in the next item),
- fixed grasp model for the bilateral contact forces: the contacts between the hands of the robots and the handles of the table in Fig. 2.7, and between the hands of the robot and the monkey bars in Fig. 2.9,

FIGURE 2.5. Biped locomotion over irregular terrain. Coulomb friction allows the robot not to slip. The friction coefficient is set to $\mu = 1$.

- joint angles limits for all the joints of the poly-articulated mechanisms (HRP-2 and the multi-fingered hand),
- bounds on the torques of all the actuators in HRP-2, except for the wrist actuators.

Note that, when applicable, the scenarios were chosen to demonstrate the performance of the planner in situations in which friction is specifically required to solve the problem, as highlighted by a relatively high coefficient of friction ($\mu = 1$). Such a high friction coefficient is required for example to cross the steepest part of the hill in Fig. 2.5 (as opposed to standing on horizontal planar surface in which low friction is enough), or to manipulate the box using only planar unilateral contact in Fig. 2.8 without resorting to bilateral grasps. Lower coefficient of friction would be sufficient for less constraining problems.

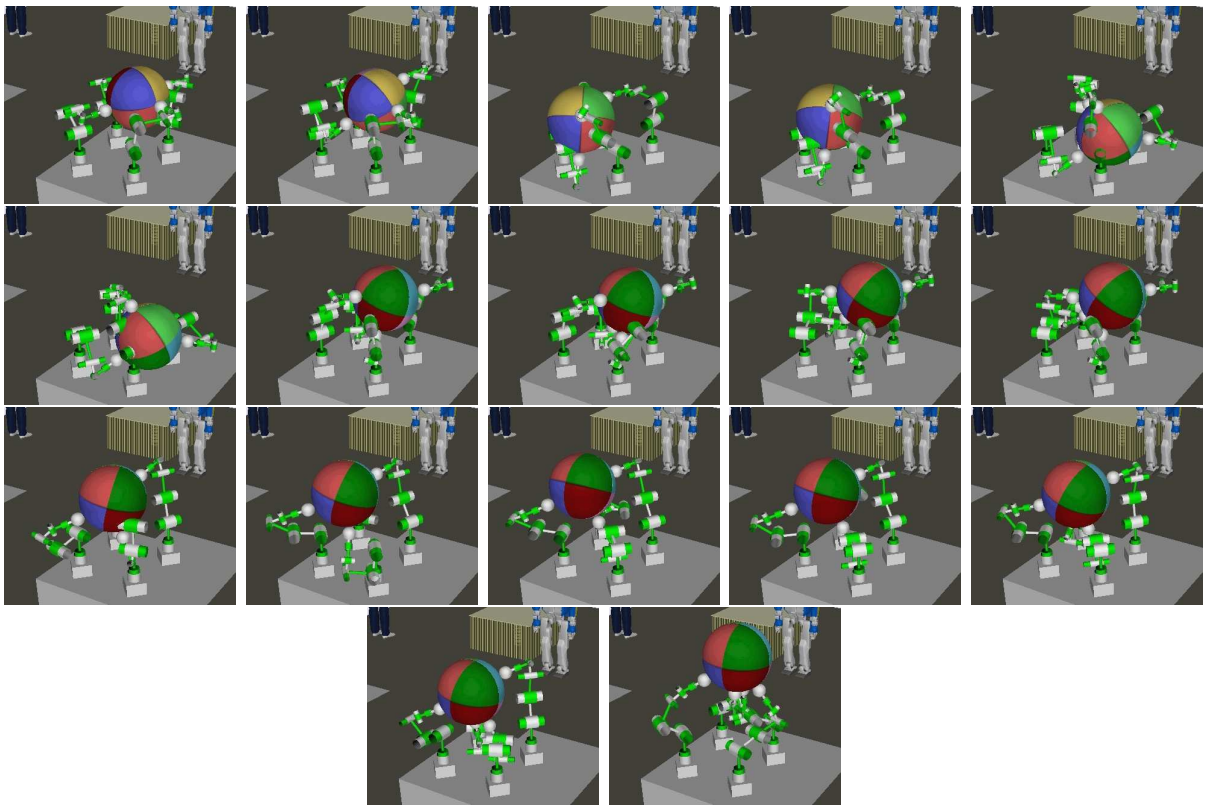


FIGURE 2.6. Dexterous manipulation. The objective is to rotate the 3 kg ball upside down. The fingers are 6-DOF elbow-like manipulators with wrist-like end-effectors. The friction coefficients between the end-effectors of the fingers and the ball are set to $\mu = 1$. No limits are considered on the torques delivered by the actuators in the fingers.

TABLE 2.1. Experimental results

	Fig. 2.5	Fig. 2.6	Fig. 2.7	Fig. 2.8	Fig. 2.9
N (robots)	1	5	3	2	1
dim($\mathscr{C}$)	46	30	98	52	46
Num. of steps	32	17	26	24	33
Size of the tree	51	846	47	144	91
Comp. time (s)	133	830	318	230	750

Tab. 2.1 gives some experimental figures concerning these scenarios made on a 3.06 GHz computer running under Windows XP. The program is compiled from a C++ implementation of the framework.

2.5. Conclusion

We wrote a multi-contact stances planning algorithm for multiple robots having to make use of contacts to perform locomotion or manipulation tasks. The autonomy of the robots is enhanced as little domain knowledge is required to plan an acyclic non-gaited sequence of stances. This autonomy is further increased by not specifying pre-discretized

FIGURE 2.7. Collaborative manipulation. Here we use an improved version of Algorithm 2 as contacts between the hands of the robots and the handles of the table are required not to be broken during the planning, as specified at problem-instantiation-time by the user.

candidate contact locations on the environment, the continuity of which is totally explored by the planner. Along with autonomy, the other key driving concept in this chapter was the generality. Our planner was not targeted for any specific model of robot or system of robots. The planner successfully performed on a set of problems taken from different sub-fields of motion planning in robotics, namely, the legged locomotion, dexterous manipulation, combined whole-body locomotion and manipulation, and collaborative manipulation problems. All these locomotion and manipulation problems were unified within the same framework.

The next step is to take the output of this algorithm as an input of a motion planning algorithm that would plan the continuous motion going through these stances. Although static criteria were considered in the stances planning stage, the continuous motion planner can use them, along with the generated configurations that correspond to each stance of the plan, as milestones to plan a dynamic trajectory. This is the purpose of Chapter 4.

Before that we will detail the inverse stance solver used to generate each single configuration (and eventually to complete the missing contact in the corresponding stance) in the next chapter (Chapter 3).

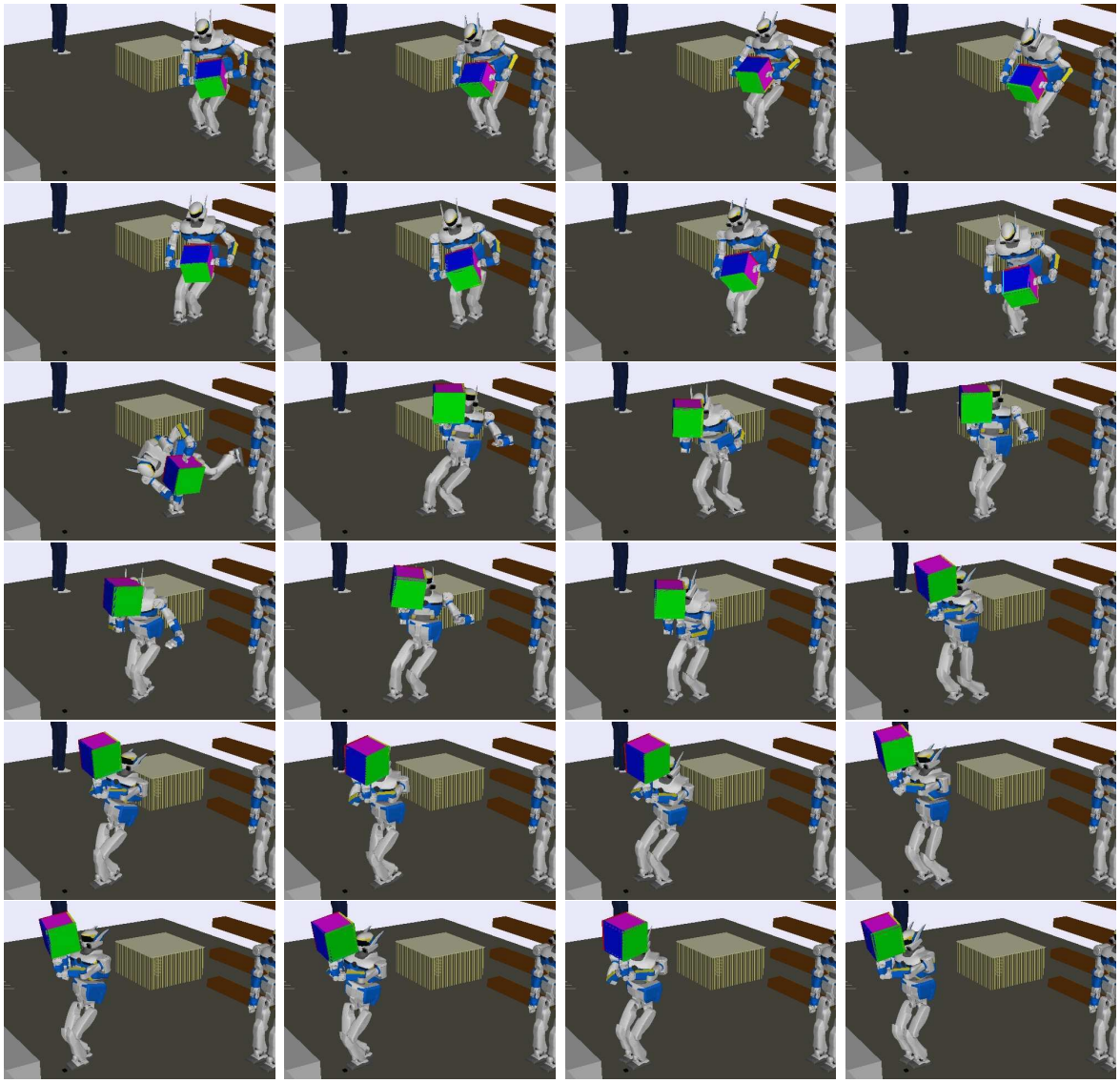


FIGURE 2.8. Combined whole-body manipulation and locomotion. The objective is for the HRP-2 robot to advance 2 m forward while simultaneously performing half rotation of the 5 kg box, bringing the purple face up. Friction coefficients between the hands and the box are set to $\mu = 1$.

2.6. Appendix: Additional Examples

Figs. 2.10 to 2.13 show some additional possible applications of the planner.

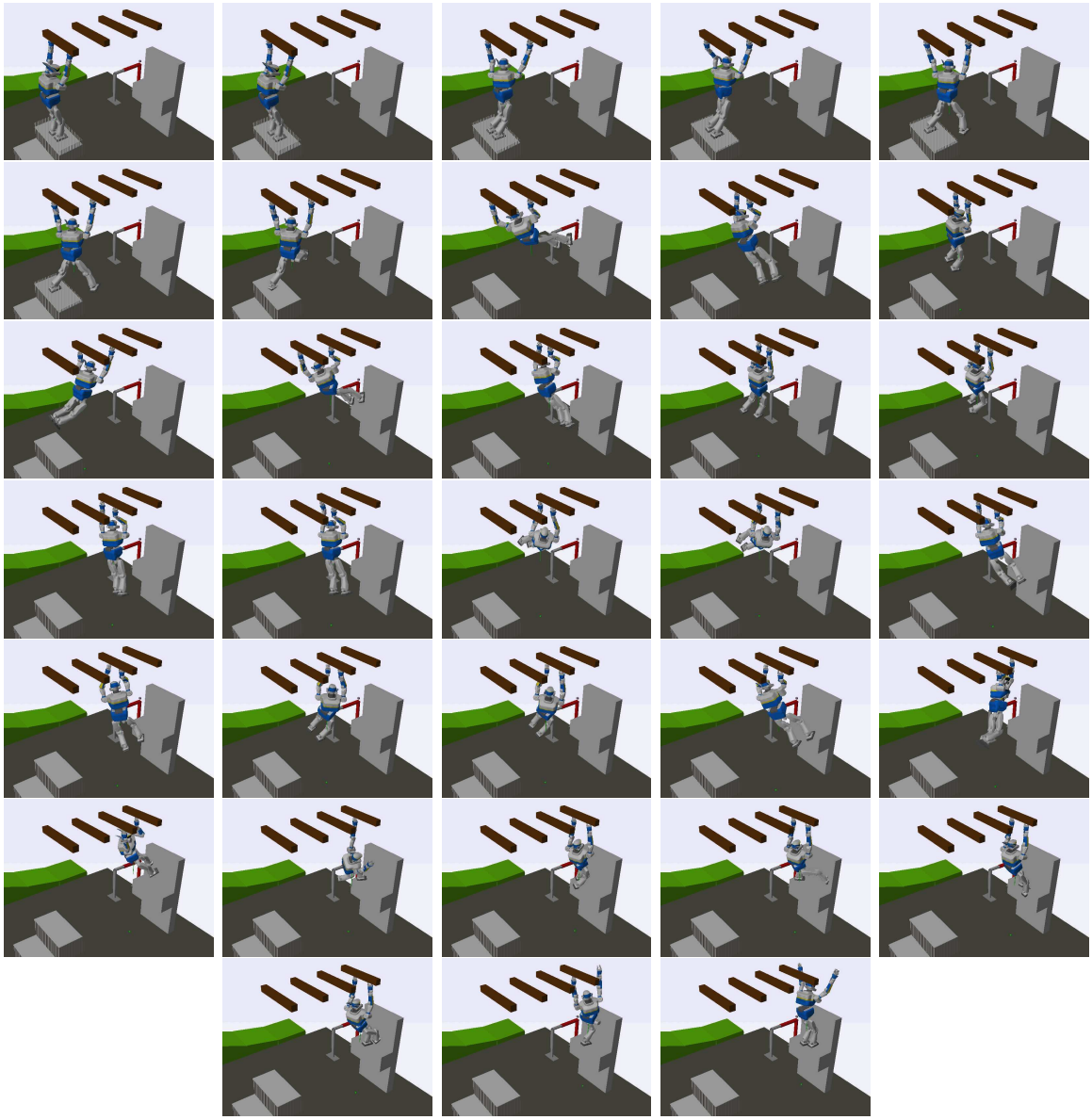


FIGURE 2.9. Bilateral contacts on monkey bars. This example illustrates the necessity of use of bilateral contacts to solve the planning problem.

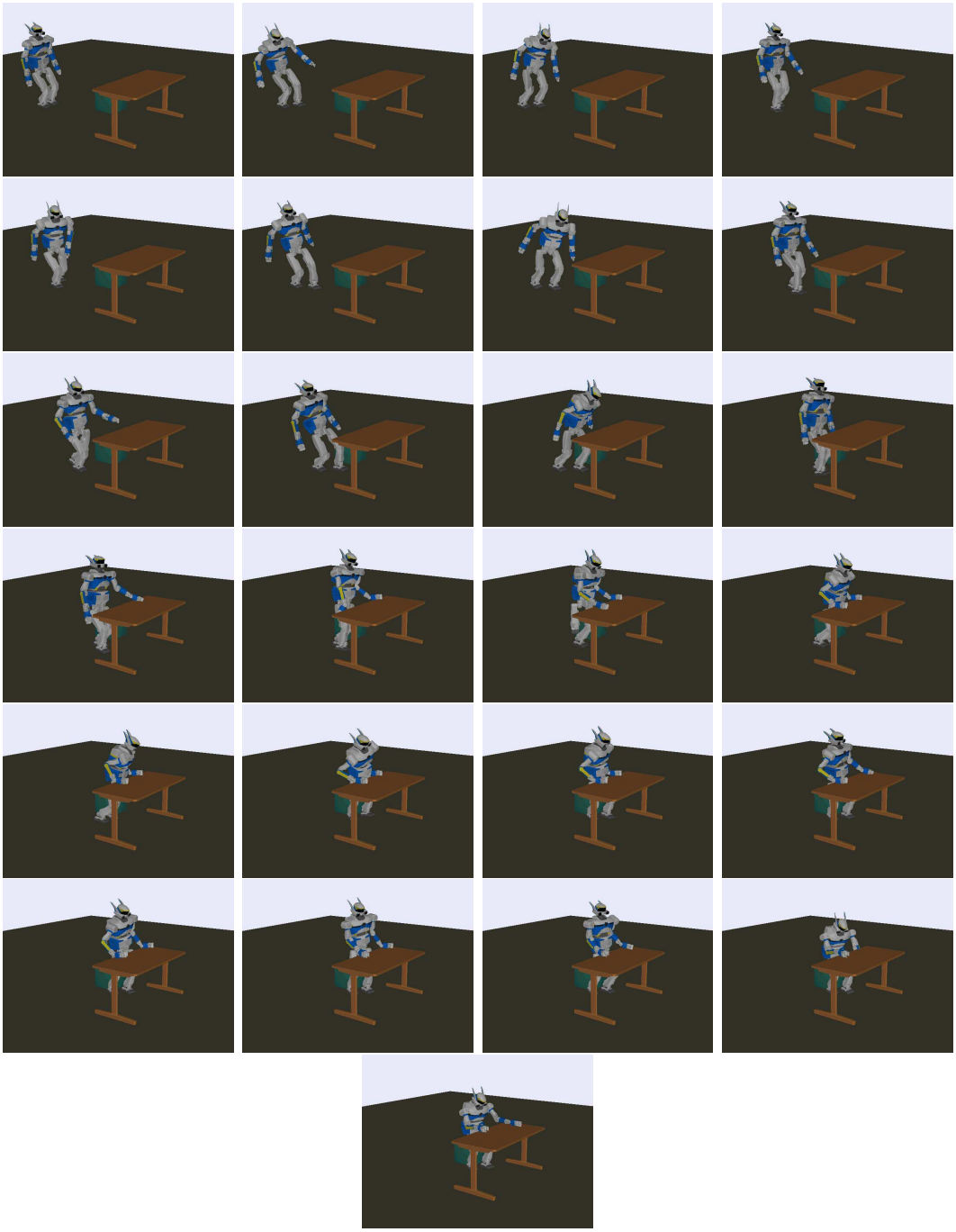


FIGURE 2.10. Locomotion. The objective is to sit down at the desk. The initial stance is the robot standing about 2m away from the desk. The final stance is the robot sitting with contacts of its tights with the chair, its feet with the floor, and its forearms with the desk. The motion comprises a first phase in which the robot bipedly walks towards the desk followed by a phase that makes the robot properly sit down.

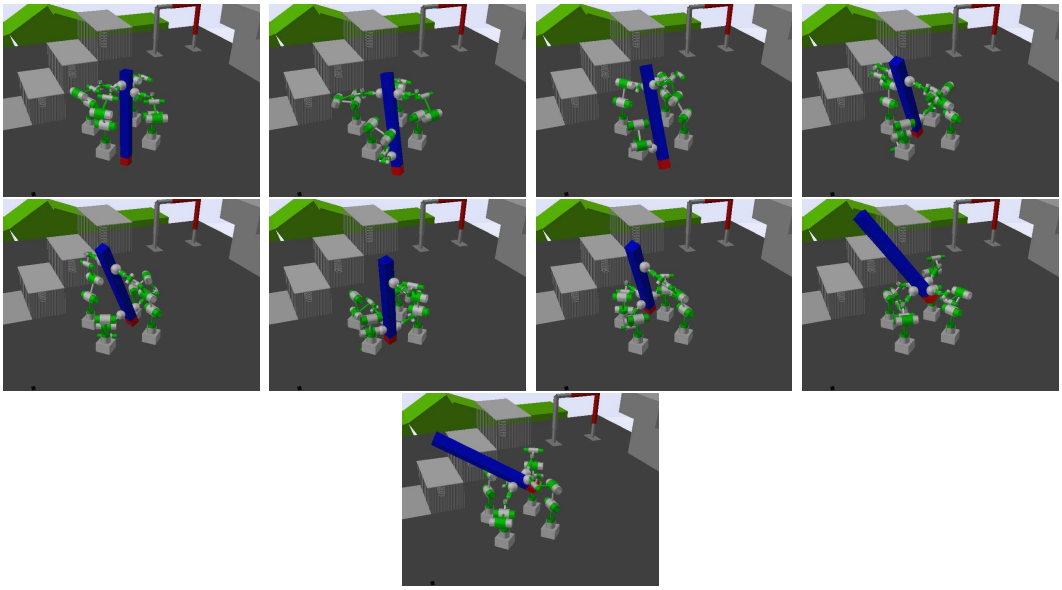


FIGURE 2.11. Dexterous manipulation. The objective is to hold the pen from the red end.

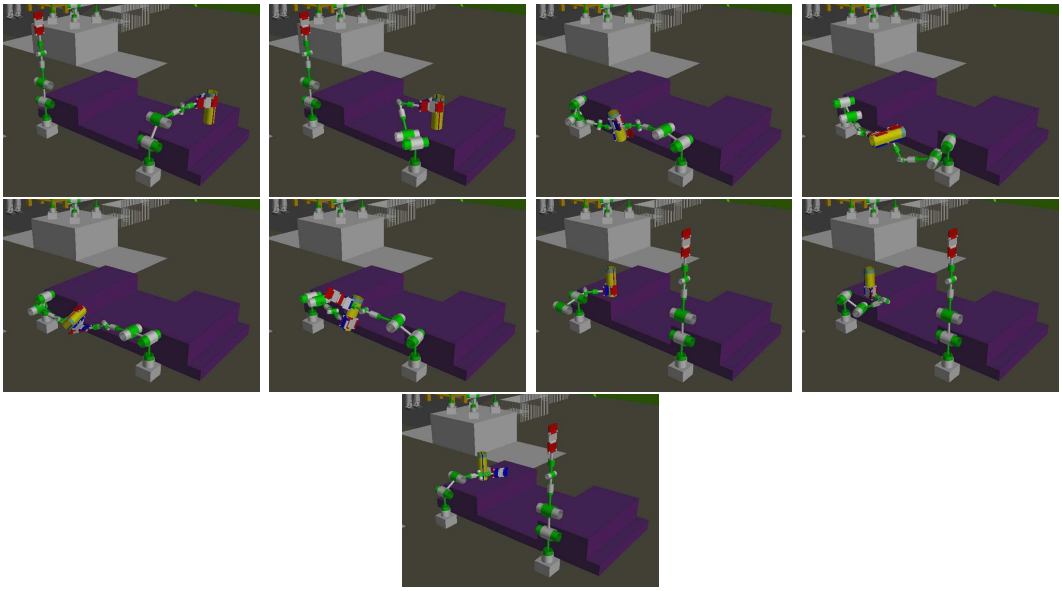


FIGURE 2.12. Dual arm manipulation. The objective is to bring the cylinder from the first platform to the second platform. The second platform is outside the workspace of the first arm. The planner finds a solution in which it needs to transfer the cylinder to the second arm. The initial stance is made of a contact between one of the plane surfaces of the cylinder and the first platform. The final stance is made of a contact between the same plane surface of the cylinder and the other platform. Along the sequence contacts are made and broken between the surfaces of the end-effectors of the grippers and the cylindrical surface of the cylinder

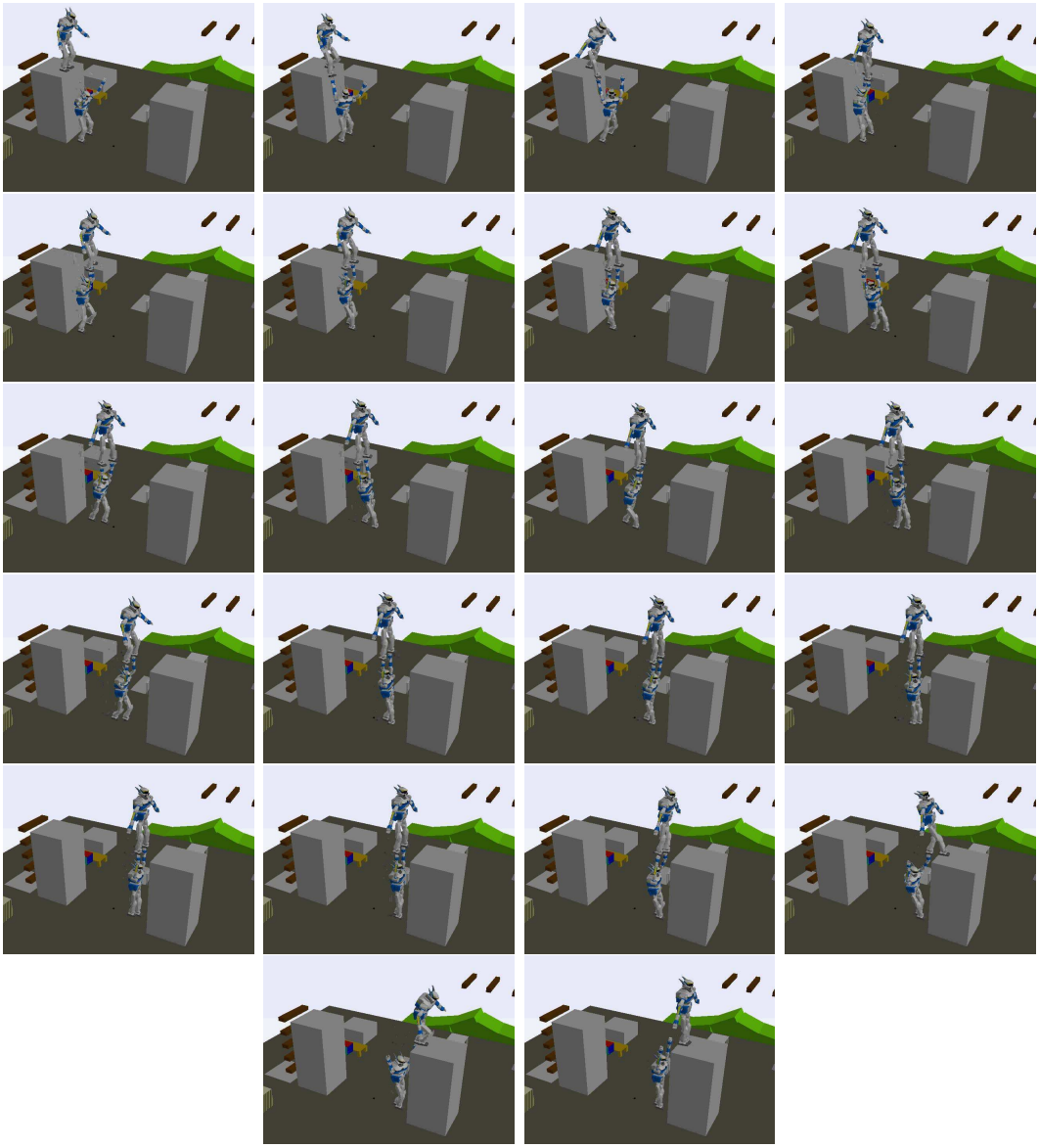


FIGURE 2.13. Toy scenario. The objective is for the robot to get from the first platform to the second platform. This cannot be achieved without using the other robot as a support. The planner finds such a solution. Contacts are created and broken between the hands of the supporting robot and the feet of the supported robot, in addition to contacts of the feet of the supported robot with the platforms and the feet of the supporting robot with the floor.

