Advanced Robotics

ISSN: 0169-1864 (Print) 1568-5535 (Online) Journal homepage: https://www.tandfonline.com/loi/tadr20

# Humanoid Robot Locomotion and Manipulation Step Planning

Karim Bouyarmane & Abderrahmane Kheddar

**To cite this article:** Karim Bouyarmane & Abderrahmane Kheddar (2012) Humanoid Robot Locomotion and Manipulation Step Planning, Advanced Robotics, 26:10, 1099-1126, DOI: 10.1080/01691864.2012.686345

**To link to this article:** https://doi.org/10.1080/01691864.2012.686345

Taylor & Francis
Taylor & Francis Group

*Advanced Robotics 26 (2012) 1099–1126*

**ADVANCED ROBOTICS**
tandfonline.com/tadr

*Full paper*

# Humanoid Robot Locomotion and Manipulation Step Planning

**Karim Bouyarmane* and Abderrahmane Kheddar**

CNRS-AIST UMI3218/CRT JRL, AIST Tsukuba, Ibaraki 305-8568, Japan; CNRS-Montpellier 2 University UMR5506 LIRMM, 34095 Montpellier Cedex 5, France

Received 23 August 2011; accepted 31 December 2011

**Abstract**

We aim at planning multi-contact sequences of stances and postures for humanoid robots. The output sequence defines the contact transitions that allow our robot to realize different kind of tasks, ranging from biped locomotion to dexterous manipulation. The central component of the planning framework is a best-first algorithm that performs a search of the contacts to be added or removed at each step, following an input collision-free guide path, and making calls to an optimization-based inverse kinematics solver under static equilibrium constraints. The planner can handle systems made of multiple robots and/or manipulated objects through a centralized multi-agent approach, opening the way for multi-robot collaborative locomotion and manipulation planning. Results are presented in virtual environments, with discussion on execution on the real robot HRP-2 in an example situation.


**Keywords**

humanoid robot, legged locomotion, dexterous manipulation, inverse kinematics, planning algorithm

## 1. Introduction

Biped locomotion, dexterous hand manipulation, dual-arm manipulation are examples of the many different skills that we expect from a humanoid robot, which in turn define as many respective motion planning problems that need to be solved in order to achieve autonomy for these systems. Despite looking different, all of these motions share a common feature in that they require sequentially establishing and breaking contacts to get from an initial configuration to a goal configuration. For example, a walking humanoid robot successively makes and breaks contacts between its feet and the ground in order to progress forward, and a hand dexter-

* To whom correspondence should be addressed. Present address: ATR Computational Neuroscience Laboratories, 2-2-2 Hikaridai, Seika-cho Soraku-gun, Kyoto 619-0288, Japan; E-mail: karim.bouyarmane@atr.jp

 DOI:10.1080/01691864.2012.686345

ously manipulating an object successively makes and breaks contacts between the fingertips and the surface of the object in order to change its orientation.

Contact-before-motion is a planning approach that first plans the sequence of contacts that will be added or removed at each step of the motion, and then plans the continuous motion that will be made of elementary step motions on fixed contact configurations [1,2]. As opposed to the footstep planning problem [3–7], this approach focuses on non-gaited acyclic motion. It targets higher level "intelligence" of the robots by erasing the knowledge-based specification of a bipedal or quadrupedal gait. The two stages of contact-before-motion planning can either be interleaved, such as in multi-modal planning [8], or decoupled, such as in our previous tackling of the problem [1,9].

In this paper, following the decoupled strategy, we focus on the first stage: the planning of the sequence of stances and postures. Our contribution with regard to previous studies of this problem lies in the generalization of the framework to multi-robot, multi-object systems. This generalization allows for a common description and treatment of locomotion and manipulation problems, either for a single robot or for multiple collaborating robots. The approach retained for the multi-robot systems is a centralized one. Decentralized strategies such as prioritized planning [10,11] or fixed-path coordination [12] 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 set, made of elements called stances.

The algorithm we use is a best-first search algorithm that relies on an inverse kinematics-and-statics solver designed to generate static equilibrium configurations. These two modules are introduced as independent stand-alone modules, respectively, in [13,14]. In [15], we briefly present them in an integrated framework. This paper constitutes an extension of [15]. Our main added value to this previously published work is to detail how these elementary building blocks assemble together to yield the complete planning framework. We additionally tackle the problem of collision-avoidance, absent from these previous publications, and demonstrate how it can be integrated as a constraint in the generated postures. One last addition is to show how the results from this approach are actually used in dynamic simulation or on a real robot.