CHAPTER 3

Static Multi-Contact Inverse Problem for Multiple Humanoid Robots and Manipulated Objects

In this chapter, as required by the algorithm presented in the previous chapter (Chapter 2), we solve the static-equilibrium constrained inverse kinematics problem for a system made of multiple humanoid robots and manipulated objects given a set of contacts between any surfaces of the robots, any surfaces of the manipulated objects, and any surfaces of the environment. In particular, inter-robots contacts are possible. The contacts considered here are neither necessarily coplanar, nor necessarily horizontal, frictional, might be unilateral (support) or bilateral (grasp). We solve both the geometric variables (configurations) and the statics variables (contact forces) simultaneously within one optimization query. In the resulting configurations all the robots and the manipulated objects are in static equilibrium under the action of gravity and actuator torques that are constrained to stay within their bounds. The main focus of the chapter is on the formulation of the problem rather than the optimization algorithm, as we consider the latter as a black box that only requires a mathematical model providing algorithms to compute the values of the objective function, the constraints functions, and their derivatives. We apply this work to quasi-static multi-contact legged locomotion planning on irregular terrain, multi-fingered dexterous manipulation planning, and collaborative manipulation planning.

3.1. Introduction

Solving the static multi-contact inverse problem is a core issue in acyclic multi-contact motion planning. The algorithms introduced in the previous chapter (Chapter 2) explore the workspace environment by growing a stances tree; a stance being a set of contacts between surfaces of the robot's cover and surfaces of the environment. To validate a stance and add it to the exploration tree, the algorithms need to test the feasibility of the stance by finding a configuration of the robot that realizes the stance. This is what we call here the stance inverse problem. For a given stance σ, let us denote $\mathscr{Q}_\sigma$ the solution set of this inverse problem, i.e. the set of all configurations that geometrically realize the stance. $\mathscr{Q}_\sigma$ is a sub-manifold of the configuration space of strictly lower dimension. Let us denote $\mathscr{F}_\sigma$ the subset of $\mathscr{Q}_\sigma$ made of all the configurations that realize the stance while being in static equilibrium. $\mathscr{F}_\sigma$ is a closed subset of $\mathscr{Q}_\sigma$ provided with its subspace topology. For $q \in \mathscr{F}_\sigma$, let us denote $\Lambda_\sigma(q)$ the set of all admissible contact forces that maintain the configuration in static equilibrium. If the stance is made of n surface contacts, each surface $i \in \{1, \dots, n\}$ being modeled by a polygon with V_i vertices, then $\Lambda_\sigma(q)$ is a subset of $\mathbb{R}^{3\sum_i V_i}$.

The fundamental problem we would like to solve is to test the feasibility of a given stance σ, i.e. to test whether

$$\mathscr{F}_\sigma \neq \varnothing \ ? \tag{3.1}$$

Then if (3.1) is true, it would also be convenient to exhibit one solution, i.e. to solve

$$\text{find } q \in \mathscr{F}_\sigma \text{ and } \lambda \in \Lambda_\sigma(q). \tag{3.2}$$

Last, we would like a more refined version of (3.2), which is to minimize a criterion over all the feasible stances and associated forces, i.e. to solve the following non-linear constrained optimization problem

$$\begin{aligned} &\min_{(q,\lambda)} \text{obj}(q,\lambda) \\ &\text{subject to } \begin{cases} q \in \mathscr{F}_\sigma \\ \lambda \in \Lambda_\sigma(q). \end{cases} \end{aligned} \tag{3.3}$$

3.2. Related Work

A lot of effort has been dedicated to solving inverse geometric queries on closed kinematic chains in the field of randomized path planning, e.g. [**LYK99**][**HA00**][**CSL02**]. Using the notations introduced in the previous section, these works solve the query

$$\text{find a random } q \in \mathscr{Q}_\sigma \tag{3.4}$$

with no other constraints, thus without considering static equilibrium (fixed-base robots). Then, given a particular $q_0 \in \mathscr{Q}_\sigma$, for example as returned by solving the problem (3.4), works like [**BL08**][**RMBO08**] are concerned with testing the static equilibrium of q_0, i.e. solving the problem

$$q_0 \in \mathscr{F}_\sigma \ ? \tag{3.5}$$

If the answer to the problem (3.5) is true, other methods, e.g. [**BW07**], allow to compute optimal contact forces, and thus solve the following problem

$$\min_{\lambda \in \Lambda_\sigma(q_0)} \text{obj}(\lambda). \tag{3.6}$$

Sequentially solving problems (3.4) then (3.5) then (3.6) gives a rejection scheme for solving the problem (3.2). We propose here another scheme that does not rely on random configuration rejection sampling which might be costly especially in the case of stances made of low number of contacts where very few geometrically valid configurations are in static equilibrium. So we decide to solve problem (3.2) directly through the problem (3.3). Both [**EKM06**] and [**HBL$^+$08**] have chosen this approach. Our contributions with regard to these two works is in the modeling of the conditions that define $\mathscr{F}_\sigma$ as we try to remain as general as possible and avoid any strong hypotheses that could have allowed us to use approximations on the static equilibrium constraint, by reducing it for example to the belonging of the ground projection of the CoM to the support polygon. We also avoid hypotheses on the rigidity of the robots as we consider the specified limits on the actuators torques needed in holding the static configuration. The Iterative Constraint Enforcement method proposed in [**HBL$^+$08**] considers torques limits only in a post processing rejection test once the rigid version of the problem has been solved. Once again we want to avoid this rejection scheme and input the torques limits constraint directly into the initial problem. A last and original contribution of this chapter is that it solves the inverse

stance problem for a system made of multiple robots and objects, which is not the case in any of the previous works.

Note that our problem (3.2), within its static *planning* context, is different from its dynamic *control* counterpart. Precisely, we are not looking for a feasible trajectory, or a steering method, that takes us from an initial configuration and tries to reach specified contact locations. As such, optimization-based iterative inverse kinematics techniques that rely on constraints prioritization, e.g. [**KSP08**][**RB09**], are not necessarily suitable for our particular purpose. Here we are not trying to satisfy constraints at best following a feasible trajectory, but rather to know whether a constrained solution exists or not.

3.3. Problem Formulation

For the notations used in this section we refer the reader to Fig. 3.1.

We suppose that we have a system of N robots and objects indexed by $r \in \{1, \dots, N\}$. To this set we append an additional index 0 referring to the environment. This way we have a coherent and unified description for robot-robot contacts, robot-environment contacts, robot-object contacts, and finally object-environment contacts. For convenience we use the term *robot* when talking about either an actual robot, or a manipulated object, or the environment.

3.3.1. Optimization variables. The configuration vector for a robot $r \in \{1, \dots, N\}$ takes the form

$$q_r = (x_r, y_r, z_r, \quad \alpha_r, \beta_r, \gamma_r, \delta_r, \quad \theta_{r,1}, \theta_{r,2}, \dots, \theta_{r,j_r}), \tag{3.7}$$

which is the concatenation of the Cartesian position of the root body, the unit quaternion representing the orientation of the root body, and the vector θ_r of the j_r joint articulations. $j_r \neq 0$ for an actual robot and $j_r = 0$ for a rigid object. For a body b of the robot r we denote $O_{r,b}(q_r)$ and $R_{r,b}(q_r)$ respectively the origin's position and the orientation matrix of the frame $\mathscr{T}_{r,b}$ attached to the body b. The body $b = 0$ corresponds to the root body of r.

Let us now start from a stance σ made of n contacts

$$\sigma = \{c_1, \dots, c_n\}. \tag{3.8}$$

Each contact c_i is defined between the surface $S_{r_{i1},b_{i1}}$ rigidly attached to the body b_{i1} of the robot $r_{i1} \in \{1, \dots, N\}$, and the surface $S_{r_{i2},b_{i2}}$ rigidly attached to the body b_{i2} of the robot $r_{i2} \in \{0, \dots, N\}$. A surface $S_{r,b}$ is a convex polygon[1] with $V_{r,b}$ vertices

$$S_{r,b} = \operatorname{conv}\left(\{p_{r,b,1}, \dots, p_{r,b,V_{r,b}}\}\right). \tag{3.9}$$

For each point $p_{r,b,v}$ fixed in the local frame of the body b we denote $P_{r,b,v}(q_r)$ its position in the world frame and $P^0_{r,b,v}(\theta_r)$ its position in the root frame of the robot r. At each point $p_{r,b,v}$ we specify a polyhedral cone $\mathscr{C}_{r,b,v}$ with finite number $K_{r,b,v}$ of generators that approximate the friction cone, the axis of which is the inward normal to the surface $S_{r,b}$

$$\mathscr{C}_{r,b,v} = \operatorname{pos}\left(\{u_{r,b,v,1}, \dots, u_{r,b,v,K_{r,b,v}}\}\right). \tag{3.10}$$

The case of a bilateral contact is simply handled by setting

$$\mathscr{C}_{r,b,v} = \mathbb{R}^3, \tag{3.11}$$

[1]This is one assumption of our work. Non-convex polygonal surfaces of the robot are decomposed into a finite set of convex polygons. Non-polygonal convex surfaces are conservatively approximated by polygons.

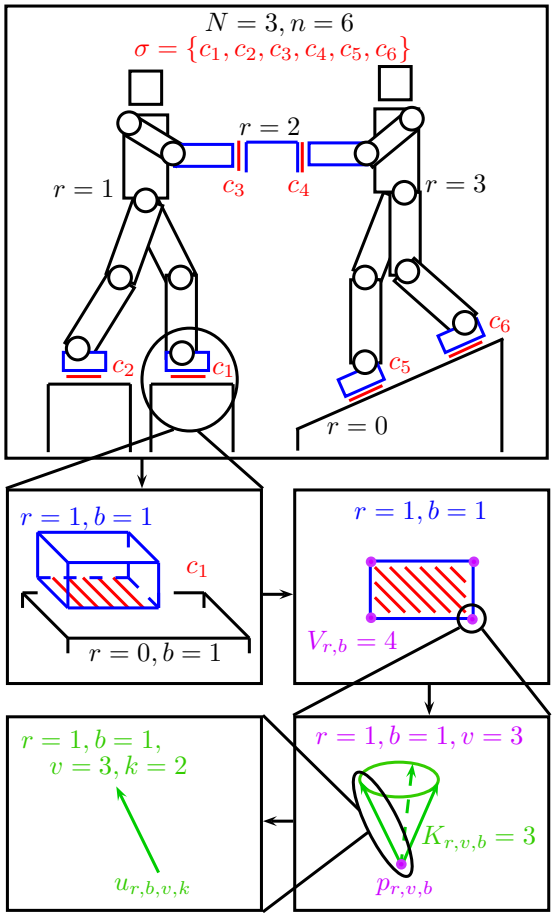


FIGURE 3.1. Illustration of the different levels of indices used in this chapter for an example made of 3 robots (4 including the environment) and a 6 contacts stance.

and, in this case, the vectors u are simply the three basis vectors of $\mathscr{T}_{r,b}$ with no positivity constraints on their coefficients. For each unit vector $u_{r,b,v,k}$ fixed in the local frame of the body b we denote $U_{r,b,v,k}(q_r)$ its coordinates in the global frame.

We can now introduce the statics variables λ. We first suppose without loss of generality (we can permute the indexes 1 and 2) that the area of the surface $S_{r_{i1},b_{i1}}$ is less than the area of the surface $S_{r_{i2},b_{i2}}$, so that when the contact c_i occurs, at the solution, we can write

$$S_{r_{i1},b_{i1}} \subset S_{r_{i2},b_{i2}}. \tag{3.12}$$

The surface contact, at the solution, is thus $S_{r_{i1},b_{i1}}$, and the continuous surface force distribution over this surface can be reduced to a finite force distribution over its vertices. At each vertex $p_{r_{i1},b_{i1},v}$, $v \in \{1, \dots, V_{r_{i1},b_{i1}}\}$, The resulting contact force $f_{r_{i1},b_{i1},v}$ is a non-negative linear combination of the polyhedral friction cone generators

$$f_{r_{i1},b_{i1},v} = \sum_{k=1}^{K_{r_{i1},b_{i1},v}} \lambda_{r_{i1},b_{i1},v,k}\, U_{r_{i1},b_{i1},v,k}(q_{r_{i1}}). \tag{3.13}$$

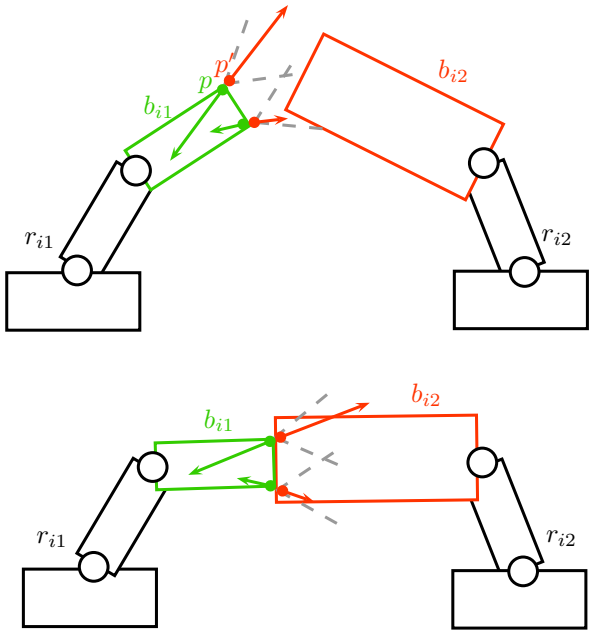


FIGURE 3.2. In green the minimum area surface's body, in red the maximum area surface's body. The contact forces applied on a body are drawn in the same color as the body. Before the contact is established at the solution (top figure), the forces applied on the red body have their application points p originally expressed in the local frame of the green body. To compute the torques resulting from the application of these forces on the red body we have to consider the virtual point p' of the red body's local frame that instantaneously coincides with p at every configurations $q_{r_{i1}}$ and $q_{r_{i2}}$ of the robots.

The forces applied on the body b_{i2} robot r_{i2} will be, at the solution, equal to $-f_{r_{i1},b_{i1},v}$, applied at the same application points. See Fig. 3.2. For each contact c_i we denote λ_i the vector of $(\mathbb{R}^+)^{K_{r_{i1},b_{i1},v}}$ made of all the $\lambda_{r_{i1},b_{i1},v,k}$

$$\lambda_i = (\lambda_{r_{i1},b_{i1},v,1}, \ldots, \lambda_{r_{i1},b_{i1},v,K_{r_{i1},b_{i1},v}}). \tag{3.14}$$

Finally, the variables of our optimization problem (3.3) can be split into:

- geometric variables $q = (q_r)_{r\in\{1,\ldots,N\}}$.
- statics variables $\lambda = (\lambda_i)_{i\in\{1,\ldots,n\}}$.

3.3.2. Geometric constraints. For each contact c_i of the stance σ, a geometric constraint sets the relative position of the frame $\mathscr{T}_{r_{i1},b_{i1}}$ in the frame $\mathscr{T}_{r_{i2},b_{i2}}$. For each couple (r, b) we choose the frame $\mathscr{T}_{r,b}$ so that its origin is inside the surface $S_{r,b}$ and its third basis vector coincides with the inward normal to the surface $S_{r,b}$. Let us denote $(\vec{x}_{r,b}(q_r), \vec{y}_{r,b}(q_r), \vec{z}_{r,b}(q_r))$ the coordinates of the basis vectors of $\mathscr{T}_{r,b}$ in the global frame. A surface contact c_i needs the realization of at least the two following constraints

$$\vec{z}_{r_{i1},b_{i1}}(q_{r_{i1}}) + \vec{z}_{r_{i2},b_{i2}}(q_{r_{i2}}) = 0 \tag{3.15}$$

$$O_{r_{i1},b_{i1}}(q_{r_{i1}})^T \vec{z}_{r_{i2},b_{i2}}(q_{r_{i2}}) = 0. \tag{3.16}$$

This leaves us with three degrees of freedom that we denote $(x_{c_i}, y_{c_i}, \theta_{c_i})$, corresponding to the three following constraints

$$O_{r_{i1},b_{i1}}(q_{r_{i1}})^T \vec{x}_{r_{i2},b_{i2}}(q_{r_{i2}}) = x_{c_i} \tag{3.17}$$

$$O_{r_{i1},b_{i1}}(q_{r_{i1}})^T \vec{y}_{r_{i2},b_{i2}}(q_{r_{i2}}) = y_{c_i} \tag{3.18}$$

$$\vec{x}_{r_{i1},b_{i1}}(q_{r_{i1}})^T \vec{x}_{r_{i2},b_{i2}}(q_{r_{i2}}) = \cos(\theta_{c_i}), \tag{3.19}$$

which can be fixed as equality constraints if we specify a fixed contact location or left as inequality constraints if we wish to realize the contact and leave its location to be decided by the optimization process as a component of the objective cost function.

3.3.3. Collision avoidance constraints. Collision-avoidance constraints are set between any two links b_{nc1} and b_{nc2} that are not connected by a joint in the kinematic-tree representation of the system. Collision-avoidance between connected link is implicitly considered in the joint limit constraint. Let d be a signed distance between two strictly-convex bounding volumes of the links b_{nc1} and b_{nc2}, that we denote respectively $\mathcal{B}_{\text{nc1}}$ and $\mathcal{B}_{\text{nc2}}$. We use as bounding volumes the Sphere-Torus-Patch Bounding Volumes (STP-BV) [**BEMK09**]. By specifying the corresponding support functions, the enhanced GJK [**GJK88**] collision-detection algorithm allows to compute such a continuous differentiable signed distance. The collision-avoidance constraint is thus simply written

$$d(\mathcal{B}_{\text{nc1}}, \mathcal{B}_{\text{nc2}}) > 0\,. \tag{3.20}$$

Fig. 3.3 shows an example of how this constraint generates different resulting configurations corresponding to different local minima.

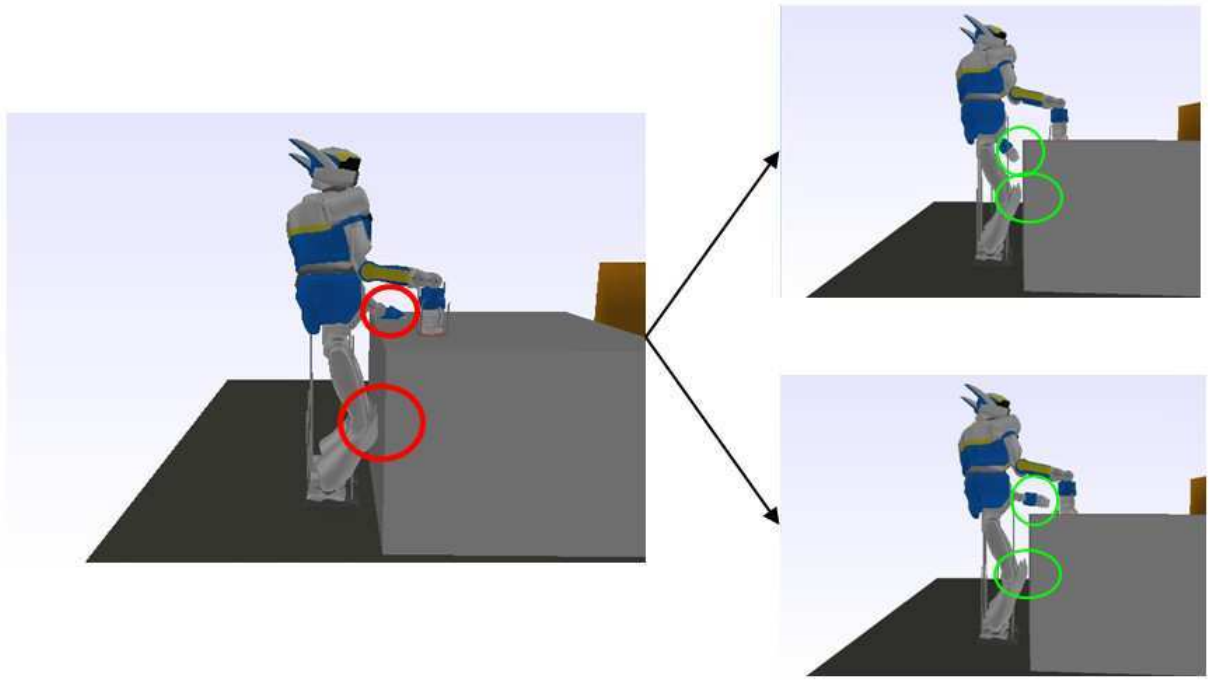


FIGURE 3.3. Collision-avoidance constraint. The left figure shows a configuration generated without collision-avoidance constraints. The two right figures show two possible solutions corresponding to two different local minima.

3.3.4. Static equilibrium constraints. We will write N static equilibrium constraints, one for each robot $r \in \{1, \dots, N\}$. Let us denote g the gravity field vector, m_r the total mass of the robot, $C_r(q_r)$ the coordinates of the CoM of the robot in the global frame. We partition the index set $I = \{1, \dots, n\}$ of the stance contacts $\sigma = \{c_1, \dots, c_n\}$ into three different subsets. $I_1(r)$ is the subset of I made of the contacts in which a surface from r is involved as the minimum area surface. $I_2(r)$ is the subset of I made of the contacts in which a surface from r is involved as the maximum area surface. $I_3(r)$ is the subset of I made of the contacts in which no surface from the robot r is involved.

$$\begin{aligned} I_1(r) &= \{i \mid r_{i1} = r\} \\ I_2(r) &= \{i \mid r_{i2} = r\} \\ I_3(r) &= \{i \mid r \notin \{r_{i1}, r_{i2}\}\}. \end{aligned} \tag{3.21}$$

A fundamental remark in our approach is that the forces acting on $r = r_{i2}$ resulting from the contacts i indexed in I_2 have their application points $(p_{r_{i1},b_{i1},v})_v$ fixed in the frame $\mathscr{T}_{r_{i1},b_{i1}}$ of the other robot r_{i1}. To calculate the torques resulting on the joints of $r = r_{i2}$ we thus need to transform the points p in the frame $\mathscr{T}_{r_{i2},b_{i2}}$. Let us denote the transformed points p' such that, for each v,

$$p'_{r_{i1},b_{i1},v}(q_{r_{i1}}, q_{r_{i2}}) = R_{r_{i2},b_{i2}}(q_{r_{i2}})^T \left(P_{r_{i1},b_{i1},v}(q_{r_{i1}}) - O_{r_{i2},b_{i2}}(q_{r_{i2}})\right). \tag{3.22}$$

For $p \in \mathbb{R}^3$ let us denote $J_{r,b}(q_r, p)$ the following Jacobian matrix

$$J_{r,b}(q_r, p) = \frac{\partial \Big[R_{r,0}(q_r)^T \Big(\big(O_{r,b}(q_r) + R_{r,b}(q_r)p\big) - O_{r,0}(q_r) \Big) \Big]}{\partial \theta_r}. \tag{3.23}$$

We can finally write the static stability constraint for r

$$\sum_{i \in I_1(r)} \sum_{v=1}^{V_{r_{i1},b_{i1}}} f_{r_{i1},b_{i1},v} - \sum_{i \in I_2(r)} \sum_{v=1}^{V_{r_{i1},b_{i1}}} f_{r_{i1},b_{i1},v} + m_r g = 0 \tag{3.24}$$

$$\sum_{i \in I_1(r)} \sum_{v=1}^{V_{r_{i1},b_{i1}}} P_{r_{i1},b_{i1},v} \times f_{r_{i1},b_{i1},v} - \sum_{i \in I_2(r)} \sum_{v=1}^{V_{r_{i1},b_{i1}}} P_{r_{i1},b_{i1},v} \times f_{r_{i1},b_{i1},v} + C_r \times m_r g = 0 \tag{3.25}$$

$$\begin{aligned} \tau_r + \sum_{i \in I_1(r)} \sum_{v=1}^{V_{r_{i1},b_{i1}}} & J_{r_{i1},b_{i1}}(q_{r_{i1}}, p_{r_{i1},b_{i1},v})^T f_{r_{i1},b_{i1},v} \\ & - \sum_{i \in I_2(r)} \sum_{v=1}^{V_{r_{i1},b_{i1}}} J_{r_{i2},b_{i2}}(q_{r_{i2}}, p'_{r_{i1},b_{i1},v})^T f_{r_{i1},b_{i1},v} + \left(\frac{\partial C_r}{\partial \theta_r}\right)^T m_r g = 0\,, \end{aligned} \tag{3.26}$$

where $\tau_r \in \mathbb{R}^{j_r}$ denotes the actuators torques vector. Equation (3.26) gives us the expression of τ_r as a function of the optimization variables q and λ, $\tau_r(q, \lambda)$, and allows us to write the inequality constraint on the maximum torques, denoting $\tau_{r,\mu}$ the μ-th component of τ,

$$\forall \mu \in \{1, \dots, j_r\} \quad |\tau_{r,\mu}(q, \lambda)| \leq \tau_{r,\mu,\max}. \tag{3.27}$$

3.3.5. Objective function. The objective function to minimize in problem (3.3) $\text{obj}(q, \lambda)$ can be chosen in different ways depending on the application we are targeting. One typical choice is a quadratic form

$$\text{obj}(q, \lambda) = (q - q_{\text{ref}})^T A (q - q_{\text{ref}}) + \lambda^T B \lambda, \tag{3.28}$$

if we want to minimize contact forces, or,

$$\text{obj}(q, \lambda) = (q - q_{\text{ref}})^T A (q - q_{\text{ref}}) + \sum_r \tau_r(q, \lambda)^T C\, \tau_r(q, \lambda), \tag{3.29}$$

if we want to minimize actuators torques, q_{ref} being a reference configuration given manually as an input and used to drive the solution towards a goal as well as to produce natural-looking solutions, which is a fundamental concern for a humanoid robot. Within the planning context this reference configuration is taken from a guide path as computed in Appendix A. A, B, C are positive semi-definite matrices. Practically we choose diagonal matrices, the coefficients of which are tuned to weight the different objectives.

3.4. Gradients Derivations

Both state-of-the-art non-linear constrained optimization algorithms we have used, feasible sequential quadratic programming [**LT96**] and interior-point filter line-search [**WB06**], require that we provide them with the gradients of the objective and constraints functions. In this section we give details on these non-trivial gradient derivations. The gradients of all the functions with respect to λ are straightforward to derive, let us focus on the gradients with respect to q.

3.4.1. Geometric Jacobians. All the geometric gradients that we need to compute are down to the expressions of the $\mathbb{R}^{3\times(7+j_r)}$ matrices

$$\frac{\partial O_{r,b}(q_r)}{\partial q_r}, \frac{\partial [R_{r,b}(q_r)\, u]}{\partial q_r}, \frac{\partial [R_{r,b}(q_r)^T u]}{\partial q_r}, \tag{3.30}$$

where u is any fixed vector of $\mathbb{R}^3$. The objective here is to derive these expressions relying only on the kinematic Jacobian of the body b with respect to the root body 0 of the robot r, for which algorithms can be found in standard textbooks such as [**KD05**]. Let us denote this kinematic Jacobian $J^k_{r,b} \in \mathbb{R}^{3\times j_r}$, its μ-th column

$$J^{k,\mu}_{r,b}(q_r) = \begin{bmatrix} \xi^\mu_{r,b}(q_r) \\ \omega^\mu_{r,b}(q_r) \end{bmatrix} \tag{3.31}$$

is the concatenation of the linear and angular velocities of the frame $\mathscr{T}_{r,b}$ with respect the the frame $\mathscr{T}_{r,0}$ expressed in this latter frame, corresponding to a unit velocity of the joint μ, $\dot\theta_{r,\mu} = 1$. If ρ denotes the mapping from unit quaternions to rotation matrices, i.e.

$$\rho(\alpha, \beta, \gamma, \delta) = \begin{pmatrix} 2(\alpha^2+\beta^2)-1 & 2(\beta\gamma-\alpha\delta) & 2(\beta\delta+\alpha\gamma) \\ 2(\beta\gamma+\alpha\delta) & 2(\alpha^2+\gamma^2)-1 & 2(\gamma\delta-\alpha\beta) \\ 2(\beta\delta-\alpha\gamma) & 2(\gamma\delta+\alpha\beta) & 2(\alpha^2+\delta^2)-1 \end{pmatrix}, \tag{3.32}$$

then we can write

$$\frac{\partial O_{r,b}(q_r)}{\partial [x_r, y_r, z_r]} = \mathbb{1}_{3\times 3} \tag{3.33}$$

$$\frac{\partial O_{r,b}(q_r)}{\partial [\alpha_r, \beta_r, \gamma_r, \delta_r]} = \left(\frac{\partial \rho}{\partial \alpha} O^0_{r,b}, \frac{\partial \rho}{\partial \beta} O^0_{r,b}, \frac{\partial \rho}{\partial \gamma} O^0_{r,b}, \frac{\partial \rho}{\partial \delta} O^0_{r,b} \right) \tag{3.34}$$

$$\frac{\partial O_{r,b}(q_r)}{\partial \theta_r} = R_{r,0}\, \xi_{r,b} \tag{3.35}$$

$$\frac{\partial [R_{r,b}(q_r)\, u]}{\partial [x_r, y_r, z_r]} = \mathbb{O}_{3\times 3} \tag{3.36}$$

$$\frac{\partial [R_{r,b}(q_r)\, u]}{\partial [\alpha_r, \beta_r, \gamma_r, \delta_r]} = \left(\frac{\partial \rho}{\partial \alpha} R^0_{r,b} u, \frac{\partial \rho}{\partial \beta} R^0_{r,b} u, \frac{\partial \rho}{\partial \gamma} R^0_{r,b} u, \frac{\partial \rho}{\partial \delta} R^0_{r,b} u \right) \tag{3.37}$$

$$\frac{\partial [R_{r,b}(q_r)\, u]}{\partial \theta_r} = \left(R_{r,0} \big[\omega^\mu_{r,b} \times (R^0_{r,b}\, u) \big] \right)_{\mu \in \{1,\ldots,j_r\}} \tag{3.38}$$

$$\frac{\partial [R_{r,b}(q_r)^T u]}{\partial [x_r, y_r, z_r]} = \mathbb{O}_{3\times 3} \tag{3.39}$$

$$\frac{\partial [R_{r,b}(q_r)^T u]}{\partial [\alpha_r, \beta_r, \gamma_r, \delta_r]} = \left({R^0_{r,b}}^T \frac{\partial \rho}{\partial \alpha}^T u, {R^0_{r,b}}^T \frac{\partial \rho}{\partial \beta}^T u, {R^0_{r,b}}^T \frac{\partial \rho}{\partial \gamma}^T u, {R^0_{r,b}}^T \frac{\partial \rho}{\partial \delta}^T u \right) \tag{3.40}$$

$$\frac{\partial [R_{r,b}(q_r)^T u]}{\partial \theta_r} = \left(- {R^0_{r,b}}^T \big[\omega^\mu_{r,b} \times (R^T_{r,0} u) \big] \right)_{\mu \in \{1,\ldots,j_r\}}, \tag{3.41}$$

where we have used the following notation

$$R^0_{r,b} = R^T_{r,0}\, R_{r,b}. \tag{3.42}$$

3.4.2. Torques gradients. Let us now derive the gradient of the constraint (3.27) for which the main difficulty resides in the derivation of

$$J_1 = \frac{\partial J^\mu_{r_{i2},b_{i2}} \Big(q_{r_{i2}}, p'_{r_{i1},b_{i1},v}(q_{r_{i1}}, q_{r_{i2}}) \Big)}{\partial q_{r_{i1}}}, \tag{3.43}$$

$$J_2 = \frac{\partial J^\mu_{r_{i2},b_{i2}} \Big(q_{r_{i2}}, p'_{r_{i1},b_{i1},v}(q_{r_{i1}}, q_{r_{i2}}) \Big)}{\partial q_{r_{i2}}}. \tag{3.44}$$

where $J^\mu_{r,b}(q_r, p)$ is the μ-th column of the matrix defined in (3.23). Let us denote $D_{q_r} J^\mu_{r,b}$ and $D_p J^\mu_{r,b}$ respectively the partial derivatives of $J^\mu_{r,b}(q_r, p)$ with respect to q_r and to p. We can write (we temporarily drop the subscripts of p')

$$J_1 = D_p J^\mu_{r_{i2},b_{i2}}(q_{r_{i2}}, p')\, \frac{\partial p'(q_{r_{i1}}, q_{r_{i2}})}{\partial q_{r_{i1}}}, \tag{3.45}$$

$$J_2 = D_{q_r} J^\mu_{r_{i2},b_{i2}}(q_{r_{i2}}, p') + D_p J^\mu_{r_{i2},b_{i2}}(q_{r_{i2}}, p')\, \frac{\partial p'(q_{r_{i1}}, q_{r_{i2}})}{\partial q_{r_{i2}}}. \tag{3.46}$$

In these expressions the derivatives

$$\frac{\partial p'(q_{r_{i1}}, q_{r_{i2}})}{\partial q_{r_{i1}}}, \frac{\partial p'(q_{r_{i1}}, q_{r_{i2}})}{\partial q_{r_{i2}}}, \tag{3.47}$$

can be obtained directly using the matrices (3.30), denoting temporarily $u = P_{r_{i1},b_{i1},v}(q_{r_{i1}}) - O_{r_{i2},b_{i2}}(q_{r_{i2}})$,

$$\frac{\partial p'(q_{r_{i1}}, q_{r_{i2}})}{\partial q_{r_{i1}}} = R_{r_{i2},b_{i2}}(q_{r_{i2}}) \frac{\partial P_{r_{i1},b_{i1},v}(q_{r_{i1}})}{\partial q_{r_{i1}}} \tag{3.48}$$

$$\frac{\partial p'(q_{r_{i1}}, q_{r_{i2}})}{\partial q_{r_{i2}}} = \frac{\partial \left[R_{r_{i2},b_{i2}}(q_{r_{i2}})^T \, u \right]}{\partial q_{r_{i2}}} - R_{r_{i2},b_{i2}}(q_{r_{i2}})^T \, \frac{\partial O_{r_{i2},l_{i2}}(q_{r_{i2}})}{\partial q_{r_{i2}}} \, . \tag{3.49}$$

We can now concentrate on the derivations of $D_{q_r} J^{\mu}_{r,b}$ and $D_p J^{\mu}_{r,b}$. First, for $D_{q_r} J^{\mu}_{r,b}$, we can write

$$\frac{\partial J^{\mu}_{r,b}}{\partial [x_r, y_r, z_r]} = \mathrm{O}_{3\times 3} \tag{3.50}$$

$$\frac{\partial J^{\mu}_{r,b}}{\partial [\alpha_r, \beta_r, \gamma_r, \delta_r]} = \left(\frac{\partial \rho}{\partial \alpha} \, \xi^{\mu}_{r,b}(p), \frac{\partial \rho}{\partial \beta} \, \xi^{\mu}_{r,b}(p), \frac{\partial \rho}{\partial \gamma} \, \xi^{\mu}_{r,b}(p), \frac{\partial \rho}{\partial \delta} \, \xi^{\mu}_{r,b}(p) \right) \tag{3.51}$$

$$\frac{\partial J^{\mu}_{r,b}}{\partial \theta_r} = R_{r,0} \left(\frac{\partial \xi^{\mu}_{r,b}(p)}{\partial \theta_{r,\nu}} \right)_{\nu \in \{1,\ldots,j_r\}} , \tag{3.52}$$

where

$$\xi^{\mu}_{r,b}(p) = \xi^{\mu}_{r,b} + \omega^{\mu}_{r,b} \times \left[R^0_{r,b} \, p \right] \tag{3.53}$$

is the velocity transported from the origin of the frame $\mathscr{T}_{r,b}$ to the point p, and

$$\frac{\partial \xi^{\mu}_{r,b}(p)}{\partial \theta_{r,\nu}} = \begin{cases} \omega^{\nu}_{r,b} \times \xi^{\mu}_{r,b}(p) & \text{if } \nu < \mu, \\ \omega^{\mu}_{r,b} \times \xi^{\mu}_{r,b}(p) & \text{if } \nu = \mu, \\ \omega^{\mu}_{r,b} \times \xi^{\nu}_{r,b}(p) & \text{if } \nu > \mu. \end{cases} \tag{3.54}$$

This latter result is a straightforward generalization of the result published in [**BS96**] from serial kinematic chains to kinematic trees such as a humanoid robot. Now that we have derived $D_{q_r} J^{\mu}_{r,b}$ let us derive $D_p J^{\mu}_{r,b}$. We can simply write

$$D_p J^{\mu}_{r,b} = R_{r,0} \, \tilde{\omega}^{\mu}_{r,b} \, R^0_{r,b} \tag{3.55}$$

where $\tilde{\omega}^{\mu}_{r,b}$ is the skew-symmetric matrix corresponding to the vector product by $\omega^{\mu}_{r,b}$. This brings our derivations to an end.

3.4.3. Distance gradient. Let us compute the gradient of the constraint (3.20). Let P_1 and P_2 be respectively the closest points on $\mathcal{B}_{\mathrm{nc}1}$ and $\mathcal{B}_{\mathrm{nc}2}$, such that $d(\mathcal{B}_{\mathrm{nc}1}, \mathcal{B}_{\mathrm{nc}2}) = ||P_2 - P_1||$ if there is no collision and the farthest points such that $d(\mathcal{B}_{\mathrm{nc}1}, \mathcal{B}_{\mathrm{nc}2}) = -||P_2 - P_1||$ in case of interpenetration. The GJK algorithm applied on the STP-BV allows for the computation of such so-called witness points. The result in [**LLB05**] makes the computation of this gradient straightforward by considering the Jacobians at the points $P_{1_{\in \mathcal{B}_{\mathrm{nc}1}}}$ and $P_{2_{\in \mathcal{B}_{\mathrm{nc}2}}}$ that are rigidly attached to $\mathcal{B}_{\mathrm{nc}1}$ and $\mathcal{B}_{\mathrm{nc}2}$ respectively and coincide with P_1 and P_2 in the configuration q at which we are computing the gradient. So we can write

$$\frac{\partial d(\mathcal{B}_{\mathrm{nc}1}, \mathcal{B}_{\mathrm{nc}2})}{\partial q} = \begin{cases} \frac{(P_1 - P_2)}{||P_1 - P_2||}^T \left(\frac{\partial P_{1_{\in \mathcal{B}_{\mathrm{nc}1}}}}{\partial q_{r_1}} - \frac{\partial P_{2_{\in \mathcal{B}_{\mathrm{nc}2}}}}{\partial q_{r_2}} \right) & \text{if there is no collision,} \\ -\frac{(P_1 - P_2)}{||P_1 - P_2||}^T \left(\frac{\partial P_{1_{\in \mathcal{B}_{\mathrm{nc}1}}}}{\partial q_{r_1}} - \frac{\partial P_{2_{\in \mathcal{B}_{\mathrm{nc}2}}}}{\partial q_{r_2}} \right) & \text{if there is interpenetration.} \end{cases} \tag{3.56}$$

TABLE 3.1. Some figures

	Circus	Coll.	Ladder
$\dim(q)$	94	101	47
$\dim(\lambda)$	48	96	48
total dimension	142	197	95
num. of eq. constr.	34	61	27
num. of ineq. const.	80	80	40
num. of iterations	30	42	19
optim. algo. time	0.732s	1.423s	0.280s
func. & grad. eval. time	7.190s	9.515s	1.454s

3.5. Results

We have tested our static stance inverse solver on different theoretic scenarios in virtual environments involving one or two humanoid robots (for the robot we used a model of HRP-2 [**KKK$^+$04**]) conjointly manipulating objects and taking unilateral or bilateral contacts, see Fig. 3.4. Our implementation being generic and totally transparent to the robot model, any other robot could have been used with no additional model-specific implementation effort. Of course some of these scenarios are not meant to be simulated or executed on real-life robots but we choose them to illustrate the generality of our approach from the conceptual point-of-view.

Within multi-contact planning queries made with a planner similar to [**EKM06**], no local minima problems were encountered. This is mainly due to the fact that during the stances exploration phase, i.e. when growing the search tree, we use the resulting configuration from the father stance node as an initial guess for testing a new stance with our solver and add it to the tree in case of success. Care should thus be taken only when choosing the very first configuration initializing the search tree.

Table 3.1 gives some figures[2] concerning queries on these scenarios made with the solver [**WB06**] on a standard 3.06 GHz computer. As we can see most of the computation time is spent on functions and gradients evaluations and can be greatly reduced, given that our current implementation splits vector constraints into individual scalar constraints and thus wastes a lot of time in redundant computations that can be factorized when using vector constraints. However, although computational time appears to be quite heavy (still being of same order of magnitude as the times reported in [**EKM06**][**HBL$^+$08**] for more complex problems in our case), it allows for solving multi-contact planning queries in times comparable to those of the aforementioned state-of-the-art planners, i.e. tens of minutes in average, while being more generic and handling a broader range of contact situations.

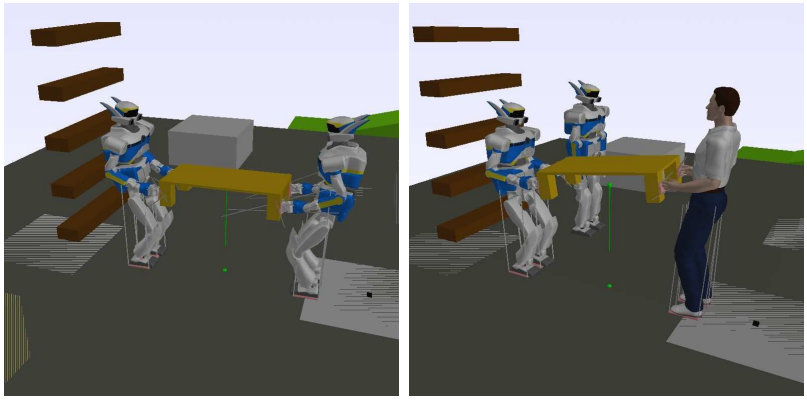
3.6. Conclusion

We provided a formulation for the multiple robots, multiple objects, multiple contacts, static stance inverse problem. The problem has been written as an optimization

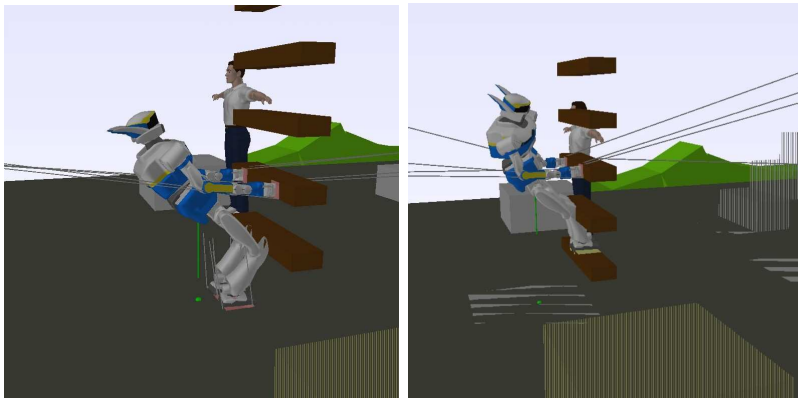
[2]The number of inequality contraints do not include the bounds on joints articulations nor the positivity conditions on λ for unilateral contacts as these bounds are handled directly as limits on the optimization variables by the solver.

problem in the geometric and statics variables conjointly. Analytical gradients based on the kinematic Jacobian and its derivatives have been derived. We have tested our approach on very high dimensional challenging scenarios for which solutions were found in a relatively small number of iterations. A possible extension of this work is considering deformable bodies of the robots or the environment (Chapter 5). In the longer term, non-static (kinetic) friction model can also be considered allowing displacement of the environment objects under the action of contact forces. Finally, coming work should decrease computational time by a finer implementation of our solver.

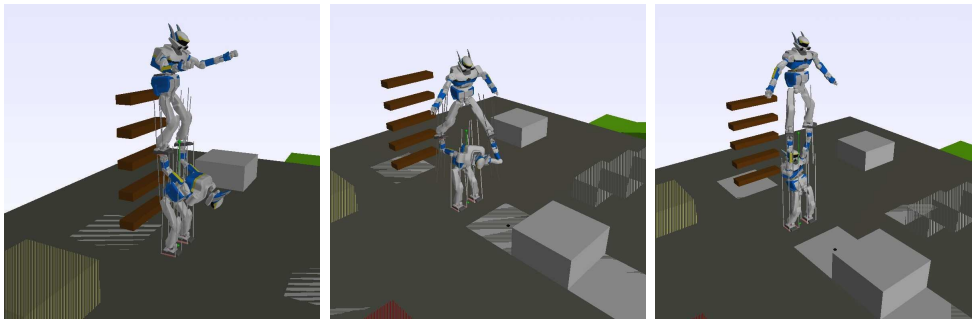
Next chapter (Chapter 4) takes into account the dynamics of the robot in tackling the continuous motion generation problem between the static postures that have been generated by the solver presented in this chapter.



(a) Collaborative object manipulation scenarios with bilateral contacts between the hands and the table.



(b) Ladder climbing scenarios with unilateral contacts at the feet and bilateral contacts at the hands.



(c) Circus scenario involving only unilateral contacts, for different initial guesses.

FIGURE 3.4. Example scenarios

CHAPTER 4

Using a Multi-Objective Controller to Synthesize Simulated Humanoid Robot Motion with Changing Contact Configurations

Our objective in this chapter is to synthesize dynamically consistent motion for a simulated humanoid robot in acyclic multi-contact locomotion using multi-objective control. We take as an input the planned sequence of static postures that represent the contact configuration transitions as computed in the previous chapters (Chapters 2 and 3); a multi-objective controller then synthesizes the motion between these postures, the objectives of the controller being decided by a finite-state machine. Results of this approach are presented in on-line-available video in the form of playback motions generated through non-real-time constraint-based dynamic simulations.

4.1. Introduction

In Chapter 2 we presented an algorithm that plans a sequence of multi-contact stances with corresponding static postures that brings a humanoid robot from an initial configuration to a desired stance/configuration. As opposed to the walking pattern generation problem [**KKK$^+$03, APE04**], this approach is aimed at generating non-gaited acyclic motion with arbitrary contact configurations (using hands, forearms, knees, etc.). The presented algorithm was the first of a two-stage contact-before-motion planning framework. The second stage, which is the main concern of the present chapter, is to synthesize a continuous motion that goes through the planned sequence of static postures. Previous approaches of the problem [**EKM06, HBL05**] used randomized motion planning techniques (RRTs, PRMs) to plan the continuous motion. However, due to the geometric nature of such techniques, which is not suited for the integration of dynamics motion constraints in the planning, their motion was restricted to be quasi-static, meaning that static equilibrium is respected at every time of the motion. Adapting kinodynamic planning or dynamic filtering techniques [**KKN$^+$02, YBEL05**] to the output motion of these works could have been one way to overcome this limitation.

In this chapter we investigate a different approach, that has both its advantages and drawbacks over the previous one. On the plus side it directly synthesizes dynamically consistent motion, for which the dynamics equation of motion is satisfied throughout the motion. The main drawback of this approach is that it does not allow for explicit formulation of collision-freeness constraint, which induces us to resort to hand-designed heuristics in order to avoid collisions. However though, the closed-loop nature of our approach makes it robust to unavoided collision (contact) events that may occur during the generation of the motion, as these collisions are seen as perturbations "absorbed" by the feedback motion generation law.

The organization of the chapter is as follows. After discussing related work (Section 4.2) and an overview of the approach (Section 4.3), we get to the detailed technical developments by first presenting the multi-objective controller used for a single step motion (Section 4.4) followed by the finite-state machine used for multiple steps motion (Section 4.5). Finally we describe the results that appear in the on-line video (Section 4.6).

4.2. Related Work and Contribution

The method we choose is inspired by recent trend in computer graphics community, synthesizing physics-based motion of simulated characters [**AdSP07, dLMH10**]. They formulate the motion generation problem as the control problem of the human character within simulation. Humanoid robotics [**YH10, SBB09**] as well as virtual reality communities [**Col09**] recently started using same/similar formulations in their applications.

[**AdSP07, YH10**] use motion-capture data to generate the motion, their objective being precisely to track these data with the simulated human character or humanoid robot. Our method here does not need such data as it relies only on the information carried by the first stage of the contact-before-motion planning framework that produces the sequence of static postures. This sequence of static postures plays the same role the motion-capture data does in the other works, i.e. solving for redundancies and producing natural looking motion. Therefore one of the main original features in our method is increased autonomy of the robot. [**dLMH10**] also does not rely on motion-capture data, but their applications are restricted to human cyclic walking and a single upstanding posture is sufficient to solve for the said redundancies, while we are targeting more general acyclic motions.

Moreover, most of these works [**AdSP07, YH10, Col09, SBB09**] produce motion for the robot in a single stance, standing either on one foot or two-feet stances, without changing their contact configuration. As in [**dLMH10**] our method adds a finite-state machine to perform motions that go through changing contact-configuration stances, but, once again as opposed to [**dLMH10**], in an acylic way.

Note, however, that although [**AdSP07, dLMH10**] report computation times that reach near real-time objectives using an LCP-based simulator on the animated characters, we do not focus in this work on optimizing the computation time to be real-time since in our current implementation some real robot constraints (self-collisions and joint limits) need to be checked a posteriori, i.e. once the full motion has been generated, before safely executing it on the robot (cf. Section 4.6). This is why our method here is presented as an off-line motion generation tool in simulation rather than an on-line control one directly embedded in the real robot.

Other approaches use different formulations to control/generate motion of humanoid robot in multi-contact stances [**SPK10, LMKY10**]. The former uses a prioritized tasks hierarchy formulation but does not explicitly take into account contact constraints, the latter approach is conceptually different as it generates optimal motion through a semi-infinite optimization formulation of B-spline parametrized motion. While no time complexity analysis has been reported for [**SPK10**], the approach in [**LMKY10**] applied to humanoid robot requires for now computation times as high as a few hours to generate a one-minute motion. Finally, the recent work of [**SRM+11**] uses hierarchy of tasks in a cascade of quadratic programs handling inequality, equality, and unilateral contact constraints based on the work of [**EMW10**].

4.3. Overview of the Method

Fig. 4.1 shows an overview of the proposed motion generation method. In this figure the multi-contact stances planner block (Chapter 2) and the simulator block [**CMK$^+$06**] are considered as black-box modules. Their implementation is not developped in this chapter, which focuses on the finite-state machine and the multi-objective controller designs. t denotes the simulation time; q, $\dot{q}$, and $\ddot{q}$ denote respectively the configuration, configuration velocities, and configuration accelerations of the humanoid robot, which include both the actuated joints and the root $SE(3)$ component of the robot; f denotes the aggregated vector of contact forces applied at finite contact points between the robot and the environment; and u denotes the actuator torques that control the simulated robot. q_s and q_g are the start and goal configuration input by the user. The motion parameters, also input by the user, include the step time, step height, etc. and are further detailed in the finite-state machine description section (Section 4.5).

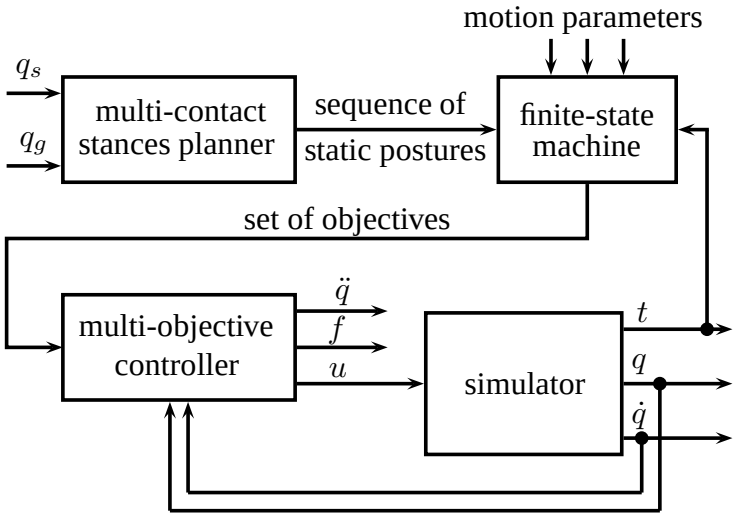


FIGURE 4.1. Overview of the motion generator

At every time step of the simulation, the finite-state machine decides on the objectives to feed to the controller, which then uses a quadratic formulation that solves for the configuration accelerations, the contact forces, and the control torques. Note that the produced configuration accelerations $\ddot{q}$ could be directly integrated to update q and $\dot{q}$. We choose however to discard the produced $\ddot{q}$, along with the computed contact forces f, and to keep only the control u that we feed to the simulator which will in turn output a more accurate f and $\ddot{q}$ to be integrated. This approach, the same as the one chosen in [**AdSP07, dLMH10**], allows for a complete decoupling of the controller and the simulator blocks, the latter can later be replaced by any other one, more accurate or faster, depending on the targeted application. In particular, replacing the simulator by the real robot can be seen as particular case, provided that adequate sensors/estimators feed us back with q and $\dot{q}$ (especially the $SE(3)$ components of these vectors).

4.4. Multi-Objective Controller

The multi-objective controller minimizes a weighted sum of objectives subject to the following constraints:

- satisfy the dynamics equation of motion,
- non-sliding of the contact points,
- contact forces inside the (linearised) friction cone,
- actuation torques within their limits.

The objectives are specified in terms of *tasks* (the term *features* is alternatively used in the literature). A task is for example driving the position of an end-effector, the center of mass of the robot, the whole configuration of the robot, etc. Conflicts between different tasks are solved through a weighted formulation rather than strict prioritization. More technical details of the optimization problem formulation is found in the following subsections:

4.4.1. The Linear Constraints. Let us suppose we have a humanoid robot made of r revolute joints and $r+1$ links indexed by $k \in \{0, \dots, r\}$. On each link k a set of contact forces $f_{k,1}, \dots, f_{k,m_k}$ are applied at the respective local-frame-expressed points $a_{k,1}, \dots, a_{k,m_k}$. Let $q = (x_0, \theta_0, \hat{q}) \in \mathbb{R}^{3+4+r}$ denote the configuration vector of the humanoid robot, where x_0 is the global-frame-expressed position of the root, θ_0 a parametrization of its orientation (a unit quaternion for instance), and $\hat{q}$ the internal (actuated) joint angles vector. For each k, $J_{tk}(p)$ denotes the $3 \times (3+4+j)$ translational Jacobian of the link k relative to the global frame with respect to q expressed at a local-frame-expressed point p.

The motion of the humanoid robot is governed by the following equation (see Section 4.8 and [**Wie05**] from which we borrow the notations for details on how we derive this equation, especially for the expressions of M and N, g is the gravity vector):

$$M(q)\ddot{q} + N(q,\dot{q})\dot{q} = M(q) \begin{pmatrix} g \\ 0_4 \\ 0_r \end{pmatrix} + \begin{pmatrix} 0_3 \\ 0_4 \\ u \end{pmatrix} + \sum_{k,i} J_{tk}(a_{k,i})^T f_{k,i}\,, \tag{4.1}$$

The motion is additionally subject to the following constraints, denoting $K_{k,i}$ the Coulomb friction cone at $a_{k,i}$,

$$\forall\, k, i \quad J_{tk}(a_{k,i})\,\dot{q} = 0\,, \tag{4.2}$$

$$\forall\, k, i \quad f_{k,i} \in K_{k,i}\,, \tag{4.3}$$

$$\forall\, j \quad u_{j,\min} \le u_j \le u_{j,\max}\,. \tag{4.4}$$

We linearise the friction cone $K_{k,i}$ by specifying a finite set of global-frame-expressed generators $\{v_{k,i,1}, \dots, v_{k,i,\nu_{k,i}}\}$ so that each contact force $f_{k,i}$ is a non-negative linear combination of the vectors v:

$$f_{k,i} = \sum_{\mu=1}^{\nu_{k,i}} \lambda_{k,i,\mu}\, v_{k,i,\mu}\,, \tag{4.5}$$

$$\forall\, k, i, \mu \quad \lambda_{k,i,\mu} \ge 0\,. \tag{4.6}$$

We denote $\lambda = (\lambda_{k,i,\mu})_{k,i,\mu}$.

By time-differentiating the constraint (4.2) we get

$$J_{tk}(a_{k,i})\,\ddot{q} + \dot{J}_{tk}(a_{k,i})\,\dot{q} = 0\,. \tag{4.7}$$

Let us define the parameter vector $X = (\ddot{q}, \lambda, u)$. For clarity we denote

$$\alpha = \dim(\ddot{q}) = 3 + 4 + r\,, \qquad \gamma = \dim(u) = r\,, \tag{4.8}$$

$$\beta = \dim(\lambda) = \sum_{k=0}^{m} \sum_{i=1}^{m_k} \nu_{k,i}\,, \qquad \zeta = \sum_{k=0}^{m} m_k\,. \tag{4.9}$$

Furthermore, for a family of same-size matrices $(Y_\iota)_{\iota\in\{1,\dots,I\}}$, we denote the block aggregation operators

$$\lfloor Y_\iota \rfloor_{\iota\in\{1,\dots,I\}} = \begin{pmatrix} Y_1 \\ \vdots \\ Y_I \end{pmatrix}\,, \quad \lceil Y_\iota \rceil_{\iota\in\{1,\dots,I\}} = \begin{pmatrix} Y_1 & \dots & Y_I \end{pmatrix}\,. \tag{4.10}$$

Finally, the equation of motion (4.1) and constraints (4.4), (4.6), (4.7) take the following linear form

$$A_1\, X = B_1\,, \quad A_2\, X \leq B_2\,, \tag{4.11}$$

where the matrices A_1,A_2 and the vectors B_1,B_2 are defined

$$A_1 = \begin{pmatrix} M(q) & -\lceil J_{tk}(a_{k,i})^T v_{k,i,\mu} \rceil_{k,i,\mu} & -\begin{pmatrix} 0_{3\times\gamma} \\ 0_{4\times\gamma} \\ 1_{\gamma\times\gamma} \end{pmatrix} \\ \lfloor J_{tk}(a_{k,i}) \rfloor_{k,i} & 0_{3\zeta\times\beta} & 0_{3\zeta\times\gamma} \end{pmatrix}\,,$$

$$B_1 = \begin{pmatrix} -N(q,\dot{q})\dot{q} + M(q)\begin{pmatrix} g \\ 0_4 \\ 0_r \end{pmatrix} \\ \lfloor -\dot{J}_{tk}(a_{k,i})\dot{q} \rfloor_{k,i} \end{pmatrix}\,,$$

$$A_2 = \begin{pmatrix} 0_{\beta\times\alpha} & -1_{\beta\times\beta} & 0_{\beta\times\gamma} \\ 0_{\gamma\times\alpha} & 0_{\gamma\times\beta} & -1_{\gamma\times\gamma} \\ 0_{\gamma\times\alpha} & 0_{\gamma\times\beta} & 1_{\gamma\times\gamma} \end{pmatrix}\,, \quad B_2 = \begin{pmatrix} 0_\beta \\ -u_{\min} \\ u_{\max} \end{pmatrix}\,. \tag{4.12}$$

Let us now write the target function to optimize.

4.4.2. The Quadratic Objectives. We define a *task* (or *feature*) as a scalar or vector function g of the configuration of the robot $g : \mathbb{R}^{3+4+r} \to \mathbb{R}^d$, where d is the dimensionality of the task. Example of such tasks include the global-frame expression of a particular point attached to one of the robot's links ($d = 3$), the CoM of the entire robot ($d = 3$), the configuration itself of the robot ($d = 3 + 4 + r$), etc. Let J_g denote the Jacobian of the task, i.e. the $(3 + 4 + r) \times d$ matrix $J_g(q) = \partial g/\partial q$.

As proposed in [**dLMH10**] we will use two kinds of objectives for the task g:

- a *set-point objective*, denoted $E_{\text{spt},g}$, used if we wish to servo the task g around an given reference value g_{ref},
- a *target objective*, denoted $E_{\text{tgt},g}$, used if we wish to steer the task g from a given initial value $(g_0, \dot{g}_0)$ to a given target final value $(g_f, \dot{g}_f)$ in given time t_f.

The Set-point Objective. The corresponding objective function component takes the form

$$\begin{aligned} E_{\text{spt},g}(X) &= \frac{1}{2}\,||\kappa_p(g_{\text{ref}} - g) - \kappa_v\dot{g} - \ddot{g}||^2\,, \\ &= \frac{1}{2}\,X^T Q X + c^T X + \frac{1}{2}\,c^T c\,, \end{aligned} \tag{4.13}$$

where

$$Q = \begin{pmatrix} J_g^T J_g & 0_{\alpha\times\beta} & 0_{\alpha\times\gamma} \\ 0_{\beta\times\alpha} & 0_{\beta\times\beta} & 0_{\beta\times\gamma} \\ 0_{\gamma\times\alpha} & 0_{\gamma\times\beta} & 0_{\gamma\times\gamma} \end{pmatrix}, c = \begin{pmatrix} -J_g^T \left(\kappa_p(g_{\text{ref}} - g) - \kappa_v J_g \dot{q} - \dot{J}_g \dot{q} \right) \\ 0_\beta \\ 0_\gamma \end{pmatrix} \tag{4.14}$$

. κ_p and κ_v are hand-tuned gain parameters, in our applications we systematically set $\kappa_v = 2\sqrt{\kappa_p}$.

The Target Objective. Let t_0 denote the current time. Let g^i be the i-th scalar component of g for $i \in \{1, \ldots, d\}$. For every such g^i our objective is to reach the specified target $(g_f^i, \dot{g}_f^i)$ at time $t_f > t_0$. The method proposed in [**dLMH10**] consists in making g^i follow a constant-jerk reference trajectory of the form

$$\ddot{g}_{\text{ref}}^i(t) = \left(1 - \frac{t - t_0}{t_f - t_0}\right) \phi_{i,t_0} + \frac{t - t_0}{t_f - t_0} \psi_{i,t_0}, \ t \in [t_0, t_f] \tag{4.15}$$

where ϕ_{i,t_0} and ψ_{i,t_0} are coefficients determined by integrating (4.15) twice and writing the boundary values conditions

$$\begin{pmatrix} (t_f - t_0)^2/3 & (t_f - t_0)^2/6 \\ (t_f - t_0)/2 & (t_f - t_0)/2 \end{pmatrix} \begin{pmatrix} \phi_{i,t_0} \\ \psi_{i,t_0} \end{pmatrix} = \begin{pmatrix} g_f^i - g^i - (t_f - t_0)\, \dot{g}^i \\ \dot{g}_f^i - \dot{g}^i \end{pmatrix}. \tag{4.16}$$

Finally, back to the target objective, the corresponding objective function component will take the form

$$\begin{aligned} E_{\text{tgt},g}(X) &= \sum_{i=1}^{d} \frac{1}{2} \left(\ddot{g}_{\text{ref}}^i(t_0) - \ddot{g}^i\right)^2 = \sum_{i=1}^{d} \frac{1}{2} \left(\phi_{i,t_0} - \ddot{g}^i\right)^2, \\ &= \frac{1}{2} X^T Q X + c^T X + \frac{1}{2} c^T c, \end{aligned} \tag{4.17}$$

where, denoting $\Phi_{t_0} = (\phi_{i,t_0})_i$,

$$Q = \begin{pmatrix} J_g^T J_g & 0_{\alpha\times\beta} & 0_{\alpha\times\gamma} \\ 0_{\beta\times\alpha} & 0_{\beta\times\beta} & 0_{\beta\times\gamma} \\ 0_{\gamma\times\alpha} & 0_{\gamma\times\beta} & 0_{\gamma\times\gamma} \end{pmatrix}, c = \begin{pmatrix} -J_g^T \left(\Phi_{t_0} - \dot{J}_g \dot{q} \right) \\ 0_\beta \\ 0_\gamma \end{pmatrix}. \tag{4.18}$$

4.4.3. Putting it Altogether: The QP Formulation. We suppose now that we have N objectives indexed by $k \in \{1, \ldots, N\}$, denoted $g_1, \ldots, g_N$. These objectives can be either set-point or target objectives, with corresponding matrices Q_k and vectors c_k as derived in the previous section. Each objective g_k is allocated a weight w_k that expresses its relative importance when conflicting with other objectives. We then denote the weighted sums

$$Q_{\text{sum}} = \sum_{k=1}^{N} w_k Q_k, \quad c_{\text{sum}} = \sum_{k=1}^{N} w_k c_k. \tag{4.19}$$

The Quadratic Program solved by the multi-objective controller at every time step takes the final form

$$\begin{aligned} \min_X \quad & \frac{1}{2} X^T Q_{\text{sum}} X + c_{\text{sum}}^T X, \\ \text{subject to} \quad & A_1 X = B_1, \ A_2 X \leq B_2. \end{aligned} \tag{4.20}$$

4.5. Finite-State Machine

Let us us now start back from the output of the multi-contact stances planner as portrayed in Fig. 4.1, which is a sequence of n statically stable configurations $(q_0, \ldots, q_{n-1})$. Each configuration q_i is associated with a *stance* σ_i; a stance being the set of contacts that the robot establishes with the environment when put in that configuration. For example when the robot stands on two feet then the corresponding stance is a set containing two contacts (one for each foot). The sequence of stances $(\sigma_0, \ldots, \sigma_{n-1})$ output by the planner are so-called *sequentially adjacent* (Chapter 2), i.e. they satisfy the following condition: each stance σ_i either adds one contact to the previous stance σ_{i-1} or removes one contact from this same previous stance σ_{i-1}. Furthermore, the sequence of configurations $(q_0, \ldots, q_{n-1})$ are so-called *transition configurations* (Chapter 2), meaning that:

- when a contact has been added then the corresponding configuration q_i has to be statically stable with non-zero contact forces applied only at the contacts of the previous stance σ_{i-1}, the contact forces applied at the newly added contact are zero,
- when a contact has been removed then the corresponding configuration q_i keeps all the contacts of the previous stance σ_{i-1} but the contact forces at the newly removed contact are zero.

The motion from q_i to q_{i+1} (from σ_i to σ_{i+1}) will be called *step* number i. So the full motion will comprise $n-1$ steps. We define a user-input parameter T which is the desired step time. So step i starts at time $t = i\,T$ and ends at time $t = (i+1)\,T$. The full duration of the motion is $(n-1)\,T$.

When step i adds a contact then the link of the added contact (the "swing" link, generalizing the terminology of swing foot in legged locomotion) will be denoted s_i and one arbitrarily chosen point attached to this link and belonging to the contact surface is denoted p_i. The global-frame-expressed position of p_i at configuration q_i (start position) is denoted $P_{i,s}$ and the global-frame-expressed position of p_i at configuration q_{i+1} (goal position) is denoted $P_{i,g}$.

4.5.1. Obstacle Collision Avoidance: Controlling the Swing Link. When step i is removing a contact we implicitly make the assumption that the motion from q_i to q_{i+1} (performed inside the sub-manifold of the configuration space corresponding to the stance σ_i) is collision-free. When step i adds a contact however, then the motion of the swing link s_i has to be more carefully controlled since there is high probability that this link collides with the target environment contact support object; e.g. when climbing stairs

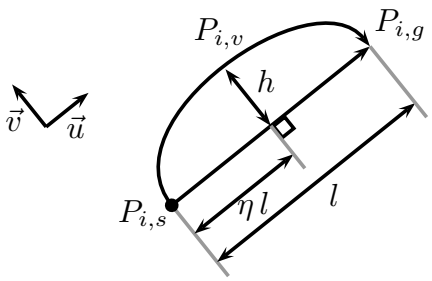


FIGURE 4.2. Controlling the point p_i of the swing link s_i

then the swing foot might collide with the next stair. We introduce a simple heuristic to avoid this, which consists in steering the swing point p_i through a global-frame-expressed *via-point* $P_{i,v}$, defined by specifying a step height h and an intermediate time $T_v < T$ (for example one might choose $T_v = T/2$). So the motion of p_i starts from $P_{i,s}$ at time $t = i\,T$, goes through $P_{i,v}$ at time $t = i\,T + T_v$, and reaches $P_{i,g}$ at time $t = (i+1)\,T$.

Let us denote the step length $l = \|P_{i,g} - P_{i,v}\|$. To define the via-point $P_{i,v}$ we decompose the motion of the swing point p_i into a *parallel component* in the direction of the vector $\vec{u} = (P_{i,g} - P_{i,v})/l$, and a *normal component* following the direction of the vector $\vec{v} = \vec{u} \times (\vec{e}_z \times \vec{u})$ (such that $\vec{v}$ is normal to $\vec{u}$ and in the plan defined by $\vec{u}$ and $\vec{e}_z$; $\vec{e}_z$ being the upwards vertical unit vector opposite to the gravity). The via-point is finally defined as

$$P_{i,v} = P_{i,g} + \eta\, l\, \vec{u} + h\, \vec{v}, \quad \eta \in [0,1]\,. \tag{4.21}$$

A typical choice of the parameter η is $\eta = 1/2$. See Fig. 4.2.

Furthermore, we impose that the swing point p_i reaches its goal $P_{i,g}$ at time $t = (i+1)\,T$ with zero velocity, and that it reaches its via-point $P_{i,v}$ at time $t = i\,T + T_v$ with a zero $\vec{v}$-component (normal) velocity.

All these objectives are formulated as *target objectives*.

4.5.2. Keeping Balance: Controlling the CoM. The balance of the simulated robot is controlled through simple strategies, depending on whether we are adding or removing a contact. If step i adds a contact from then, following the *transition configurations* condition, the whole motion has to be performed by staying balanced on the initial stance σ_i, so the objective for the CoM in this case is a *set-point objective* that regulates its position around its position at the start configuration q_i. If step i removes a contact, then the robot has to "transfer its weight" from stance σ_i to stance σ_{i+1} in time T. For this purpose a *target objective* is defined for the CoM to reach at time $(i+1)\,T$ its position computed at the goal configuration q_{i+1}, with zero velocity.

4.5.3. Solving the Redundancy: Controlling the Configuration. The remaining redundancies are solved by controlling the whole configuration of the robot, with once again different strategies when adding or removing a contact. When adding a contact at step i, the posture is controlled through a set-point objective with the reference posture being set at $q_{\text{ref}} = q_i$ for the time interval $t \in [i\,T, i\,T + T_v]$ and set at $q_{\text{ref}} = q_{i+1}$ for the time interval $t \in [i\,T + T_v, (i+1)\,T]$ with low stiffness κ_p. When removing a contact then the reference configuration for the low-stiffness set-point objective is set at $q_{\text{ref}} = q_{i+1}$ during the whole step time interval $t \in [i\,T, (i+1)\,T]$.

4.5.4. Putting it Altogether: the FSM. As a summary of this section, Fig. 4.3 shows a graphical representation of the FSM. Details of the objectives are found in the previous subsections. The initial configuration of the robot at time $t = 0$ is $q = q_0 = q_s$ with $\dot{q} = 0$.

4.6. Playback Simulation Results

The video available at [**Bou**] shows some example applications of the proposed approach. These examples are: a basic walk motion, a single stair climbing motion, a multiple stairs climbing motion, a sitting motion, a one-step walk-on-hands motion. See Fig. 4.4 for snapshots of this video.

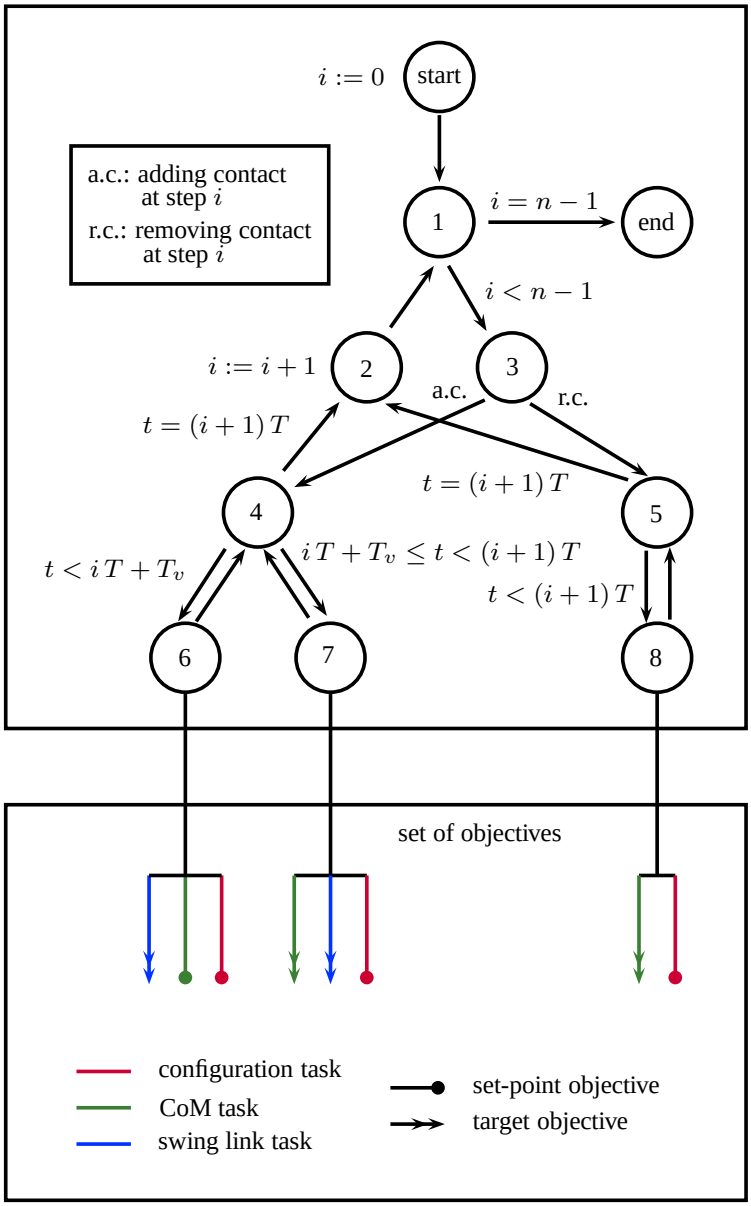
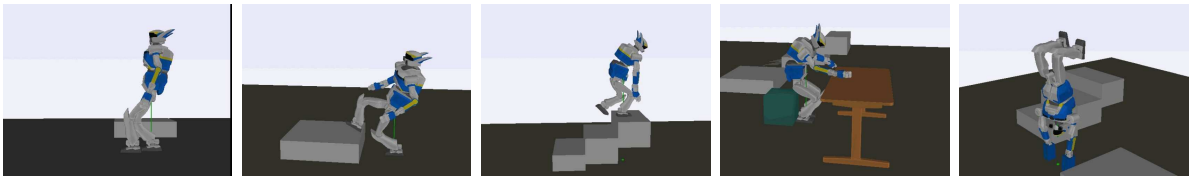


FIGURE 4.3. The finite-state machine (note: contains color information). States are represented as circles (the numbers inside have no particular meaning) and transitions as arrows between states. Labels next to transitions are the conditions for the transitions to be triggered. Transitions without labels are automatically triggered (condition always true). Labels next to states, when present, are actions performed when the machine reaches the states.

4.6.1. Experimental Framework. The humanoid robot model used is HRP-2 [**KKK+04**] with some modifications in terms of torque limits and arm links for the walk-on-hand motion, though our implementation is transparent to the particular robot model. The simulator used is described in [**CMK+06**] and the multi-contact static stances planner in Chapter 2. Collision detection between the robot and the environment is performed



(a) basic walk (b) single stair climbing (c) multiple stairs climbing (d) sitting (e) one step on hands

FIGURE 4.4. Snapshots from the on-line video

TABLE 4.1. motion generation parameters

	walk	single stair	multiple stairs	sitting	hands		all scenarios
number of steps n	10	6	8	3	2	w_{config}	10^1
step duration T	0.8 s	1.5 s	4 s	4 s	4 s	w_{CoM}	10^4
step height h	1 cm	30 cm/10 cm	55 cm/30 cm	10 cm/0 cm	10 cm	w_{slink}	10^3
parameter T_v	0.4 s	0.75 s	2 s	2 s	2 s	$\kappa_{p,\text{config}}$	10^1
parameter η	0.5					$\kappa_{p,\text{CoM}}$	10^3

using the PQP proximity queries package [**LGLM00**], and the QP solver used for multi-objective control is the QL convex quadratic programming solver [**Sch07**]. Table 4.1 gives the parameters used for these motions.

The video starts by showing elementary motions (single steps) produced by the multi-objective controller with fixed objectives. Then the five above-mentioned motions generated by coupling the multi-objective controller with the finite-state machine are sequentially played. Each of the five motions starts by first showing the output of the multi-contact stances planner used as input for that motion, i.e. the finite sequence of static postures $(q_0, \ldots, q_{n-1})$.

4.6.2. Discussion and Limitations. The motions displayed on this video have not been generated in real time. We used a time step of 1 ms for the simulator, but each iteration of the motion generator cycle took approximately 30 ms to compute on our 3 GHz Pentium IV system. However, real-time on-line control is not, at this stage, the main preoccupation of our work, so no particular effort has been devoted to reducing this computation time in our prototype implementation. Still, as a motion generation tool, the method is much faster than global motion optimization techniques [**LMKY10**].

Another limitation that currently prevents our method from being used as such as a control tool for the real robot is the absence of self-collision checking in the simulation (walk scenario), and joint limits constraints (single stair scenario). A basic strategy to reduce self-collision occurrences and to stay within joint limits that we implemented is the introduction of repulsive torques that are activated when a joint comes too close to its limit, but this does not absolutely guarantee that the limits are not reached.

An interesting feature that appears in these motions is the robustness to collisions with the environment and to uncertainty with regard to contact locations. In particular, we can see that when climbing the stairs, the swing foot can slightly collide with the stair but the robot does not lose balance. Also, even if the contact is not precisely put at its planned position this does not prevent the motion from being successfully carried out on

the stance including that contact. These remarks are encouraging in the perspective of later using the method for the control of the real robot.

There were cases however in which the collision of the swing link with the environment led to an impact from which the robot could not recover and ended up falling down. A posteriori tuning of the CoM objectives weights and gains sometimes enabled to regenerate a stable motion.

"Falling down" is what happens when the constraints of the QP (4.19) cannot be satisfied. This means that the robot reached a state $(q, \dot{q})$ outside of the *viability kernel* [**Wie02**]. If we had used a prioritization approach, this would have led to either a dynamically feasible motion that breaks the contacts, or a non-dynamically-consistent motion that maintains the contacts, both cases resulting in an ill-posed QP formulation in the subsequent simulation step. This is why we did not see the necessity to use prioritized formulation, and the motion generation fails ("crashes") in case the robot reaches such a non-viable state. Recovery strategies from these situations should be further investigated. Note however that falling down can also occur while all the constraints are satisfied, since the CoM control strategy does not strictly quantify the "stability keeping" notion that is not well defined outside a ZMP-applicable framework (e.g. [**Wie02**] for a discussion, and [**HHH+06**] for an approach of such notion of stability beyond the ZMP criterion).

4.7. Conclusion

We investigated a method inspired from computer graphics animation to generate simulated humanoid robot motion. This method allows the robot to benefit from full autonomy from the multi-contact planning stage to the motion generation stage, and thus the two stages of the contact-before-motion framework are achieved.

A number of issues have to be addressed to convert this motion generation tool into an on-line control tool. Most important is reaching real-time performance. Collision avoidance constraints might be included in the QP formulation by using repulsive potential field approaches.

We are also studying extendibility to problems such as object manipulation and multiple robot collaboration, as our generic multi-contact stances planner can handle these.

Another possible improvement worth investigating is to add to the framework a reduced-model planning phase that would produce more dynamic motions.

We now go back to the static problem solved in Chapter 3 and we try to extend it to the case of deformable contact support in the next chapter (Chapter 5).

4.8. Appendix: Dynamics Equation

Our objective in this appendix section is to recall the work published in [**Wie05**] and apply it to our particular root-link-quaternion-parametrized humanoid robot. So we suppose we have a humanoid robot made of r revolute joints indexed by $j \in \{1, \dots, r\}$ and $m+1$ bodies indexed by $k \in \{0, \dots, m\}$. The root body is the body 0. On each body k a set of contact forces $f_{k,1}, \dots, f_{k,m_k}$ are applied at the respective local-frame-expressed points $a_{k,1}, \dots, a_{k,m_k}$. Let $q = (x_0, \theta_0, \hat{q}) \in \mathbb{R}^{3+4+r}$ denote the configuration vector of the humanoid robot, where x_0 is the global-frame-expressed position of the root, θ_0 a parametrization of its orientation (a unit quaternion for instance), and $\hat{q}$ the internal joint angles vector.

Let R_k denote the orientation of the body k and x_k the global-frame-expressed position of the origin of the local frame k; in particular, R_0 is the orientation of the root body. Let ω_0^{global} and ω_0^{root} denote, respectively, the global-frame-expressed and the root-frame-expressed rotational velocity of the root body. Also, let $J_{\omega 0}^{\text{global}}$ and $J_{\omega 0}^{\text{root}}$ be respectively the mappings

$$\omega_0^{\text{global}} = J_{\omega 0}^{\text{global}}\,\dot{\theta}_0\,, \tag{4.22}$$

$$\omega_0^{\text{root}} = J_{\omega 0}^{\text{root}}\,\dot{\theta}_0\,. \tag{4.23}$$

If R_{01},R_{02},R_{03} denote the three columns of R_0, then we have

$$J_{\omega 0}^{\text{global}} = R_{03}R_{02}^T\frac{\partial R_{01}}{\partial\theta_0} + R_{02}R_{01}^T\frac{\partial R_{03}}{\partial\theta_0} + R_{01}R_{03}^T\frac{\partial R_{02}}{\partial\theta_0}\,, \tag{4.24}$$

and

$$J_{\omega 0}^{\text{root}} = R_0^T J_{\omega 0}^{\text{global}}\,. \tag{4.25}$$

We denote $J_{tk}(p)$, J_{rk}^{local}, and J_{rk}^{global}, respectively, the translational Jacobian at a local-frame-expressed point p, the local-frame-expressed rotational Jacobian, and the global-frame-expressed rotational Jacobian of the body k with respect to q. Similarly, $\hat{J}_{tk}(p)$, $\hat{J}_{rk}^{\text{local}}$, and $\hat{J}_{rk}^{\text{root}}$ are, respectively, the translational Jacobian at a local-frame-expressed point p, the local-frame-expressed rotational Jacobian, and the root-frame-expressed rotational Jacobian of the body k relative to the root body with respect to $\hat{q}$.

We have, for the translational Jacobian,

$$J_{tk}(p) = \left[\mathbf{1}_{3\times 3}\middle| -(x_k + R_k\,p - x_0)\times R_0\,J_{\omega 0}^{\text{root}}\middle| R_0\,\hat{J}_{tk}(p)\right]. \tag{4.26}$$

Remark: if we denote $p^0 = R_0^T(x_k + R_k\,p - x_0)$ as the root-frame-expressed position of p, which does not depend on θ_0, then we can show that

$$-(x_k + R_k\,p - x_0)\times R_0 J_{\omega 0}^{\text{root}} = -[R_0\,p^0\times]J_{\omega 0}^{\text{global}} \tag{4.27}$$

$$= \frac{\partial R_0\,p^0}{\partial\theta_0} \tag{4.28}$$

$$= \left[\frac{\partial R_0}{\partial\theta_{0,i}}\,p^0\right]_i. \tag{4.29}$$

We also have the following expression for the rotational Jacobian

$$J_{rk}^{\text{local}} = \left[\mathbf{0}_{3\times 3}\middle| R_k^T R_0\,J_{\omega 0}^{\text{root}}\middle| \hat{J}_{rk}^{\text{local}}\right], \tag{4.30}$$

$$J_{rk}^{\text{global}} = R_k\,J_{rk}^{\text{local}} = \left[\mathbf{0}_{3\times 3}\middle| J_{\omega 0}^{\text{global}}\middle| R_0\,\hat{J}_{rk}^{\text{root}}\right]. \tag{4.31}$$

Remark 1: The latter expressions is consistent with the composition rule of rotational velocities

$$\omega_k^{\text{global}} = J_{rk}^{\text{global}}\dot{q} \tag{4.32}$$

$$= J_{\omega 0}^{\text{global}}\dot{\theta}_0 + R_0(\,\hat{J}_{rk}^{\text{root}}\hat{\dot{q}}) \tag{4.33}$$

$$= \omega_0^{\text{global}} + R_0\,\omega_{k/\text{root}}^{\text{root}}\,. \tag{4.34}$$

Remark 2: We can easily check that the two following transport formulas are consistent with the above derivations of the translational and rotational Jacobians:

$$J_{tk}(a) = J_{tk}(b) - R_k[(a-b)\times]J_{rk}^{\text{local}}\,, \tag{4.35}$$

$$\hat{J}_{tk}(a) = \hat{J}_{tk}(b) - R_0^T R_k[(a-b)\times]\hat{J}_{rk}^{\text{local}}\,. \tag{4.36}$$

Let c_k denote the local-frame-expressed center of mass of the body k, μ_k its mass, and $I_k(p)$ its inertia matrix expressed at a local-frame-expressed point p, we have:

$$I_k(c_k) = I_k(\mathbf{0}_3) - \mu_k\left((c_k^T c_k)\,\mathbf{1}_{3\times 3} - c_k \times c_k\right)\,. \tag{4.37}$$

The motion of the humanoid robot is governed by the following equation

$$M(q)\ddot{q} + N(q,\dot{q})\dot{q} = M(q)\begin{pmatrix} g \\ \mathbf{0}_4 \\ \mathbf{0}_r \end{pmatrix} + \begin{pmatrix} \mathbf{0}_3 \\ \mathbf{0}_4 \\ u \end{pmatrix} + \sum_{k=0}^{m}\sum_{i=1}^{m_k} J_{tk}(a_{k,i})^T f_{k,i}\,, \tag{4.38}$$

$u \in \mathbb{R}^r$ being the joint actuators torques and g the gravity vector. The expressions of $M(q)$ and $N(q,\dot{q})$ are given as follows:

$$M(q) = \sum_{k=0}^{m} J_{tk}(c_k)^T \mu_k J_{tk}(c_k) + {J_{rk}^{\text{local}}}^T I_k(c_k) J_{rk}^{\text{local}}\,, \tag{4.39}$$

$$N(q,\dot{q}) = \sum_{k=0}^{m} J_{tk}(c_k)^T \mu_k \dot{J}_{tk}(c_k) + {J_{rk}^{\text{local}}}^T I_k(c_k)\dot{J}_{rk}^{\text{local}} - {J_{rk}^{\text{local}}}^T\left(I_k(c_k)J_{rk}^{\text{local}}\dot{q}\right)\times J_{rk}^{\text{local}}\,. \tag{4.40}$$

CHAPTER 5

FEM-based Static Posture Planning for a Humanoid Robot on Deformable Contact Support

In this chapter, after reaching the dynamics-consistent motion in the previous chapter (Chapter 4), we go back to statics and we extend the work presented in Chapter 3, ie. solving the inverse kinematics problem for a humanoid robot in general multi-contact stances under physical limitations and static equilibrium constraints, to the case in which the contact is made on a non-rigid deformable environment support. We take a finite element approach to solve the static equilibrium equations for the system made of the robot and the deformable support within the linear elasticity model. Example simulation results show the humanoid robot HRP-2 taking contact support with hand or foot link on a deformable cube.

5.1. Introduction

In Chapter 3 we presented an optimization-based solution for the inverse kinematics problem on non-horizontal non-coplanar frictional multi-contact stances for a humanoid robot subject to joint and torque limits under the static equilibrium constraint. This work is used within a contact-before-motion planning framework presented in Chapter 2 that extended the seminal works of [**EKM06, HBL05**] to general multi-agent systems for solving indifferently locomotion and manipulation planning problems centred around the humanoid robot.

One common assumption in all of these works is the rigidity hypothesis, for both the robot links and the environment objects. Our aim in this work is to further extend the capabilities of these contact-before-motion planners to cope with deformable objects in the environment under the linear elasticity hypothesis. This can be made possible if the underlying inverse kinematics solver under static equilibrium constraint can deal with such linear elasticity models. Thus we focus on this latter task, extending the solver presented in Chapter 3 to the case in which the contact prints are positioned on a surface belonging to a deformable object in the environment. For other planning problems involving deformable objects, we can cite for example the works [**SI07, vdBMGA10**].

The approach we choose to solve for the static equilibrium equations of the elastic material is based on the finite element method. The deformation of the contact support is related to the corresponding position of the supported link of the robot and is as such a function of the configuration of the robot. This deformation generates reaction forces that have to be taken into account in the equilibrium equation of the robot. The main contribution of this work is thus to relate the induced deformation forces to the configuration of the robot in a way that will allow us to derive the gradient of the extended static equilibrium constraint fed to the non-linear constrained optimization solver.

The rest of the chapter is organized as follows. In Section 5.2 we introduce the notations used by recalling the finite element method for linear elasticity models. We then write the constraint and its gradient in Section 5.3 which constitutes the main development of the chapter. Example applications are presented in Section 5.4, before concluding the chapter by discussing limitations and perspectives in Section 5.5.

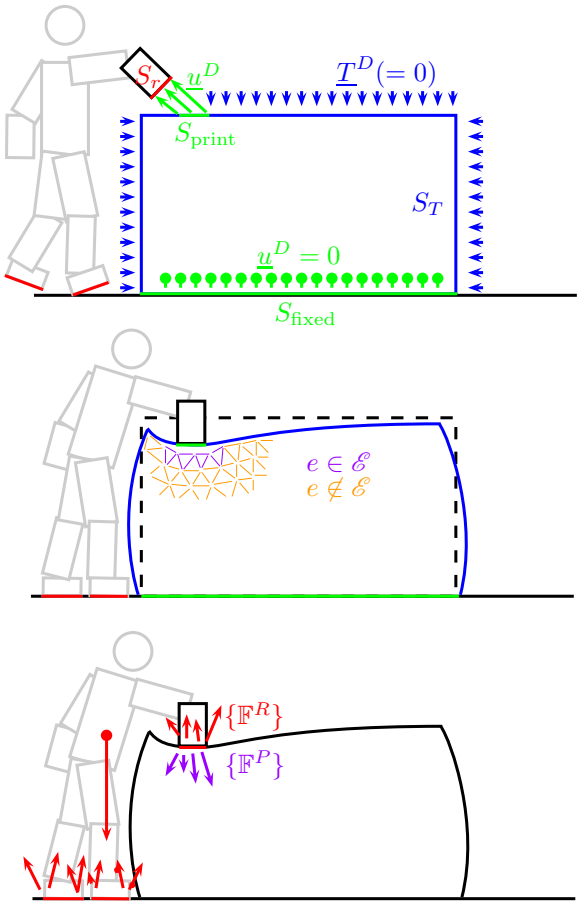


FIGURE 5.1. Overview illustration of the method.

5.2. The Finite Element Method

We first recall the finite element method we use to formulate and solve the problem. This section is mainly adapted from the reference textbook [**BF05**] that we reproduce here in order to introduce the notations that we need for the sake of our formulation.[1]

So let us consider a solid object that occupies in the Euclidean space a volume denoted Ω subject to the behaviour model of linear elasticity under the small deformation

[1]We encourage the reader familiar with the method to still go through this section as a minimum requirement to understand the notations and reasoning of the rest of the chapter.

hypothesis. The following equations govern the static equilibrium of the object:

$$\underline{\underline{\varepsilon}}(\underline{x}) = \frac{1}{2}\left(\nabla \underline{u} + \nabla^T \underline{u}\right)(\underline{x})\,, \tag{5.1}$$

$$\operatorname{div} \underline{\underline{\sigma}}(\underline{x}) + \rho\, \underline{f}(\underline{x}) = 0\,, \tag{5.2}$$

$$\underline{\underline{\sigma}}(\underline{x}) = \underline{\underline{\underline{\underline{\mathcal{A}}}}} : \underline{\underline{\varepsilon}}(\underline{x})\,, \tag{5.3}$$

where $\underline{x} \in \Omega$, $\underline{\underline{\varepsilon}}$ is the strain tensor field, $\underline{u}$ is the displacement field, $\underline{\underline{\sigma}}$ is the Cauchy stress tensor field, $\underline{\underline{\underline{\underline{\mathcal{A}}}}}$ is the elasticity tensor, ρ is the mass density of the material, $\underline{f}$ is the body force density field. The boundary conditions for a well-posed problem are specified as:

$$\underline{\underline{\sigma}}(\underline{x}).\underline{n}(\underline{x}) = \underline{T}^D(\underline{x}) \quad (\underline{x} \in S_T)\,, \tag{5.4}$$

$$\underline{u}(\underline{x}) = \underline{u}^D(\underline{x}) \quad (\underline{x} \in S_u)\,, \tag{5.5}$$

where $\underline{T}^D$ and $\underline{u}^D$ are respectively the prescribed surface force density (traction) and prescribed displacement fields on the the surfaces S_T and S_u that constitute a partition of the frontier $\partial\Omega$ ($S_T \cap S_u = \varnothing$ and $S_T \cup S_u = \partial\Omega$), and $\underline{n}(\underline{x})$ is the unit normal to the surface $\partial\Omega$ at the point $\underline{x}$.

By applying the virtual work principle, or by following a variational approach minimizing potential energy, we can get to the weak formulation of the problem, in which we look for a displacement field $\underline{u}$ satisfying (5.5) such that:

$$\int_\Omega \underline{\underline{\varepsilon}}[\underline{u}] : \underline{\underline{\underline{\underline{\mathcal{A}}}}} : \underline{\underline{\varepsilon}}[\underline{w}]\, dV = \int_\Omega \rho \underline{f}.\underline{w}\, dV + \int_{S_T} \underline{T}^D.\underline{w}\, dS\,, \tag{5.6}$$

for all the virtual displacement fields $\underline{w}$ that are zero on the surface S_u, where we have used the notation

$$\underline{\underline{\varepsilon}}[\underline{v}] = \frac{1}{2}\left(\nabla \underline{v} + \nabla^T \underline{v}\right). \tag{5.7}$$

We approximate the domain Ω by a domain $\Omega_h = \cup_e E_e$ ($1 \le e \le N_E$) made of isoparametric elements E_e of characteristic dimension h (the subscript h will be used to make distinction between the exact problem and the approximated problem) that constitute a mesh of Ω_h, the nodes of which are denoted $\underline{x}^{(n)}$ (globally within the whole mesh, or $\underline{x}_e^{(k)}$, $1 \le k \le n_e$, locally within each element e). The position of a point $\underline{x} \in E_e$ is interpolated from the positions of the nodes of the element using the local shape functions N_k

$$\underline{x} = \sum_{k=1}^{n_e} N_k(\underline{a})\underline{x}_e^{(k)}\,, \tag{5.8}$$

where $\underline{a}$ is a parameter varying in a reference non-deformed "unit" element Δ_e, and we choose to interpolate accordingly an arbitrary displacement field $\underline{v}_h$ on the nodal displacements $\underline{v}^{(k)}$ using the same interpolation

$$\underline{v}_h = \sum_{k=1}^{n_e} N_k(\underline{a})\underline{v}^{(k)}\,. \tag{5.9}$$

We also introduce an injective index function $\mathrm{dof}(n, j)$ such that $\mathrm{dof}(n, j) > 0$ if the coordinate j (along the basis vector $\underline{e}_j$) of the node $\underline{x}^{(n)}$ is free, meaning that the node $\underline{x}^{(n)}$ does not belong to the prescribed-displacement surface $S_{u,h} \subset \partial\Omega_h$, and $\mathrm{dof}(n, j) \le 0$ otherwise.

Finally we apply the Galerkin method. We look for a displacement field of the form

$$\underline{u}_h(\underline{x}) = \underline{u}_h^{(D)}(\underline{x}) + \underline{u}_h^{(0)}(\underline{x})\,, \tag{5.10}$$

where the fields $\underline{u}_h^{(D)}$ and $\underline{u}_h^{(0)}$, respectively satisfying the boundary condition (5.5) and vanishing on the surface $S_{u,h}$, are interpolated as

$$\underline{u}_h^{(D)}(\underline{x}) = \sum_{(n,j)|\mathrm{dof}(n,j)\leq 0} \tilde{N}_n(\underline{x})\, u_j^{(D)}(\underline{x}^{(n)})\, \underline{e}_j\,, \tag{5.11}$$

$$\underline{u}_h^{(0)}(\underline{x}) = \sum_{(n,j)|\mathrm{dof}(n,j)>0} \tilde{N}_n(\underline{x})\, u_j^{(n)}\, \underline{e}_j\,, \tag{5.12}$$

where $\tilde{N}_n$ are the global shape functions constructed from the functions N_k in (5.8) so as to represent the position of a point expressed in the whole domain Ω_h. The virtual displacement field defined in the weak formulation (5.6) takes the form

$$\underline{w}(\underline{x}) = \sum_{(n,j)|\mathrm{dof}(n,j)>0} \tilde{N}_n(\underline{x})\, w_j^{(n)}\, \underline{e}_j\,, \tag{5.13}$$

and the weak formulation (5.6) amounts now to finding a field $\underline{u}_h^{(0)}$ of the form (5.12) such that for every field $\underline{w}$ of the form (5.13) we have

$$\begin{aligned}\int_{\Omega_h} \underline{\underline{\varepsilon}}[\underline{u}_h^{(0)}] : \underline{\underline{\underline{\underline{\mathcal{A}}}}} : \underline{\underline{\varepsilon}}[\underline{w}]\, dV = &- \int_{\Omega_h} \underline{\underline{\varepsilon}}[\underline{u}_h^{(D)}] : \underline{\underline{\underline{\underline{\mathcal{A}}}}} : \underline{\underline{\varepsilon}}[\underline{w}]\, dV \\ &+ \int_{\Omega_h} \rho \underline{f}.\underline{w}\, dV + \int_{S_{T,h}} \underline{T}^D.\underline{w}\, dS\,.\end{aligned} \tag{5.14}$$

By gathering the free nodes displacements coordinates $u_j^{(n)}$ and $w_j^{(n)}$ ($\mathrm{dof}(n,j) > 0$) respectively in the vectors $\{\mathbb{U}^F\}$ and $\{\mathbb{W}\}$, the formulation (5.14) takes the following linear system form

$$\{\mathbb{W}\}^T[\mathbb{K}^F]\{\mathbb{U}^F\} = \{\mathbb{W}\}^T\{\mathbb{F}\}\,, \tag{5.15}$$

or equivalently

$$[\mathbb{K}^F]\{\mathbb{U}^F\} = \{\mathbb{F}\}\,, \tag{5.16}$$

where the rigidity matrix $[\mathbb{K}^F]$ and generalized nodal forces $\{\mathbb{F}\}$ are defined as sums of elementary integrals over the elements E_e through identification respectively in

$$\{\mathbb{W}\}^T[\mathbb{K}^F]\{\mathbb{U}^F\} = \sum_{e=1}^{N_E} \int_{E_e} \underline{\underline{\varepsilon}}[\underline{u}_h^{(0)}] : \underline{\underline{\underline{\underline{\mathcal{A}}}}} : \underline{\underline{\varepsilon}}[\underline{w}]\, dV\,, \tag{5.17}$$

and

$$\{\mathbb{W}\}^T\{\mathbb{F}\} = \sum_{e=1}^{N_E} - \int_{E_e} \underline{\underline{\varepsilon}}[\underline{u}_h^{(D)}] : \underline{\underline{\underline{\underline{\mathcal{A}}}}} : \underline{\underline{\varepsilon}}[\underline{w}]\, dV + \int_{E_e} \rho \underline{f}.\underline{w}\, dV + \int_{S_{T,h}\cap E_e} \underline{T}^D.\underline{w}\, dS\,. \tag{5.18}$$

The Voigt Notation. Let us differentiate the two relations (5.8) and (5.9)

$$d\underline{x} = \underline{\underline{J}}(\underline{a})\, d\underline{a}\,, \tag{5.19}$$

$$d\underline{v}_h = \underline{\underline{H}}(\underline{a})\, d\underline{a}\,. \tag{5.20}$$

This allows us to rewrite the relation (5.7) applied to the field $\underline{v}_h$ as

$$\underline{\underline{\varepsilon}}[\underline{v}_h](\underline{x}) = \frac{1}{2}\left(\underline{\underline{H}}(\underline{a}).\underline{\underline{J}}^{-1}(\underline{a}) + \left(\underline{\underline{H}}(\underline{a}).\underline{\underline{J}}^{-1}(\underline{a})\right)^T\right). \tag{5.21}$$

By introducing the Voigt representations of the symmetric tensors $\underline{\underline{\varepsilon}}$ and $\underline{\underline{\sigma}}$, which are the $\mathbb{R}^6$ vectors containing the 6 independents components of the two tensors

$$\{\sigma\} = \{\sigma_{11}\ \sigma_{22}\ \sigma_{33}\ \sigma_{12}\ \sigma_{13}\ \sigma_{23}\}^T\,, \tag{5.22}$$

$$\{\varepsilon\} = \{\varepsilon_{11}\ \varepsilon_{22}\ \varepsilon_{33}\ 2\,\varepsilon_{12}\ 2\,\varepsilon_{13}\ 2\,\varepsilon_{23}\}^T\,, \tag{5.23}$$

the relation (5.21) takes the form

$$\{\varepsilon\} = [B_e(\underline{a})]\{\mathbb{V}_e\}\,, \tag{5.24}$$

where $\{\mathbb{V}_e\}$ is the $\mathbb{R}^{3n_e}$ vector concatenating the nodal displacements $\underline{v}^{(k)}$ and $[B_e(\underline{a})]$ is a $6 \times 3n_e$ matrix obtained through identification in (5.21). Moreover, the relation (5.3) simplifies into

$$\{\sigma\} = [A]\{\varepsilon\}\,, \tag{5.25}$$

where $[A]$ is the 6×6 matrix written in terms of the Lamé parameters λ and μ for an isotropic homogeneous material

$$[A] = \begin{bmatrix} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{bmatrix}. \tag{5.26}$$

These relations allows for a simple evaluation of the elementary integrals in (5.17) and (5.18) using a Gauss-point-based numerical method.

5.3. Formulation of the Planning Problem

Let us now consider the problem of a humanoid robot in multi-contact stance with its environment, in which one of the contacts (we will refer to it as the "deformable contact") is made on the surface of the deformable object introduced in the previous section. See Fig. 5.1. Let the corresponding contact surface on the robot be denoted S_r, which is a planar surface defined on a link l of the robot r. The desired relative position and orientation of the deformable contact $(x, y, \theta) \in SE(2)$ define a *contact print*, that is the image of the surface S_r projected onto $\partial\Omega$ and positioned according to (x, y, θ). Let this contact print be denoted $S_{\text{print}} \subset \partial\Omega$, and the corresponding bijective projection mapping $p_{\text{print}} : S_r \to S_{\text{print}}$, which is simply a rigid transformation. Furthermore, a portion S_{fixed} of the frontier $\partial\Omega$ is fixed on the environment, for instance the base of the deformable

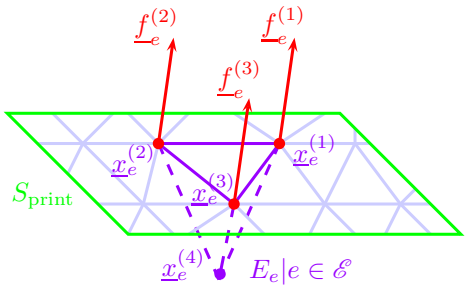


FIGURE 5.2. Nodal reaction forces.

object contacting the rigid floor. Let $\mathscr{P}$ and $\mathscr{F}$ be the subsets of the surface nodes of the mesh $\underline{x}^{(n)}$ that lie inside S_{print} and S_{fixed} respectively

$$\mathscr{P} = \{\underline{x}^{(n)} \mid \underline{x}^{(n)} \in S_{\text{print}}\}\,, \tag{5.27}$$

$$\mathscr{F} = \{\underline{x}^{(n)} \mid \underline{x}^{(n)} \in S_{\text{fixed}}\}\,. \tag{5.28}$$

The prescribed-displacement surface in the boundary condition (5.5) is in this case $S_u = S_{\text{print}} \cup S_{\text{fixed}}$, and the prescribed nodal displacements are

$$\underline{u}^{(D)}(\underline{x}^{(n)}) = \begin{cases} 0 & \text{if } \underline{x}^{(n)} \in \mathscr{F}, \\ p_{\text{print}}^{-1}(\underline{x}^{(n)}) - \underline{x}^{(n)} & \text{if } \underline{x}^{(n)} \in \mathscr{P}. \end{cases} \tag{5.29}$$

On the remaining surface $S_T = \partial\Omega \setminus (S_{\text{print}} \cup S_{\text{fixed}})$ the prescribed traction is set to zero

$$\underline{T}^D(\underline{x}) = 0 \quad (\underline{x} \in S_T)\,, \tag{5.30}$$

and the body force density is also set to zero

$$\underline{f}(\underline{x}) = 0 \quad (\underline{x} \in \Omega)\,. \tag{5.31}$$

In these conditions, by concatenating the prescribed nodal displacements $\underline{u}^{(D)}(\underline{x}^{(n)})$ into the vector $\{\mathbb{U}^D\}$, the nodal forces vector (5.18) takes the form

$$\{\mathbb{F}\} = -[\mathbb{K}^D]\{\mathbb{U}^D\}\,, \tag{5.32}$$

where the matrix $[\mathbb{K}^D]$ is defined through identification in

$$\{\mathbb{W}\}^T[\mathbb{K}^D]\{\mathbb{U}^D\} = \sum_{e=1}^{N_E} \int_{E_e} \underline{\underline{\varepsilon}}[\underline{u}_h^{(D)}] : \underline{\underline{\underline{\underline{\mathcal{A}}}}} : \underline{\underline{\varepsilon}}[\underline{w}]\, dV\,. \tag{5.33}$$

Finally equation (5.16) reduces to

$$[\mathbb{K}^F]\{\mathbb{U}^F\} + [\mathbb{K}^D]\{\mathbb{U}^D\} = 0\,, \tag{5.34}$$

which can be rewritten as

$$\{\mathbb{U}\} = \begin{Bmatrix} \mathbb{U}^F \\ \mathbb{U}^D \end{Bmatrix} = \begin{bmatrix} -[\mathbb{K}^F]^{-1}[\mathbb{K}^D] \\ I \end{bmatrix} \{\mathbb{U}^D\}\,, \tag{5.35}$$

the vector $\{\mathbb{U}\}$ containing now the displacements of all the nodes of the mesh, and I being the identity matrix of dimension $\dim(\{\mathbb{U}^D\})$. We rewrite this latter equation in a more compact form

$$\{\mathbb{U}\} = [\mathbb{K}]\{\mathbb{U}^D\}\,. \tag{5.36}$$

Nodal Reaction Forces. We would like now compute the nodal reaction forces $\{\mathbb{F}^R\}$ that are applied through the nodes of the contact print $\mathscr{P}$ on the contact link l of the robot r. See Fig. 5.2. First let us define what we mean by such nodal reaction forces. Let $\underline{T}^P$ be the traction that is applied on the the deformable object through the surface S_{print}. For a point $\underline{x} \in S_{\text{print}}$ we have

$$\underline{T}^P(\underline{x}) = \underline{\underline{\sigma}}(\underline{x}).\underline{n}(\underline{x})\,. \tag{5.37}$$

We approximate the the surface print S_{print} by $S_{\text{print},h}$ defined as the union of the surfaces of the elements that have all of their frontier nodes belonging to $\mathscr{P}$. These elements are members of the set

$$\mathscr{E} = \left\{ e \in \{1, \ldots, N_E\} \mid E_e \cap \partial\Omega_h \subset S_{\text{print}} \right\}, \tag{5.38}$$

and thus $S_{\text{print},h}$ is

$$S_{\text{print},h} = \bigcup_{e \in \mathscr{E}} E_e \cap \partial\Omega_h\,. \tag{5.39}$$

For every $e \in \mathscr{E}$, we would like to compute the traction $\underline{T}^P(\underline{x})$ when $\underline{x}$ varies in $E_e \cap \partial\Omega_h$. Since we chose to use tetrahedron elements, the matrix $[B_e(\underline{a})]$ defined in (5.24) can be shown to be independent of the parameter $\underline{a}$, $[B_e(\underline{a})] = [B_e]$, and the stress field $\underline{\underline{\sigma}}(\underline{x})$ is thus constant within every element E_e, $\underline{\underline{\sigma}}(\underline{x}) = \underline{\underline{\sigma}}_e$. Since $E_e \cap \partial\Omega_h$ reduces in this case to a planar triangle, the normal $\underline{n}(\underline{x})$ is constant throughout $E_e \cap \partial\Omega_h$, we denote it $\underline{n}_e$, and subsequently $\underline{T}^P(\underline{x})$ is also constant throughout $E_e \cap \partial\Omega_h$, we denote it $\underline{T}^P_e$. The nodal surface forces $\{\mathbb{F}^P_e\}$ are defined such that for every virtual displacement field $\underline{w}(\underline{x}), x \in E_e \cap \partial\Omega_h$ interpolating the nodal displacements $\{\mathbb{W}^P_e\}$ of the three surface triangle nodes through the interpolation (5.9) we have

$$\{\mathbb{W}^P_e\}^T\{\mathbb{F}^P_e\} = \int_{E_e \cap \partial\Omega_h} \underline{T}^P_e.\underline{w}\, dS\,. \tag{5.40}$$

If α_e denotes the area of the triangle $E_e \cap \partial\Omega_h$, we can show that identification in this latter relation leads to

$$\{\mathbb{F}^P_e\} = \frac{\alpha_e}{3}\, N\, \underline{\underline{\sigma}}_e.\underline{n}_e\,, \tag{5.41}$$

where N is a duplication matrix

$$N = \begin{bmatrix} I_{3\times3} & I_{3\times3} & I_{3\times3} \end{bmatrix}^T\,. \tag{5.42}$$

$\{\mathbb{F}^R_e\}$, the contribution of the element $e \in \mathscr{E}$ to the nodal reaction forces $\{\mathbb{F}^R\}$, is the opposite of this vector

$$\{\mathbb{F}^R_e\} = -\{\mathbb{F}^P_e\} = -\frac{\alpha_e}{3}\, N\, \underline{\underline{\sigma}}_e.\underline{n}_e\,, \tag{5.43}$$

The application points of $\{\mathbb{F}^R_e\}$ are the vertices $(\underline{x}^{(1)}_e, \underline{x}^{(2)}_e, \underline{x}^{(3)}_e)$ of the triangle $E_e \cap \partial\Omega_h$. Finally the reaction surface force distribution over the triangle $E_e \cap \partial\Omega_h$ is equivalent from a virtual work point of view to the set of three point forces

$$\left\{ \begin{array}{ll} \underline{f}^{(1)}_e = -\dfrac{\alpha_e}{3}\, \underline{\underline{\sigma}}_e.\underline{n}_e & \text{applied at } \underline{x}^{(1)}_e \\ \underline{f}^{(2)}_e = -\dfrac{\alpha_e}{3}\, \underline{\underline{\sigma}}_e.\underline{n}_e & \text{applied at } \underline{x}^{(2)}_e \\ \underline{f}^{(3)}_e = -\dfrac{\alpha_e}{3}\, \underline{\underline{\sigma}}_e.\underline{n}_e & \text{applied at } \underline{x}^{(3)}_e \end{array} \right\}. \tag{5.44}$$

The Optimization Approach. We recall now the approach followed in Chapter 3 for solving an inverse problem. if q denotes the configuration of the robot (including the free-flying base component in $SE(3)$) and Λ the set of non-negative coefficients along the linearised friction cone generators at the contact points, the approach consists in solving the non-linear constrained optimization problem of an arbitrary objective function obj[2]

$$\min_{(q,\Lambda)} \quad \text{obj}(q,\Lambda) \tag{5.45}$$

$$\text{under} \quad \text{joint limits}, \tag{5.46}$$

$$\text{torque limits}, \tag{5.47}$$

$$\text{friction cone}, \tag{5.48}$$

$$\text{and static equilibrium constraints}. \tag{5.49}$$

Taking into account the deformable contact is straightforward by adding the forces (5.44) to the set of contact forces applied on the robot in the formulation of the torque limits and static equilibrium constraints (5.47) and (5.49) of the formulation (5.45). One difficulty arises in computing the contribution of these forces to the gradient of these two constraints (5.47) and (5.49).

So let us consider one of these forces $\underline{f}_e^{(j)}$ ($e \in \mathscr{E}$ and $j \in \{1,2,3\}$) and try to explicit its dependency on the configuration of the robot q. We have

$$\underline{f}_e^{(j)}(q) = -\frac{\alpha_e}{3}\,\underline{\underline{\sigma}}_e(q).\underline{n}_e(q)\,. \tag{5.50}$$

Note that since the projection operator p_{print} is a rigid transformation the area of the element frontier triangle α_e is constant and does not depend on q. Let $[D]$ be the 9×6 duplication matrix

$$[D] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \tag{5.51}$$

such that we can write the definition of the Voigt notation (5.22) of the stress tensor $\{\sigma_e\}$ as a vectorization relation

$$[D]\,\{\sigma_e\} = \text{vec}(\underline{\underline{\sigma}}_e)\,, \tag{5.52}$$

where $\text{vec}(M)$ means the column vector obtained by concatenating all the columns of M into one column vector [**MN99**]. Since $\underline{\underline{\sigma}}_e.\underline{n}_e$ is already a column vector then its

[2]The objective function is designed in a way to minimize a distance to a reference posture and to optimize the repartition of contact forces or actuation torques. In the present case an additional weighted component aimed at minimizing the deformation can be added by minimizing the norm of the nodal reaction forces that will are derived in the subsequent development of the chapter. This is done in particular in the presented results at the end of the chapter.

vectorization is trivial

$$\mathrm{vec}(\underline{\underline{\sigma}}_e.\underline{n}_e) = \underline{\underline{\sigma}}_e.\underline{n}_e\,. \tag{5.53}$$

The algebra of the vectorization operation tells us that the vectorization of a matrix product can be derived using the Kronecker product operation $\otimes$

$$\mathrm{vec}(M_1\,M_2) = (M_2^T \otimes I_{k\times k})\,\mathrm{vec}(M_1)\,, \tag{5.54}$$

where k is the number of rows of M_1. So the relation (5.53) using (5.52) becomes

$$\underline{\underline{\sigma}}_e.\underline{n}_e = \mathrm{vec}(\underline{\underline{\sigma}}_e.\underline{n}_e)\,, \tag{5.55}$$

$$= (\underline{n}_e^T \otimes I_{3\times 3})\,\mathrm{vec}(\underline{\underline{\sigma}}_e)\,, \tag{5.56}$$

$$= (\underline{n}_e^T \otimes I_{3\times 3})\,[D]\,\{\sigma_e\}. \tag{5.57}$$

Moreover, from (5.24) and (5.25) we can write

$$\{\sigma_e\} = [A][B_e]\{\mathbb{U}_e\}\,, \tag{5.58}$$

where $\{\mathbb{U}_e\}$ are the nodal displacements of the four vertices of the element E_e, which can be obtained from (5.36) as

$$\{\mathbb{U}_e\} = [\mathbb{K}_e]\{\mathbb{U}^D\}\,, \tag{5.59}$$

$[\mathbb{K}_e]$ being the matrix extracted by keeping only the 12 rows of the matrix $[\mathbb{K}]$ corresponding to the 12 components $\{\mathbb{U}_e\}$ in $\{\mathbb{U}\}$. Finally we can rewrite an explicit expression of (5.50)

$$\underline{f}_e^{(j)}(q) = -\frac{\alpha_e}{3}\left(\underline{n}_e(q)^T \otimes I_{3\times 3}\right)[D][A][B_e][\mathbb{K}_e]\left\{\mathbb{U}^D(q)\right\}. \tag{5.60}$$

The gradient of (5.60) with respect to q can now be derived based on the two computationally available Jacobian matrices of the contact link l of the robot r

$$\frac{\partial \underline{n}_e(q)}{\partial q}\,, \tag{5.61}$$

$$\frac{\partial\left\{\mathbb{U}^D(q)\right\}}{\partial q}\,. \tag{5.62}$$

(Recall that at the solution $S_r = S_{\text{print}}$ and thus $\underline{n}_e(q)$ and the non-zero components of $\left\{\mathbb{U}^D(q)\right\}$ can be considered as rigidly attached to S_r ie. rigidly attached to the link l). This gradient takes the final form

$$\frac{\partial \underline{f}_e^j}{\partial q} = -\frac{\alpha_e}{3}\left[\left(\left[\frac{\partial \underline{n}_e}{\partial q_i}\right]^T \otimes I_{3\times 3}\right)[D][A][B_e][\mathbb{K}_e]\left\{\mathbb{U}^D\right\}\right]_{i=1}^{\dim(\mathrm{q})} - \frac{\alpha_e}{3}\left(\underline{n}_e^T \otimes I_{3\times 3}\right)[D][A][B_e][\mathbb{K}_e]\frac{\partial\left\{\mathbb{U}^D\right\}}{\partial q}\,. \tag{5.63}$$

The gradients of the moment of the force $\underline{f}_e^{(j)}(q)$ and the torques resulting from it follow directly using the Jacobians at the application points that can also be considered as being attached to the surface S_r and thus to the link l of the robot r

$$\frac{\partial \underline{x}_e^{(j)}(q)}{\partial q}\,. \tag{5.64}$$

Finally the computation of these gradients allows us to use non-linear optimization solvers such as [**LT96, WB06**] to solve the problem (5.45) taking into account the nodal reaction forces.

5.4. Simulation Results

We applied the presented method to an example scenario in which the humanoid robot HRP-2 [**KKK+04**] takes support with both feet on the rigid ground and with a modified hand link on a deformable object.

The deformable object is a simple 1m × 1m × 1m cube with an isoparametric mesh made of 166 nodes and 570 tetrahedron elements. See Fig. 5.3. Table 5.1 gives the physical properties of the material that constitutes the cube.

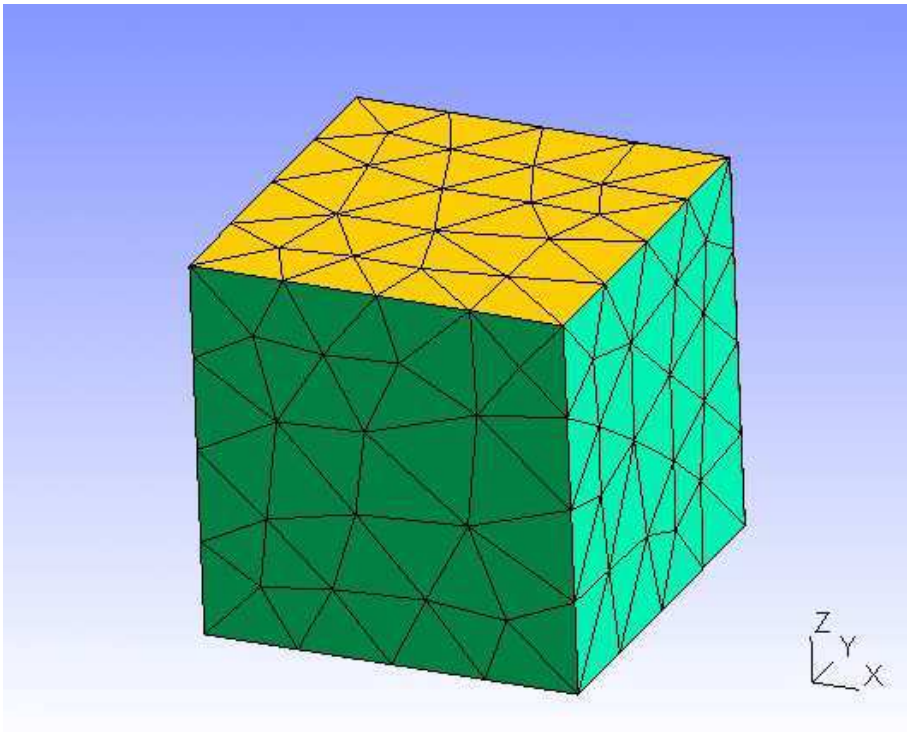


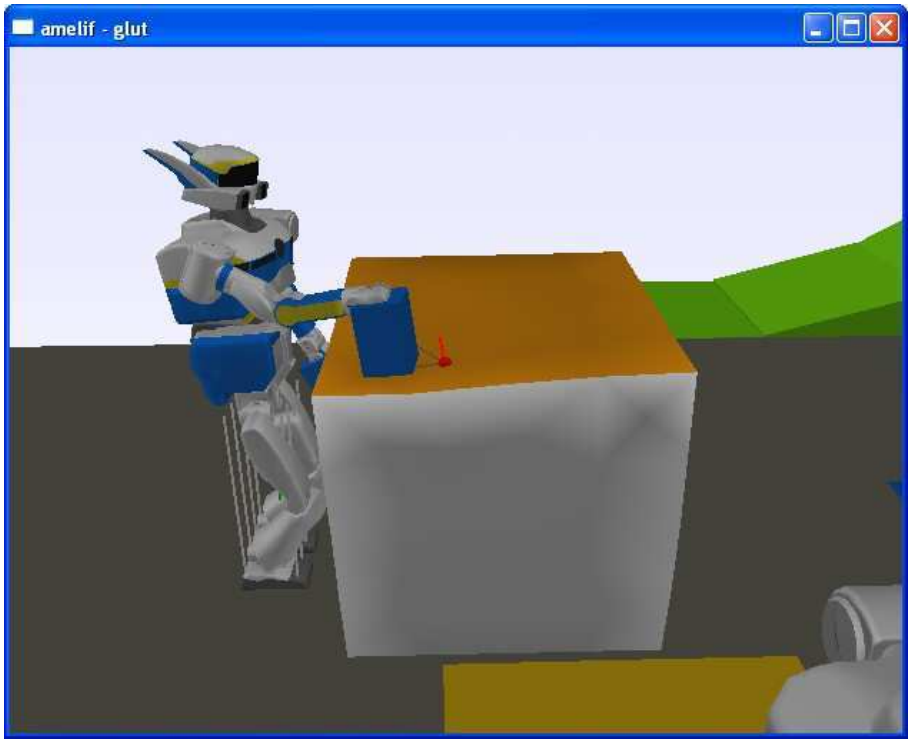
FIGURE 5.3. The mesh of the deformable environment contact support.

Young's modulus E	10^6 Pa
Poisson's ratio ν	0.4
Mass density ρ	10^3 kg/m^3

TABLE 5.1. Properties of the deformable material.

Fig. 5.4 shows the resulting configuration together with snapshots configurations along the optimisation iteration process. Note that we are only interested in the final iterate, the intermediate configurations do not have physical meaning.

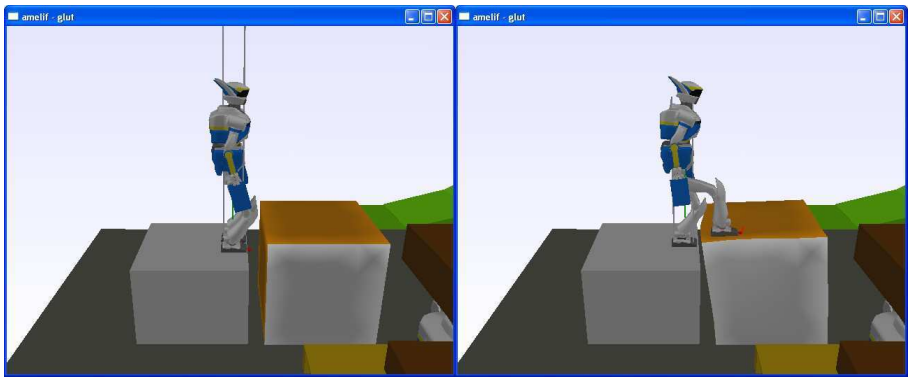
In another example scenario, shown in Fig. 5.5, the HRP-2 robot has its left foot supporting on a rigid object and its right foot supporting on the same deformable cube.



(a) Final result

(b) $i = 0/68$ (c) $i = 1/68$ (d) $i = 20/68$ (e) $i = 50/68$ (f) $i = 68/68$

FIGURE 5.4. Example of the execution of the optimisation algorithm. i is the iteration counter. The total number of iterations is 68.



(a) Non-deformed configuration (b) Deformed cube after taking a step

FIGURE 5.5. HRP-2 taking a step on the deformable cube.

Finally, Fig. 5.6 shows for the sake of visualization an on-purpose exaggerated deformation resulting from lower Young's modulus of the material constituting the cube. This configuration is not physically valid since the linear elasticity regime should be applied under the small deformation hypothesis, which occurs only in the first case. For large deformations, non-linear approaches such as [**BJ05, YC00**] should be investigated.

As for execution time, the orders of magnitude as reported in Chapter 3 range from one to ten seconds per query. Adding the FEM resolution step keeps it in the order of tens of seconds, without any effort devoted to reducing this time in our prototype implementation.

(a) $E = 10^6$Pa (b) $E = 5 \times 10^5$Pa

FIGURE 5.6. Resulting configuration with different Young's mudulus E.

5.5. Conclusion

We extended our multi-contact static posture planning optimisation framework to take into account non-rigid linear-elasticity-based deformable model as a possible contact support. The linear behaviour made it possible to derive the gradient of the nodal reaction forces with respect to the configuration of the robot, which defines the boundary conditions of the deformation.

One limitation of this approach resides in its non applicability to the planning of the whole sequence of postures in the framework of contact-before-motion planning in its continuous formulation as presented in Chapter 2, Section 2.3.2, Algorithm 2. The reason is that the position of the contact print (x, y, θ) on the deformable surface should be specified and fixed beforehand in the current approach. If we were to keep this position (x, y, θ) as an optimisation variable, then the reaction forces would not any more be continuous functions of the configuration since the set $\mathscr{E}$ of the finite elements belonging the non-fixed contact print S_{print} would vary in a discrete non-continuous way. Thus it is not possible to use a finite-element-based approach to plan for the sequence of postures under continuous search of the best positions of the contacts.

One way to overcome this limitation is to resort to a contact-before-motion planning approach in its pre-discretized contact positions formulation as in [**HBL05**] (Chapter 2, Section 2.3.1, Algorithm 1), where we pre-process the environment by sampling a finite set of possible contact positions (fixed) on the environment, in particular on the deformable support, and perform a discrete search along these sampled positions. The approach presented in this chapter is thus suitable in this case.

Finally, one remaining difficulty lies in the formulation of the collision-avoidance constraint with deformable objects.

Conclusion

In this PhD work we contributed to the general autonomy of humanoid robots by providing them with a tool that allows them to plan locomotion or manipulation motions changing their contact configurations.

We first needed to set up the desired level autonomy and, for that purpose, to identify the kind of objectives in terms of which we would like to specify a task to be realized by the autonomous robot. We chose to specify the task without distinction between a locomotion and a manipulation component. The robot autonomously plans either of them or interleaves both of them if necessary in order to realize the required task. This is what we called the non-decoupled locomotion-and-manipulation planning. We analytically demonstrated this paradigm on low-dimensional example systems.

We then tackled the full-scale real-life systems made of humanoid robots armed with their dexterous locomotion and manipulation capabilities, having in mind the objective of extending existing multi-contact planning algorithms in accordance with our non-decoupled paradigm, scaled to fit the humanoid planning problems. We reached this objective by considering general centralized multi-agent systems of which the humanoid robots, the dexterous hands, the manipulator arms, the manipulated object, appear as being only particular instances. This way we generated sequences of stances and postures that encode the control-wise significant changes in the contact configurations necessary to realize a wide variety of tasks of different natures with one unique planner.

Each posture in this sequence was generated with an inverse kinematics solver that outputs a configuration of the system of robots under static equilibrium, within their joint and torque limits, avoiding collision, and reaching a specified set of contacts that are frictional, non-horizontal, non-co-planar, and as such general enough to consider the desired variety of humanoid planning problems.

We subsequently synthesized dynamically-consistent continuous motion that goes through this sequence of static configurations considered as milestones of the motion. This was done by a feedback multi-objective controller, the objectives of which are decided by a finite-state machine built upon the sequence of static postures.

Finally we envisioned the possibility of considering deformable environment objects used as contact support. We re-solved the inverse kinematics problem integrating finite-element models of linear-elastic materials that constitute the deformable environment.

From here on, possible future research in the direction sketched by the conclusion of this PhD work (besides the technical issues listed at the conclusive section of each respective chapter of the dissertation) might go towards more autonomy and more reactivity.

First the autonomy. The strongest hypothesis assumed in the present work is the full knowledge of the environment by our autonomous humanoid agents. Sensing is considered as a black box that provides them with perfect model of the environment. Integrating the sensing problem inside the contact planning state of the art is an unavoidable requirement to pretend reaching actual autonomy.

As for reactivity, the current computational cost of the different phases of the planning and execution framework in this work is far too prohibitive to rely on or hope for any short-term improvement of computational power of the processors in order to reach real-time performances. We should thus tackle from a theoretical angle this technical limitation of the implementation hardware support, by investigating algorithms parallelizability and distributivity on multi-core processors, for example.

Finally both autonomy and reactivity necessitate dealing with uncertainty. This uncertainty should be resolved at the sensing, the planning, and the execution levels. Not to mention the absence of unpredictable free-willing agents, namely real human, in the environment of the humanoids, which is a core limitation given that one of the declared objectives of studying humanoids is precisely the interactivity with humans.

Yet we hope that our thesis contributes with a step along this complicated path towards the full autonomy of humanoids.

APPENDIX A

Potential Field Guide for Humanoid Multi-Contact Acyclic Motion Planning

We present a motion planning algorithm that computes rough trajectories used by a contact-points planner as a guide to grow its search graph. We adapt collision-free motion planning algorithms to plan a path within the guide space, a submanifold of the configuration space included in the free space in which the configurations are subject to static stability constraint. We first discuss the definition of the guide space. Then we detail the different techniques and ideas involved: relevant C-space sampling for humanoid robot, task-driven projection process, static stability test based on polyhedral convex cones theory's double description method. We finally present results from our implementation of the algorithm.

A.1. Introduction

In Chapter 2 we adapted Best First Planning to multi-contact planning by growing the search tree in the space of sets of contacts. A key element of this contacts planner is the potential field that drives the search. It has to be carefully chosen as the planner may get trapped in local minima, which occur for example when we choose too simple potential fields such as the Euclidian distance to goal. An inappropriate potential field may also lead to the planning of complicated paths and postures. In [**EKMG08**], a solution is given by building the potential field around a rough trajectory, a *contact-points guide*, that gives an approximation of the intended path in the workspace as well as an idea of the postures that the robot has to adopt along this path. This trajectory was given manually as an input to the planner. Our aim in this work is to provide such a trajectory automatically, thus giving more autonomy to the robot.

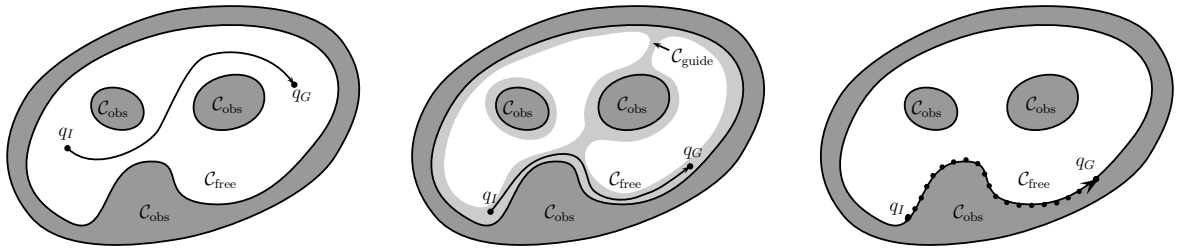
A.2. Solution

The main idea is to adapt existing collision-free motion planning algorithms to plan the contact-points guide.

A.2.1. General algorithm. The collision-free motion planning problem can be formalized as follows (adapted from [**LaV06**]):

Problem A.2.1 (collision-free motion planning problem).

- *a* world $\mathcal{W} = \mathbb{R}^3$.
- *an* obstacle region $\mathcal{O} \subset \mathcal{W}$.
- *a* robot $\mathcal{R}$ *defined in* $\mathcal{W}$ *as a kinematic tree of* m joints $\mathcal{J}_1, \mathcal{J}_2, \ldots, \mathcal{J}_m$ *to which* rigid bodies $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_m$ *are attached.*
- *the* configuration space *(also called* C-space*)* $\mathcal{C}$ *defined as the set of all possible transformations that may be applied to the robot. The image of the robot* $\mathcal{R}$ *in the*



(a) Collision-free motion planning. (b) Contacts guide planning. (c) Contact-points planning.

FIGURE A.1. Illustration of the problem.

configuration q *is denoted* $\mathcal{R}(q)$. *From* $\mathcal{C}$ *we derive* $\mathcal{C}_{\text{free}} = \{q \in \mathcal{C} \mid \mathcal{R}(q) \cap \mathcal{O} = \varnothing\}$ *and* $\mathcal{C}_{\text{obs}} = \mathcal{C} \setminus \mathcal{C}_{\text{free}}$.

- *a* query pair $(q_I, q_G) \in \mathcal{C}_{\text{free}}^2$ *of initial and goal configurations.*
- *an algorithm must compute a continuous* path $\tau : [0,1] \to \mathcal{C}_{\text{free}}$ *such that* $\tau(0) = q_I$ *and* $\tau(1) = q_G$.

Two classes of methods exist so far to address this problem [**LaV06**]: combinatorial motion planning and sampling-based motion planning. The difference between the two lies in that the latter avoids explicit construction of $\mathcal{C}_{\text{obs}}$. Instead it uses a sampling of the C-space to grow a discrete graph $\mathcal{G}(V, E)$, called a *roadmap*, of which every vertex $v \in V$ represents a configuration $q \in \mathcal{C}_{\text{free}}$ and every edge $e \in E$ represents a continuous path in $\mathcal{C}_{\text{free}}$, that progressively covers $\mathcal{C}_{\text{free}}$. The search for the path is then conducted into the constructed roadmap that supposedly represents an approximation of the connectivity of $\mathcal{C}_{\text{free}}$. Different instantiations of sampling-based motion planning as a general approach exist [**KScLO96**][**LK01**]. Algorithm 3 gives the general frame of the one we take as a starting point for our study, keeping in mind that it is possible to choose any other instantiation modulo adequate modifications.

Algorithm 3 sampling-based collision-free motion planning.

```
1: initialize G(V ← {q_I, q_G}, E ← ∅)
2: while no path found in G do
3:    sample a random configuration q_s in C
4:    if q_s ∈ C_free then
5:       for all q_V ∈ V ∩ NEIGHBOURHOOD(q_s) do
6:          if the direct path τ_d(q_s, q_V) lies in C_free then
7:             V.add(q_s) and E.add(τ_d(q_s, q_V))
8:          end if
9:       end for
10:   end if
11: end while
```

Now we would like to adapt algorithm 3 in order to plan a contact-points guide. The problem is that the path yielded by a contact-points planner lies on the *boundary* of $\mathcal{C}_{\text{obs}}$: $\partial\mathcal{C}_{\text{obs}}$. Simply replacing $\mathcal{C}_{\text{free}}$ with $\partial\mathcal{C}_{\text{obs}}$ in algorithm 3 would be a failing strategy as the measure of $\partial\mathcal{C}_{\text{obs}}$ is equal to zero . This means that the rejection rate at line 4 would be equal to 1. The second problem with this strategy concerns the linear direct paths

in line 6, as $\partial\mathcal{C}_{\text{obs}}$ is generally a non linear submanifold, a linear edge joining two of its elements will almost always be completely outside the submanifold.

Our solution is to consider a submanifold of $\mathcal{C}$ of non-zero measure, we label it $\mathcal{C}_{\text{guide}}$, that can be visually represented as a layer wrapping each connected component of $\partial\mathcal{C}_{\text{obs}}$. The idea, to some extent similar to [**ABD+98**], is to sample configurations "near" the obstacles; however, work in [**ABD+98**] focuses on 6D rigid robots, whereas our primary targets are polyarticulated humanoid robots. We will now detail our definition of $\mathcal{C}_{\text{guide}}$.

A *contact situation* between a body $\mathcal{B}_i$ of the robot and the obstacle region $\mathcal{O}$ is normally defined as

$$\partial\mathcal{B}_i \cap \partial\mathcal{O} \neq \varnothing \quad \text{and} \quad \text{int}(\mathcal{B}_i) \cap \text{int}(\mathcal{O}) = \varnothing \tag{A.1}$$

One way of adding a dimension, and thus creating a "volume", to the submanifold $\mathcal{C}_{\text{guide}}$ could be to consider the body $\mathcal{B}_i$ as in contact with $\mathcal{O}$ if $d(\mathcal{B}_i, \mathcal{O}) < \varepsilon_{\text{contact}}$, which is a positive fixed threshold. d denotes the Euclidian distance.

Definition A.2.2 (body-obstacle contact situation). *A rigid body $\mathcal{B}$ is in contact with an obstacle region $\mathcal{O}$ if*

$$0 < d(\mathcal{B}, \mathcal{O}) < \varepsilon_{\text{contact}} \tag{A.2}$$

In this situation, we denote by $A_{\mathcal{B}}$ and $A_{\mathcal{O}}$ respectively the closest points on the body and on the obstacle and by $\mathbf{n} = \overrightarrow{A_{\mathcal{O}}A_{\mathcal{B}}}/\|\overrightarrow{A_{\mathcal{O}}A_{\mathcal{B}}}\|$ the normal of the contact. The robot $\mathcal{R}$ is in contact in configuration $q \in \mathcal{C}_{\text{free}}$ if at least one of its bodies is in contact in configuration q.

We can now define $\mathcal{C}_{\text{guide}}$ as

$$\mathcal{C}_{\text{guide}} = \{q \in \mathcal{C}_{\text{free}} \mid \mathcal{R} \text{ is in contact in configuration } q\} \tag{A.3}$$

and then plan a collision-free path in $\mathcal{C}_{\text{guide}}$ using algorithm 3 and replacing in it all the occurrences of $\mathcal{C}_{\text{free}}$ by $\mathcal{C}_{\text{guide}}$. This would produce a path that could be tricky to follow by the contact-points planner as the latter will have to compute *statically stable* configurations along this path, and may need to stray significantly from the given path to find these stable configurations. So we have to refine the definition of $\mathcal{C}_{\text{guide}}$ to take static stability into account.

Considering the laws of rigid body dynamics applied to $\mathcal{R}$ and assuming that there are no limits to the torques we can apply to the robot joints (which is only an approximation), the static stability condition is simply written

$$\begin{cases} \sum_{\mathbf{f}\in\mathcal{F}} \mathbf{f} + m\mathbf{g} = \mathbf{0} \\ \sum_{\mathbf{f}\in\mathcal{F}} \mathscr{M}_O(\mathbf{f}) + \mathscr{M}_O(m\mathbf{g}) = \mathbf{0} \end{cases} \tag{A.4}$$

where $\mathcal{F}$ is the set of all *contact forces* applied to the robot, and $\mathscr{M}_O$ is the moment of a force in a point $O \in \mathbb{R}^3$. m is the mass of the robot and $\mathbf{g}$ the gravity vector. For simplicity we have modeled any surface contact as a discrete set of punctual contacts applied at chosen points distributed over the contact surface (we intentionally do not make it explicit in our formulas for readability's sake). Each contact force $\mathbf{f} \in \mathcal{F}$ applied on the robot at a point $A \in \partial\mathcal{R}$ with a normal $\mathbf{n}$ lies in a *friction cone* $\mathscr{C}_{A,\mathbf{n},\theta}$, θ being the angle of the cone that depends on the friction coefficient between the body and the obstacle, A is the apex of the cone, and $\mathbf{n}$ defines the revolution axis of the cone.

DEFINITION A.2.3 (static stability situation). *The robot $\mathcal{R}$ placed in a configuration $q \in \mathcal{C}_{\text{free}}$ is statically stable if*

$$\forall i \in I(q),\ \exists\, \mathbf{f}_i \in \mathscr{C}_{A_{\mathcal{B}_i},\mathbf{n}_i,\theta_i}, \text{s.t.} \begin{cases} \sum_{i\in I(q)} \mathbf{f}_i + m\mathbf{g} = \mathbf{0} \\ \sum_{i\in I(q)} \mathscr{M}_O(\mathbf{f}_i) + \mathscr{M}_O(m\mathbf{g}) = \mathbf{0} \end{cases} \tag{A.5}$$

where

$$I(q) = \Big\{ i \in \{1,\dots,m\} \mid 0 < d(\mathcal{B}_i(q), \mathcal{O}) < \varepsilon_{\text{contact}} \Big\} \tag{A.6}$$

We can now introduce our new definition of $\mathcal{C}_{\text{guide}}$ as

$$\mathcal{C}_{\text{guide}} = \{q \in \mathcal{C}_{\text{free}} \mid \mathcal{R} \text{ is statically stable in configuration } q\} \tag{A.7}$$

and once again try to adapt algorithm 3. This is still not enough, as the rejection rate of our sampling would still be very high. This is the reason why we have decided to split the sampling procedure into two distinct phases: the sampling of a more or less uniform random configuration q_{s} in $\mathcal{C}$, followed by a projection process of q_{s} to try to make it fit inside $\mathcal{C}_{\text{guide}}$. This projection process is for now only applied on the sampled configurations, and on some discretization points along the linear direct path. There is no guaranty, however, that the whole continuous direct path is inside $\mathcal{C}_{\text{guide}}$.

Finally, we get algorithm 4, which is the adaptation of algorithm 3 taking into account the previously discussed points. $p : \mathcal{C}_{\text{free}} \longrightarrow \mathcal{C}_{\text{guide}}$ denotes the projection function.

Algorithm 4 contact-points guide planning

```
1: initialize G(V ← {q_I, q_G}, E ← ∅)
2: while no path found in G do
3:    sample a random configuration q_s in C
4:    if q_s ∈ C_free then
5:       apply projection q_p = p(q_s) ∈ C_guide
6:       for all q_V ∈ V ∩ NEIGHBOURHOOD(q_p) do
7:          if (a discretization of) τ_d(q_p, q_V) lies in C_guide then
8:             V.add(q_p) and E.add(τ_d(q_p, q_V))
9:          end if
10:      end for
11:   end if
12: end while
```

We will now get into the detail of the different steps of execution of algorithm 4, especially the lines 3 and 5.

A.2.2. Sampling random configurations. In this section we detail line 3 of algorithm 4.

Our humanoid robot $\mathcal{R}$ is represented as a kinematic tree of m joints $\mathcal{J}_1, \dots, \mathcal{J}_m$. The root joint $\mathcal{J}_1$ is a six-dimensional free flyer that evolves in the C-space $\mathbb{R}^3 \times SO(3)$, or, if the translations are bounded, $[x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}] \times [z_{\min}, z_{\max}] \times SO(3)$. The remaining joints are revolute joints yielding the C-space $\prod_{i=2}^{m} [\theta_{i,\min}, \theta_{i,\max}]$. The total C-space is consequently

$$\mathcal{C} = \mathbb{R}^3 \times SO(3) \times \prod_{i=2}^{m} [\theta_{i,\min}, \theta_{i,\max}] \tag{A.8}$$

that we can write in a more expressive way as

$$\mathcal{C} = \mathcal{C}_{\text{position}} \times \mathcal{C}_{\text{orientation}} \times \mathcal{C}_{\text{posture}} \tag{A.9}$$

A random C-space variable Q is as such a vector of three independent random C-space variables $Q = (Q_{\text{position}}, Q_{\text{orientation}}, Q_{\text{posture}})$.

A.2.2.1. *Position sampling.* Q_{position} can be either a uniform random variable if the workspace $\mathcal{W}$ is bounded or a spatial Gaussian random variable otherwise.

A.2.2.2. *Orientation sampling.* For the orientation we would like to bias the sampling in order to favor some interesting orientations for a humanoid robot, such as the standing-up orientation for a walk, the laying-down orientation for a crawl, or a slightly front-leant orientation for a climb. $SO(3)$ being homeomorphic to the unit quaternion sphere $\mathbb{S}^3$, we need a random variable that looks like a Gaussian distribution on the sphere $\mathbb{S}^3$ around one of its points q_0 that would represent one of the orientations above. The *Von Mises - Fisher distribution* [**MJ00**] achieves this very purpose. Given a *mean* unit vector q_0 and a *concentration parameter* $\kappa \in \mathbb{R}^+$, the probability density function of the Von Mises-Fisher distribution on the sphere $\mathbb{S}^{p-1} \subset \mathbb{R}^p$ is

$$f_{q_0,\kappa}(q) = C_p(\kappa) \exp\left(\kappa q_0^T q\right) \tag{A.10}$$

$C_p(\kappa)$ is a normalization constant

$$C_p(\kappa) = \frac{\kappa^{p/2-1}}{(2\pi)^{p/2} I_{p/2-1}(\kappa)} \tag{A.11}$$

where I_v denotes the modified Bessel function of the first kind and order v. The parameter κ controls the concentration of the distribution around q_0. The bigger κ the more concentrated the distribution. $\kappa = 0$ yields a uniform distribution over the sphere. An algorithm for simulating a Von Mises-Fisher random variable is given in [**Woo94**].

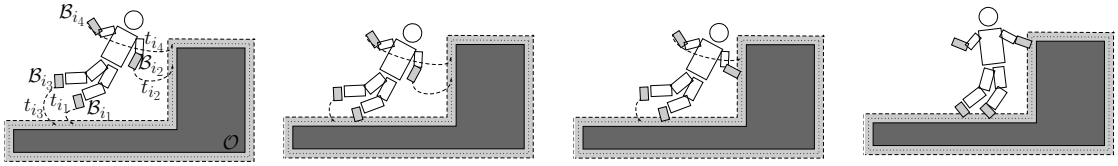
A.2.2.3. *Posture sampling.* Now we want to sample the posture space $\mathcal{C}_{\text{posture}} = \prod_{i=2}^{m} [\theta_{i,\min}, \theta_{i,\max}]$. We could immediately choose for Q_{posture} a uniform random variable. However, this would produce postures that once again are not interesting enough for a humanoid robot, especially when the dimension $m-1$ of this manifold is relatively high ($m - 1 = 30$ in our humanoid platform). To solve this problem we choose to reduce the dimensionality of $\mathcal{C}_{\text{posture}}$ by sampling in the affine space generated by the standing-up posture q_{key_0} and a certain number of *key postures* $q_{\text{key}_1}, \ldots, q_{\text{key}_n}$. These latter postures should be relevant for a humanoid robot and could represent for example the sitting-down posture, the four-legged posture, etc. To remain within the joints limits, we consider the bounded space

$$\mathcal{C}_{\text{posture}} = \left\{ q_{\text{key}_0} + \sum_{i=1}^{n} \lambda_i (q_{\text{key}_i} - q_{\text{key}_0}) \mid (\lambda_i)_i \in (\mathbb{B}_k^n)^+ \right\} \tag{A.12}$$

where $(\mathbb{B}_k^n)^+$ is the positive quadrant of the unit ball of dimension n for the k-norm $\|.\|_k$

$$(\mathbb{B}_k^n)^+ = \left\{ (\lambda_i)_i \in [0,1]^n \mid \sum_{i=1}^{n} \lambda_i^k \leq 1 \right\} \tag{A.13}$$

that we sample uniformly.



(a) Initial configuration. (b) First iteration. (c) Second iteration. (d) Final configuration statically stable.

FIGURE A.2. Illustration of the projection process.

A.2.3. Projection process. We detail now line 5 of algorithm 4. What we mean by *projection* here is an operation that tries to bring a given configuration sample in $\mathcal{C}_{\text{free}}$ inside $\mathcal{C}_{\text{guide}}$. The idea of projection was introduced in [**CSL02**] and further investigated in [**Sti07**]. The solution we choose is to use a *stack of tasks* solver based on *generalized inverse kinematics* called *hppGik* and presented in [**YKEJL06**]. A *task* is a function $f : \mathcal{C} \longrightarrow \mathbb{R}$ that we would like to bring to zero, *i.e* to solve $f(q) = 0, q \in \mathcal{C}$. Suppose we have sampled a random configuration q_{s}. From this configuration we want to compute a statically stable configuration, thus we have to create contacts with the neighboring obstacles, given that the more contacts we create the more stable the configuration is likely to be. On the other hand, the more contacts we create the more we deform the original posture and reduce the mobility for the next posture, this is why we should create the "minimum" number of contacts to ensure the stability. To create a contact between a body $\mathcal{B}$ and the obstacle region $\mathcal{O}$ we need to bring it to a distance closer than $\varepsilon_{\text{contact}}$. Let us define the *goal point* A_{goal} as the point translated from $A_{\mathcal{O}}$ by a $\varepsilon_{\text{contact}}/2$ distance following $\mathbf{n}$, and the *goal plan* P_{goal} as the plan normal to $\mathbf{n}$ in A_{goal}. The task that we want to formalise is "bring the point $A_{\mathcal{B}}$ in the plan P_{goal}", *i.e.* bring to 0 the corresponding task function

$$f(q) = (\overrightarrow{A_{\text{goal}} A_{\mathcal{B}}(q)} | \mathbf{n}) \tag{A.14}$$

where $(.|.)$ denotes the Euclidian scalar product. To solve the task $f(q) = 0$ we implent the *Newton's method* for finding zeros of a function (the same idea is suggested [**HBL05**]). To do so we linearize f around a start configuration q_0 as

$$f(q) \simeq f(q_0) + \frac{\partial f}{\partial q}(q_0).dq \tag{A.15}$$

where $dq = q - q_0$ and then we solve the linear system

$$f(q_0) + \frac{\partial f}{\partial q}(q_0).dq = 0 \tag{A.16}$$

using generalized inverse kinematics to compute the pseudo-inverse of $J(q_0) = \frac{\partial f}{\partial q}(q_0)$ that we denote $J(q_0)^{\dagger}$.The solution q_1 of the system is thus given by

$$q_1 = q_0 - J(q_0)^{\dagger} f(q_0) \tag{A.17}$$

The Newton's method consists in iterating again starting now from q_1, meaning that we construct a sequence $(q_n)_{n \in \mathbb{N}}$ recursively as

$$q_{n+1} = q_n - J(q_n)^{\dagger} f(q_n) \tag{A.18}$$

that supposedly converges to the solution. However, in our task of bringing the body close to the obstacle, we do not really need to converge to the exact solution, but rather to converge towards a static stability situation, even though this latter is far from the exact solution. This is why we have chosen the Newton's method, as we can stop its execution after each single iteration to test the static stability of the intermediate solutions, and can reach the static stability after few iterations. Now we would like to bring not only one body $\mathcal{B}$ close to the obstacle region $\mathcal{O}$, but the maximum number of bodies $\mathcal{B}_1, \ldots, \mathcal{B}_m$ to $\mathcal{O}$, this means that we need to solve the system of equations:

$$\bigcap_{i=1}^{m} f_i(q) = 0 \tag{A.19}$$

or the linearized version

$$\bigcap_{i=1}^{m} f_i(q_0) + \frac{\partial f_i}{\partial q}(q_0)dq = 0 \tag{A.20}$$

The stack of tasks solver *hppGik* [**YBEL05**] allows us to solve such a system with *priorities*, meaning that it solves the first equation, then it tries to solve the second equation at best while remaining in the solution space of the first equation, and so on. The priority we choose is the distance to obstacle, as we try to bring closer with the highest priority the closest body to the obstacles. Let $i_1, \ldots, i_m \in \{1, \ldots, m\}$ be the indexes of the bodies sorted in increasing order of distance to $\mathcal{O}$, *i.e*:

$$d(\mathcal{B}_{i_1}, \mathcal{O}) \leq d(\mathcal{B}_{i_2}, \mathcal{O}) \leq \ldots \leq d(\mathcal{B}_{i_m}, \mathcal{O}) \tag{A.21}$$

The *hppGik* solver solves, in the order of priority, the following stack of tasks:

$$\bigcap_{j=1}^{m} t_j : f_{i_j}(q_0) + \frac{\partial f_{i_j}}{\partial q}(q_0)dq = 0 \tag{A.22}$$

where t_j is the task of priority j.

Finally we give algorithm 5 of the projection process, in which we introduce one new task per iteration in order to deform as little as possible the posture. We also stop the process after a maximum number of iterations, after which we discard the current configuration and we start again the process with a new q_s according to algorithm 4.

Algorithm 5 projection process

1: sample a random configuration q_s
2: set $q_0 \leftarrow q_s$
3: COUNTER $\leftarrow 1$
4: **while** q_0 is not statically stable and COUNTER $<$ MAX_ITERATIONS **do**
5: sort the bodies $d(\mathcal{B}_{i_1}, \mathcal{O}) \leq \ldots \leq d(\mathcal{B}_{i_m}, \mathcal{O})$
6: $q_0 \leftarrow$ solution of the stack of tasks $(t_1, \ldots, t_{\text{COUNTER}})$
7: COUNTER $\leftarrow$ COUNTER $+ 1$
8: **end while**
9: **return** q_0

We will now get into the detail of line 4 of algorithm 5, in which we have to test the static stability of a configuration.

A.2.4. Testing the static stability. Suppose we have the robot $\mathcal{R}$ in configuration q and we want to check whether or not it is statically stable in this configuration, according to definition A.2.3. In order to get a linear system, we need to consider the modeling of each friction cone $\mathscr{C}_{A_{\mathcal{B}_i},\mathbf{n}_i,\theta_i}$ as discrete *polyhedral cone* with a finite number of generators $\mathbf{u}_{i,1},\ldots,\mathbf{u}_{i,n_i}$

$$\mathscr{C}_{A_{\mathcal{B}_i},\mathbf{n}_i,\theta_i} = \mathscr{C}(\mathbf{u}_{i,1},\ldots,\mathbf{u}_{i,n_i}) \tag{A.23}$$

$$= \Big\{ \sum_{j=1}^{n_i} \lambda_j \mathbf{u}_{i,j} \ / \ \lambda_1,\ldots,\lambda_{n_i} \in \mathbb{R}^+ \Big\} \tag{A.24}$$

which is the set of all *non negative* linear combinations of the generators. With this modeling, we have

$$\mathbf{f}_i \in \mathscr{C}_{A_{\mathcal{B}_i},\mathbf{n}_i,\theta_i} \Longleftrightarrow \exists\,(\lambda_{i,j})_{j=1..n_i} \in \left(\mathbb{R}^+\right)^{n_i},\ \mathbf{f}_i = \sum_{j=1}^{n_i} \lambda_{i,j}\mathbf{u}_{i,j} \tag{A.25}$$

allowing us to rewrite the static stability condition as a linear problem

$$\exists\,(\lambda_{i,j})_{\substack{i\in I(q)\\ j=1..n_i}} \in \prod_{i\in I(q)} \left(\mathbb{R}^+\right)^{n_i},\ \text{s.t.} \begin{cases} \sum_{\substack{i\in I(q)\\ j=1..n_i}} \lambda_{i,j}\mathbf{u}_{i,j} + m\mathbf{g} = \mathbf{0} \\ \sum_{\substack{i\in I(q)\\ j=1..n_i}} \mathscr{M}_O(\lambda_{i,j}\mathbf{u}_{i,j}) + \mathscr{M}_O(m\mathbf{g}) = \mathbf{0} \end{cases} \tag{A.26}$$

The system of two 3-dimensional equations can be written as a single 6-dimensional equation, putting

$$\mathbf{a}_{i,j} = \begin{pmatrix} \mathbf{u}_{i,j} \\ \mathscr{M}_O(\mathbf{u}_{i,j}) \end{pmatrix} \quad \text{and} \quad \mathbf{v} = -\begin{pmatrix} m\mathbf{g} \\ \mathscr{M}_O(m\mathbf{g}) \end{pmatrix} \tag{A.27}$$

the static stability condition then becomes

$$\exists\,(\lambda_{i,j})_{\substack{i\in I(q)\\ j=1..n_i}} \in \prod_{i\in I(q)} \left(\mathbb{R}^+\right)^{n_i},\ \text{s.t.} \sum_{\substack{i\in I(q)\\ j=1..n_i}} \lambda_{i,j}\mathbf{a}_{i,j} = \mathbf{v} \tag{A.28}$$

which can be read as the membership of $\mathbf{v}$ in the cone generated by the $\mathbf{a}_{i,j}$ vectors

$$\mathbf{v} \in \mathscr{C}(\mathbf{a}_{i,j})_{i,j} \tag{A.29}$$

To solve this system, we used some results that come from the *polyhedral convex cone theory* that we detail hereafter.

Polyhedral convex cone theory. Let $\mathscr{C}(\mathbf{a}_1,\ldots,\mathbf{a}_m)$ be the cone generated by $\mathbf{a}_1,\ldots,\mathbf{a}_m$ in $\mathbb{R}^n$

$$\mathscr{C}(\mathbf{a}_1,\ldots,\mathbf{a}_m) = \Big\{ \sum_{j=1}^{m} \lambda_j \mathbf{a}_j \mid \lambda_1,\ldots,\lambda_m \in \mathbb{R}^+ \Big\} \tag{A.30}$$

the *dual cone* (also called the *polar cone*) $\mathscr{C}^p$ is defined as

$$\mathscr{C}^p(\mathbf{a}_1,\ldots,\mathbf{a}_m) = \Big\{ \mathbf{x} \in \mathbb{R}^n \mid \forall i \in \{1,\ldots,m\}\ \mathbf{x}^T\mathbf{a}_i \leq 0 \Big\} \tag{A.31}$$

Minkowski [**Min11**] demonstrated that the polar cone is a cone too, *i.e.* $\exists\,\mathbf{b}_1,\ldots,\mathbf{b}_k \in \mathbb{R}^n$ such that

$$\mathscr{C}^p(\mathbf{a}_1,\ldots,\mathbf{a}_m) = \mathscr{C}(\mathbf{b}_1,\ldots,\mathbf{b}_k) \tag{A.32}$$

The *Farkas lemma* [**Far02**] states that $(\mathscr{C}^p)^p = \mathscr{C}$ *i.e.*

$$\mathscr{C}(\mathbf{a}_1, \ldots, \mathbf{a}_m) = \mathscr{C}^p(\mathbf{b}_1, \ldots, \mathbf{b}_k) \tag{A.33}$$

this result allows us to test the membership of a vector $\mathbf{x} \in \mathbb{R}^n$ in the dual of the dual cone instead of the cone itself

$$\mathbf{x} \in \mathscr{C}(\mathbf{a}_1, \ldots, \mathbf{a}_m) \Longleftrightarrow \mathbf{x} \in \mathscr{C}^p(\mathbf{b}_1, \ldots, \mathbf{b}_k) \tag{A.34}$$

or

$$\exists (\lambda_j)_j \in (\mathbb{R}^+)^m \ \mathbf{x} = \sum_{j=1}^{m} \lambda_j \mathbf{a}_j \Longleftrightarrow \forall i \in \{1, \ldots, k\} \ \mathbf{x}^T \mathbf{b}_i \leq 0 \tag{A.35}$$

The second member of this latter equivalence is much easier to check than the first one, if we could compute the vectors $\mathbf{b}_1, \ldots, \mathbf{b}_k$. The *Motzkin's double description algorithm* [**MRTT53**] achieves this. We implemented a variation of the original algorithm, proposed by Padberg [**Pad99**], that allows us to compute a minimal set of generators for the dual cone.

A.3. Results

We implemented the ideas presented in the previous section within the HPP framework using KineoCAM's software Kineo Path Planner and KineoWorks as a core collision-free motion planning and collision detection module. The model we used for the humanoid robot is HRP-2 [**KKK$^+$04**] which has 36 degrees of freedom (including the free-flyer). The collision-free path planning algorithms we choose are either basic PRM [**KScLO96**] or bidirectional RRT [**LK01**].

The main scenario we considered is the highly constrained one demonstrated in [**EKMG08**] which consists in standing up from a chair and going away from a table. The robot is sitting on the chair in initial configuration and is standing by the table at final configuration. The guide obtained is shown in figure A.3 while the contacts points plan is illustrated in figure A.4. Using the distance to goal as a potential function the robot ends up climbing the table and the contacts planning stops after having consumed all the memory resource of the computer. With the provided guide the contacts planner finds the solution in approximately 3h30min on a standard Pentium IV system, after approximately 10min of computation for the guide.

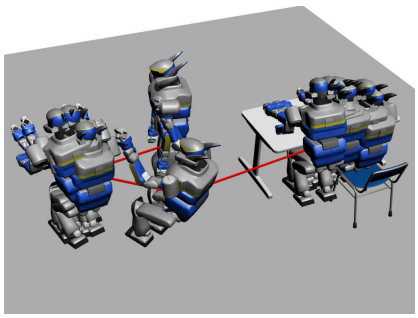


FIGURE A.3. Guide planning for the out-of-table-and-chair scenario.

We also tested the guide planner on other scenarios on which we have not yet tested the contacts planner, simply to demonstrate the ability of the guide planner of going through different situations (Figs. A.5a and A.5b).

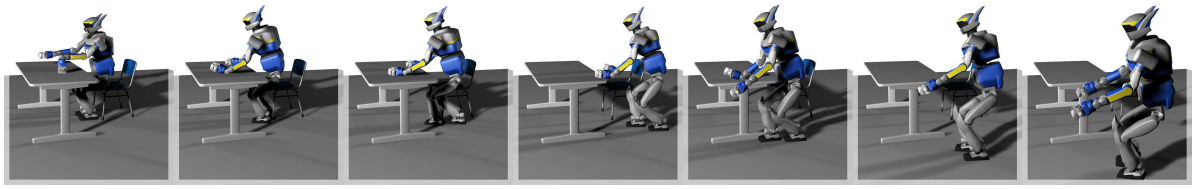
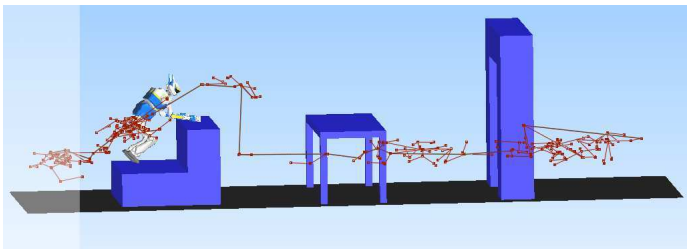
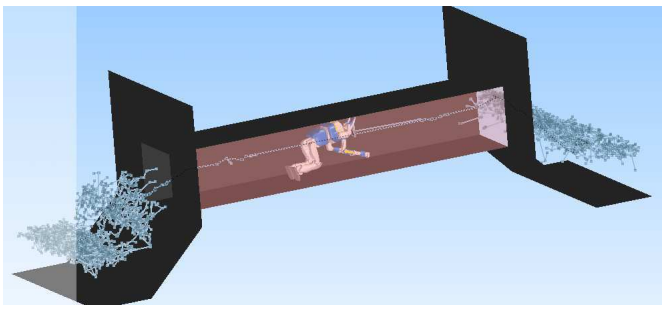


FIGURE A.4. Contacts planning for the out-of-table-and-chair scenario following the guide provided by the contacts guide planner.



(a) Over the sofa.



(b) Through the tunnel.

FIGURE A.5. Different scenarios.

Although the gain in computing time that we achieve at the contact-points planner's level is theoretically infinite, computing time at the contacts-guide planner's level remains relatively high for scenarios such as A.5a and A.5b (a few hours). The time is consumed both on distance computation and stack of tasks solving which are solicited at each iteration of the algorithm.

A.4. Conclusion

Improvements of our contacts guide planner are still possible and need to be considered, especially regarding line 7 of algorithm 4. Ensuring that the continuous direct path linking two configurations in guide space lies in the guide space remains an unanswered question in our work. We also still need to work on the linking method that computes the direct path between two guide space's configurations, and which is for now a linear direct path linking method. We added a dimension and thus "volume" to $\mathcal{C}_{\text{guide}}$ in order to pass the test line 7 with higher probability; however, a better solution would be to apply a projection to the whole linear direct path in order to make it fit inside $\mathcal{C}_{\text{guide}}$. These are all questions we plan to investigate in future work.

Bibliography

[AB88] Francis Avnaim and Jean-Daniel Boissonnat, *Polygon Placement Under Translation and Rotation*, Symposium on Theoretical Aspects of Computer Science, 1988, pp. 322–333.

[ABD+98] Nancy M. Amato, O. Burchan Bayazit, C. Lucia K. Dale, Chris Jones, and Daniel Vallejo, *OBPRM: An Obstacle-Based PRM for 3D Workspaces*, Workshop on Algorithmic Foundations of Robotics, 1998.

[AdSP07] Yeuhi Abe, Marco da Silva, and Jovan Popovic, *Multiobjective Control with Frictional Contacts*, Eurographics/ACM SIGGRAPH Symposium on Computer Animation, 2007.

[ALS95] Rachid Alami, Jean-Paul Laumond, and Thierry Siméon, *Two Manipulation Planning Algorithms*, Proceedings of the Workshop on Algorithmic Foundations of Robotics, 1995, pp. 109–125.

[APE04] Christine Azevedo, Philippe Poignet, and Bernard Espiau, *Artificial Locomotion Control: from Human to Robots*, Robotics and Autonomous Systems **47** (2004), no. 4, 203–223.

[BDG85] J.E. Bobrow, S. Dubowsky, and J.S. Gibson, *Time-Optimal Control of Robotic Manipulators Along Specified Paths*, International Journal of Robotics Research **4** (1985), no. 3, 3–17.

[BELK09] Karim Bouyarmane, Adrien Escande, Florent Lamiraux, and Abderrahmane Kheddar, *Potential Field Guide for Multicontact Humanoid Motion Planning*, Proceedings of the IEEE International Conference on Robotics and Automation, 2009.

[BEMK09] Mehdi Benallegue, Adrien Escande, Sylvain Miossec, and Abderrahmane Kheddar, *Fast C^1 Proximity Queries Using Support Mapping of Sphere-Torus-Patches Bounding Volumes*, Proceedings of the IEEE International Conference on Robotics and Automation, 2009.

[BF05] M. Bonnet and A. Frangi, *Analyse des Solides Déformables par la Méthode des Eléments Finis*, Ecole Polytechnique, 2005.

[BJ05] Jernej Barbic and Doug L James, *Real-Time Subspace Integration for St. Venant-Kirchhoff Deformable Models*, ACM Transactions on Graphics **24** (2005), no. 3.

[BK10] Karim Bouyarmane and Abderrahmane Kheddar, *Static Multi-Contact Inverse Problem for Multiple Humanoid Robots and Manipulated Objects*, Proceedings of IEEE-RAS International Conference on Humanoid Robots, 2010.

[BK11a] ———, *FEM-based Static Posture Planning for a Humanoid Robot on Deformable Contact Support*, Proceedings of IEEE-RAS International Conference on Humanoid Robots, 2011.

[BK11b] ———, *Multi-Contact Stances Planning for Multiple Agents*, Proceedings of the IEEE International Conference on Robotics and Automation, 2011.

[BK11c] ———, *Using a Multi-Objective Controller to Synthesize Simulated Humanoid Robot Motion with Changing Contact Configurations*, Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2011.

[BL00] Francesco Bullo and Andrew D. Lewis, *Geometric Control of Mechanical Systems*, Springer, 2000.

[BL08] T. Bretl and S. Lall, *Testing Static Equilibrium for Legged Robots*, IEEE Transactions on Robotics **24** (2008), no. 4, 794–807.

[Bou] Karim Bouyarmane, *Homepage*, http://staff.aist.go.jp/karim.bouyarmane.

[Bre06] Tim Bretl, *Motion Planning of Multi-Limbed Robots Subject to Equilibrium Constraints: The Free-Climbing Robot Problem*, International Journal of Robotics Research **27** (2006), no. 4, 317–342.

[BS96] H. Bruyninckx and J. De Schutter, *Symbolic Differentiation of the Velocity Mapping for a Serial Kinematic Chain*, Mechanism And Machine Theory **31** (1996), no. 2, 135–148.

[BT08] Pinhas Ben-Tzvi, *Hybrid Mobile Robot Systems: Interchanging Locomotion and Manipulation*, Ph.D. thesis, University of Toronto, 2008.

[BTGZ08] P. Ben-Tzvi, A.A. Goldenberg, and J.W. Zu, *Design and Analysis of a Hybrid Mobile Robot Mechanism with Compounded Locomotion and Manipulation Capability*, Transactions of the ASME Journal of Mechanical Design **130** (2008), 1–13.

[BW07] Stephen P. Boyd and Ben Wegbreit, *Fast Computation of Optimal Contact Forces*, IEEE Transactions on Robotics **23** (2007), no. 6, 1117–1132.

[CKNK03] Joel Chestnutt, James Kuffner, Koichi Nishiwaki, and Satoshi Kagami, *Planning Biped Navigation Strategies in Complex Environments*, Proceedings of the IEEE-RAS International Conference on Humanoid Robots, 2003.

[CLH+05] Howie Choset, Kevin M. Lynch, Seth Hutchinson, George Kantor, Wolfram Burgard, Lydia E. Kavraki, and Sebastian Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*, The MIT Press, 2005.

[CLK+05] J. Chestnutt, M. Lau, J.J. Kuffner, G. Cheung, J. Hodgins, and T. Kanade, *Footstep Planning for the ASIMO Humanoid Robot*, Proceedings of the IEEE International Conference on Robotics and Automation, 2005.

[CMK+06] J-R. Chardonnet, S. Miossec, A. Kheddar, H. Arisumi, H. Hirukawa, F. Pierrot, and K. Yokoi, *Dynamic Simulator for Humanoids Using Constraint-Based Method with Static Friction*, Proceedings of the IEEE International Conference on Robotics and Biomimetics, 2006.

[Col09] Cyrille Collette, *Virtual Humans Dynamic Control: Robust Balance and Task Management*, Ph.D. thesis, University of Paris VI, June 2009.

[Cra04] John J. Craig, *Introduction to Robotics: Mechanics and Control (3rd Edition)*, Prentice Hall, 2004.

[CSL02] Juan Cortes, Thierry Siméon, and Jean-Paul Laumond, *A Random Loop Generator for Planning the Motions of Closed Kinematic Chains Using PRM Methods*, Proceedings of the IEEE International Conference on Robotics and Automation, 2002.

[dLMH10] M. de Lasa, I. Mordatch, and A. Hertzmann, *Feature-based locomotion controllers*, ACM Transactions on Graphics (Proc. SIGGRAPH 2010) **29** (2010), no. 3.

[EAPL06] Claudia Esteves, Gustavo Arechavelata, Julien Pettré, and Jean-Paul Laumond, *Animation Planning for Virtual Characters Cooperation*, ACM Transactions on Graphics **25** (2006), no. 2, 319–339.

[EK09] Adrien Escande and Abderrahmane Kheddar, *Contact Planning for Acyclic Motion with Tasks Constraints*, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009.

[EKM06] Adrien Escande, Abderrahmane Kheddar, and Sylvain Miossec, *Planning Support Contact-Points for Humanoid Robots and Experiments on HRP-2*, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006.

[EKMG08] Adrien Escande, Abderrahmane Kheddar, Sylvain Miossec, and Sylvain Garsault, *Planning Support Contact-Points for Acyclic Motions and Experiments on HRP-2*, International Symposium on Experimental Robotics, 2008.

[ELP87] M.A. Erdmann and T. Lozano-Perez, *On Multiple Moving Objects*, Algorithmica **2** (1987), 477–521.

[EMW10] Adrien Escande, Nicolas Mansard, and Pierre-Brice Wieber, *Fast Resolution of Hierarchized Inverse Kinematics with Inequality Constraints*, Proceedings of the IEEE International Conference on Robotics and Automation, 2010.

[Esc08] Adrien Escande, *Planification de Points d'Appui pour la Génération de Mouvements Acycliques : Application aux Humanoides*, Ph.D. thesis, Université d'Evry-Val d'Essone, December 2008.

[Far02] J. Farkas, *Uber der Einfachen Ungleichungen*, Journal fuer die Reine und Angewandte Mathematik **124** (1902), 1–27.

[GB01] Bill Goodwine and Joel Burdick, *Controllability of Kinematic Control Systems on Stratified Configuration Spaces*, IEEE Transactions on Automatic Control **46** (2001), no. 3, 358–368.

[GB02] ———, *Motion Planning for Kinematic Stratified Systems With Application to Quasi-Static Legged Locomotion and Finger Gaiting*, IEEE Transactions on Robotics and Automation **18** (2002), no. 2, 209–222.

[GJK88] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi, *A Fast Procedure for Computing the Distance Between Complex Objects in Three-Dimentional Space*, IEEE Transactions on Robotics and Automation **4** (1988), 193–203.

[HA00] Li Han and Nancy M. Amato, *A Kinematics-Based Probabilistic Roadmap Method for Closed Chain Systems*, Proceedings of the Workshop on Algorithmic Foundations of Robotics, 2000.

[Hau08] Kris Hauser, *Motion Planning for Legged and Humanoid Robots*, Ph.D. thesis, Stanford University, December 2008.

[HBL05] Kris Hauser, Tim Bretl, and Jean-Claude Latombe, *Non-Gaited Humanoid Locomotion Planning*, Proceedings of the IEEE-RAS International Conference on Humanoid Robots, 2005.

[HBL+08] Kris Hauser, Tim Bretl, Jean-Claude Latombe, Kensuke Harada, and Brian Wilcox, *Motion Planning for legged Robots on Varied Terrain*, International Journal of Robotics Research **27** (2008), no. 11-12, 1325–1349.

[HHH+06] Hirohisa Hirukawa, Shizuko Hattori, Kensuke Harada, Shuuji Kajita, Kenji Kaneko, Fumio Kanehiro, Kiyoshi Fujiwara, and Mitsuharu Morisawa, *A Universal Stability Criterion of the Foot Contact of Legged Robots - Adios ZMP*, Proceedings of the IEEE International Conference on Robotics and Automation, 2006.

[HL08] Kris Hauser and Jean-Claude Latombe, *Multi-Modal Motion Planning for Non-Expansive Spaces*, Proceedings of the Workshop on the Algorithmic Foundations of Robotics, 2008.

[HL10] ———, *Multi-Modal Motion Planning in Non-expansive Spaces*, International Journal of Robotics Research **29** (2010), no. 7, 897–915.

[HNTH11] K. Hauser and V. Ng-Thow-Hing, *Randomized Multi-Modal Motion Planning for a Humanoid Robot Manipulation Task*, International Journal of Robotics Research **30** (2011), no. 6, 678–698.

[HNTHGB07] K. Hauser, V. Ng-Thow-Hing, and H. Gonzalez-Banos, *Multi-Modal Motion Planning for a Humanoid Manipulation Task*, Proceedings of the International Symposium on Robotics Research, 2007.

[HW86] John Hopcroft and Gordon Wilfong, *Motion of Objects in Contact*, International Journal of Robotics Research **4** (1986), no. 4, 32–46.

[Ish99] Chris Isham, *Modern Differential Geometry for Physicists*, 2nd ed., World Scientific Lecture Notes in Physics, vol. 61, World Scientific, 1999.

[KD05] W. Khalil and E. Dombre, *Modeling, Identification and Control of Robots*, Kogan Page Limited, 2005.

[KE08] Shuuji Kajita and Bernard Espiau, *Springer Handbook of Robotics*, ch. Legged Robots, Springer, 2008.

[KFH+08] Charles C. Kemp, Paul Fitzpatrick, Hirohisa Hirukawa, Kazuhito Yokoi, Kensuke Harada, and Yoshio Matsumoto, *Springer Handbook of Robotics*, ch. Humanoids, Springer, 2008.

[KKK+03] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kiyoshi Fujiwara, Kensuke Harada, Kazuhito Yokoi, and Hirohisa Hirukawa, *Biped Walking Pattern Generation by Using Preview Control of Zero-Moment Point*, Proceedings of the IEEE International Conference on Robotics and Automation, 2003.

[KKK+04] K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi, and T. Isozumi, *Humanoid Robot HRP-2*, Proceedings of the IEEE International Conference on Robotics and Automation, 2004.

[KKN+02] J.J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue, *Dynamically-Stable Motion Planning for Humanoid Robots*, Autonomous Robots **12** (2002), 105–118.

[KL10] O. Kanoun and J.P. Laumond, *Optimizing the Stepping of a Humanoid Robot for a Sequence of Tasks*, Proceedings of the IEEE-RAS International Conference on Humanoid Robots, 2010.

[KLY11] O. Kanoun, J.P. Laumond, and E. Yoshida, *Planning Foot Placements for a Humanoid Robot : a Problem of Inverse Kinematics*, International Journal of Robotics Research **30** (2011), no. 4, 476–485.

[KMB95] Evangelos Kokkevis, Dimitri Metaxas, and Norman I. Badler, *Autonomous Animation and Control of Four-Legged Animals*, Proceedings of Graphics Interface, 1995.

[KNK+01] J.J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue, *Footstep Planning Among Obstacles for Biped Robots*, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2001.

[KScLO96] Lydia E. Kavraki, Petr Svestka, Jean claude Latombe, and Mark H. Overmars, *Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces*, IEEE Transactions on Robotics and Automation **12** (1996), 566–580.

[KSP08] Oussama Khatib, Luis Sentis, and Jaeheung Park, *A Unified Framework for Whole-Body Humanoid Robot Control with Multiple Constraints and Contacts*, European Robotics Symposium, 2008, pp. 303–312.

[Lat91] Jean Claude Latombe, *Robot Motion Planning*, Kluwer Academic Publishers, 1991.

[Lau98] Jean-Paul Laumond, *Robot Motion Planning and Control*, Springer, 1998.

[LaV06] Steven M. LaValle, *Planning Algorithms*, Cambridge University Press, 2006.

[LGLM00] Eric Larsen, Stefan Gottschalk, Ming C. Lin, and Dinesh Manocha, *Fast Proximity Queries with Swept Sphere Volumes*, Proceedings of the IEEE International Conference on Robotics and Automation, 2000.

[LK01] S.M. LaValle and J.J. Kuffner, *Rapidly-Exploring Random Trees: Progress and Prospects*, Algorithmic and Computational Robotics: New Direction (B.R. Donald, K.M. Lynch, and D. Rus, eds.), A. K. Peters, Wellesley, 2001, pp. 293–308.

[LLB05] O. Lefebvre, F. Lamiraux, and D. Bonnafous, *Fast Computation of Robot-Obstacle Interactions in Nonholonomic Trajectory Deformation*, Proceedings of the IEEE International Conference on Robotics and Automation, 2005.

[LMKY10] S. Lengagne, P. Mathieu, A. Kheddar, and E. Yoshida, *Generation of Dynamic Multi-Contact Motions: 2D Case Studies*, Proceedings of the IEEE-RAS International Conference on Humanoid Robots, 2010.

[LS93] Gerardo Lafferriere and Hector J. Sussmann, *A Differential Geometric Approach to Motion Planning*, Nonholonomic Motion Planning, Kluwer, 1993, pp. 235–270.

[LT96] Craig T. Lawrence and Andre L. Tits, *Nonlinear Equality Constraints in Feasible Sequential Quadratic Programming*, Optimization Methods and Software **6** (1996), 265–282.

[LYK99] Steven M. LaValle, Jeffrey Yakey, and Lydia Kavraki, *A Probabilistic Roadmap Approach for Systems with Closed Kinematic Chains*, Proceedings of the IEEE International Conference on Robotics and Automation, 1999.

[MA04] Andrew Miller and Peter K. Allen, *Graspit!: A Versatile Simulator for Robotic Grasping*, IEEE Robotics and Automation Magazine **11** (2004), no. 4, 110–122.

[Min11] Hermann Minkowski, *Theorie der Konvexen Korpern, Insbesonder der Begrundung Ihres Oberflchenbegriffs*, Gesammelte Abhandlungen **2** (1911), 131–229.

[MJ00] Kanti V. Mardia and Peter E. Jupp, *Directional Statistics*, Wiley, 2000.

[MLS94] Richard M. Murray, Zexiang Li, and S. Shankar Sastry, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, 1994.

[MN99] Jan R. Magnus and Heinz Neudecker, *Matrix Differential Calculus with Applications in Statistics and Econometrics*, Wiley, 1999.

[Mon88] David J. Montana, *The Kinematics of Contact and Grasp*, International Journal of Robotics Research **7** (1988), no. 3, 17–32.

[MRTT53] T. S. Motzkin, H. Raiffa, G. L. Thomson, and R. M. Thrall, *The Double Description Method*, Contributions to Theory of Games **2** (1953), 51–73.

[Pad99] M. W. Padberg, *Linear Optimization and Extensions*, 2 ed., Springer, 1999.

[RB09] D. Raunhardt and R. Boulic, *Motion Constraint*, The Visual Computer **25** (2009), no. 5-7, 509–518.

[RMBO08] E. Rimon, R. Mason, J. W. Burdick, and Y. Or, *A General Stance Stability Test Based on Stratified Morse Theory With Applicaiton to Quasi-Static Locomotion Planning*, IEEE Transactions on Robotics **24** (2008), no. 3, 626–641.

[SBB09] Joseph Salini, Sebastien Barthelemy, and Philippe Bidaud, *LQP Controller Design for Generic Whole Body Motion*, Climbing and Walking Robots and the Support Technologies for Mobile Machines, 2009.

[Sch07] K. Schittkowski, *QL: A Fortran Code for Convex Quadratic Programming - User's Guide*, Tech. report, Department of Computer Science, University of Bayreuth, 2007.

[SEM05] Siddhartha S. Srinivasa, Michael A. Erdmann, and Matthew T. Mason, *Using Projected Dynamics to Plan Dynamic Contact Manipulation*, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005.

[SI07] M. Saha and P. Isto, *Manipulation Planning for Deformable Linear Objects*, IEEE Transactions on Robotics **23** (2007), no. 6, 1141–1150.

[SLCS04] Thierry Siméon, Jean-Paul Laumond, Juan Cortés, and Anis Sahbani, *Manipulation Planning With Probabilistic Roadmaps*, International Journal of Robotics Research **23** (2004), no. 7-8, 729–746.

[SLL02] T. Simeon, S. Leroy, and J.-P. Laumond, *Path Coordination for Multiple Mobile Robots: A Resolution Complete Algorithm*, IEEE Transactions on Robotics and Automation **18** (2002), 42–49.

[SPK10] Luis Sentis, Jaeheung Park, and Oussama Khatib, *Compliant Control of Multi-Contact and Center of Mass Behaviors in Humanoid Robots*, IEEE Transactions on Robotics **26** (2010), no. 3, 483–501.

[SRM+11] L. Saab, O. Ramos, N. Mansard, P. Soueres, and J. Y. Fourquet, *Generic Dynamic Motion Generation with Multiple Unilateral Constraints*, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2011.

[SSEKP07] Jean-Philippe Saut, Anis Sahbani, Sahar El-Khoury, and Véronique Perdereau, *Dexterous Manipulation Planning Using Probabilistic Roadmaps in Continuous Grasp Subspaces*, Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007.

[SSV08] Bruno Siciliano, Lorenzo Sciavicco, and Luigi Villani, *Robotics: Modelling, Planning and Control*, Springer, 2008.

[Sti07] Mike Stillman, *Task constrained Motion Planning in Robot Joint Space*, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007.

[TvdP98] N. Torkos and M. van de Panne, *Footprint-based Quadruped Motion Synthesis*, Proceedings of Graphics Interface, 1998.

[vdBMGA10] J. van den Berg, S. Miller, K. Goldberg, and P. Abbeel, *Gravity-Based robotic Cloth Folding*, Workshop on Algorithmic Foundations of Robotics, 2010.

[vdBO05] J. van den Berg and M. Overmars, *Prioritized Motion Planning for Multiple Robots*, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005.

[WB06] Andreas Wachter and Lorenz T. Biegler, *On the Implementation of an Interior-Point Filter Line-Search Algorithm for Large-Scale Nonlinear Programming*, Mathematical Programming **106** (2006), 25–57.

[Wie02] Pierre-Brice Wieber, *On the Stability of Walking Systems*, Proceedings of the International Workshop Humanoid and Human Friendly Robotics, 2002.

[Wie05] Pierre-Brice Wieber, *Some Comments on the Structure of the Dynamics of Articulated Motion*, Proceedings of the International Symposium on Fast Motions in Biomechanics and Robotics, 2005.

[Woo94] A. T. A. Wood, *Simulation of the Von Mises Fisher Distribution*, Communications in Statistics. Simulation and Computation **23** (1994), no. 1, 157–164.

[XKL07] Jijie Xu, John Koo, and Zexiang Li, *Finger Gaits Planning for Multifingered Manipulation*, Proceedings of IEEE/RSJ Internatonal Conference on Intelligent Robots and Systems, 2007.

[YBEL05] Eiichi Yoshida, Igor Belousov, Claudia Esteves, and Jean-Paul Laumond, *Humanoid Motion Planning For Dynamic tasks*, Proceedings of the IEEE-RAS International Conference on Humanoid Robots, 2005.

[YC00] Y.Zhuang and J. Canny, *Real-Time Global Deformations*, Proceedings of the Workshop on Algorithmic Foundations of Robotics, 2000.

[YH10] Katsu Yamane and Jessica Hodgins, *Simultaneous Tracking and Balancing of Humanoid Robots for Imitating Human Motion Capture Data*, Proceedings of the IEEE-RAS International Conference on Humanoid Robots, 2010.

[YKEJL06] E. Yoshida, O. Kanoun, C. Esteves-Jaramillo, and J. P. Laumond, *Task-driven Support Polygon Reshaping for Humanoids*, Proceedings of the IEEE-RAS International Conference on Humanoid Robots, 2006.

[YKH04] Katsu Yamane, James Kuffner, and Jessica K Hodgins, *Synthesizing Animations of Human Manipulation Tasks*, ACM Transactions on Graphics (Proc. SIGGRAPH 2004) **23** (2004), no. 3.

[YLE+08] Eiichi Yoshida, Jean-Paul Laumond, Claudia Esteves, Oussama Kanoun, Takeshi Sakaguchi, and Kazuhito Yokoi, *Whole-Body Locomotion, Manipulation and Reaching for Humanoids*, Motion In Games, 2008.

[YSY03] Masahito Yashima, Yoshikazu Shiina, and Hideya Yamaguchi, *Randomized Manipulation Planning for a Multifingered Hand by Switching Contact Modes*, Proceedings of IEEE International Conference on Robotics and Automation, 2003.