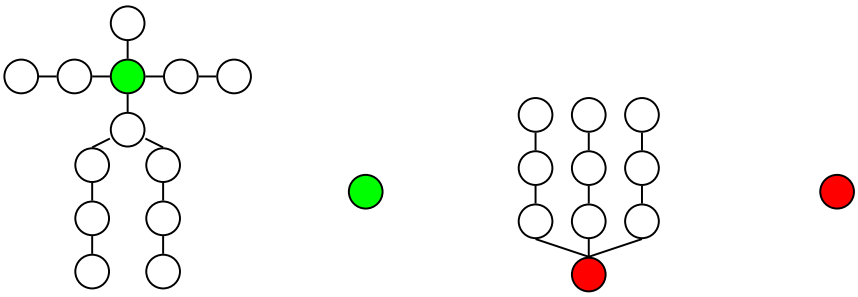
The rest of the paper is organized as follows. We first introduce the systems we consider and the definitions and notations that will help us formulate the problem in a general enough scheme (Sections 2 and 3). We then present the two main modules that make up the planner: the search algorithm (Section 4) and the inverse kinematics-and-statics solver (Section 5). Finally we present different results from the implementation and execution of the planner (Section 6), and discuss how to generate motions from these results (Section 7). In the Appendix section, Appendix A derives the gradients of the non-linear optimization constraints, Appendix B discusses the complexity of the algorithm, and Appendix C contains a synthetic overview of all the notations used throughout the paper.

## 2. Systems

The systems we consider are made of a number $N$ of *entities*. An entity is either a fixed-base robot, a floating-base robot, or a rigid object. We also consider the environment as a particular case of an entity. The entities are indexed by $r$ (for "robot") which varies in the set $\{0, \ldots, N\}$. $r=0$ refers to the environment, while $r \in \{1, \ldots, N\}$ is a robot or an object. For example, a humanoid robot is a floating-base robot, a dexterous hand is a fixed-base robot, a dual-arm robot is a fixed-base robot, etc. Rigid objects are particular cases of floating-base robot consisting of only one link, and the environment is a particular case of a fixed-base robot also consisting of one link. Each of these entities is represented as a kinematic tree, with the root node being either fixed or floating (see Fig. 1). For $r \in \{1, \ldots, N\}$ we denote by $q_r$ the configuration of the robot $r$ (vector of generalized coordinates) and by $\mathcal{C}_r$ its configuration space. For $r \in \{0, \ldots, N\}$ we denote $L_r$ the number of links of the robot, $\Theta_r$ the number of actuated degrees of freedom (DOF), and $\theta_r \in \mathbb{R}^{\Theta_r}$ the vector of joint angles. For a rigid object or the environment, we have $L_r = 1$ and $\Theta_r = 0$. The links are indexed in $\{0, \ldots, L_r - 1\}$, with the link 0 being the root link, and the joints are indexed in $\{1, \ldots, \Theta_r\}$. Finally, if $r$ refers to a floating-base robot, $\xi_r \in \mathbb{R}^3$ and $\phi_r \in \mathbb{R}^4$ denote respectively the position and the unit-quaternion-parametrized orientation of the root link. As a summary ($\mathrm{SE}(3)$ is the special Euclidean group):

$$\mathcal{C}_r = \begin{cases} \mathbb{R}^{\Theta_r} & \text{if } r \text{ is a fixed-base robot} \\ \mathrm{SE}(3) \times \mathbb{R}^{\Theta_r} & \text{if } r \text{ is a floating-base robot} \\ \mathrm{SE}(3) & \text{if } r \text{ is a rigid object.} \end{cases} \quad (1)$$

and



(a) Floating-base robot (b) Rigid object (c) Fixed-base robot (d) The environment

**Figure 1.** Kinematic trees. Red nodes are root nodes of fixed base entities. Green nodes are root nodes of floating base entities.

$$q_r = \begin{cases} (\theta_r) & \text{if } r \text{ is a fixed-base robot} \\ (\xi_r, \phi_r, \theta_r) & \text{if } r \text{ is a floating-base robot} \\ (\xi_r, \phi_r) & \text{if } r \text{ is a rigid object.} \end{cases} \quad (2)$$

For every link $l$ of the entity $r$, $\Omega_{r,l}(q_r)$, and $R_{r,l}(q_r)$ denote respectively the global-frame-expressed position and the orientation matrix of the link. Their expressions in the local frame of the root body are respectively denoted $\Omega^0_{r,l}(\theta_r)$ and $R^0_{r,l}(\theta_r)$.

We take a centralized multi-agent approach, so all the robots and the objects in $\{1, \ldots, N\}$ form one unique system, the configuration space of which is

$$\mathcal{C} = \prod_{r=1}^{N} \mathcal{C}_r, \quad (3)$$

and an element of $\mathcal{C}$ is denoted $q$

$$q = (q_1, \ldots, q_N). \quad (4)$$

Let us now define the contact surface patches. For each entity $r \in \{0, \ldots, N\}$ and each link $l \in \{0, \ldots, L_r - 1\}$ we define a number $S_{r,l}$ of planar surface patches that cover the link. A surface is thus encoded into the triplet $(r, l, s) \in \{0, \ldots, N\} \times \{0, \ldots L_r - 1\} \times \{1, \ldots, S_{r,l}\}$, but for short we will just call it $s$. For each such surface $s$, we denote $O_s$ its origin (a point belonging to the surface) expressed in the local frame of $l$, $O_s(q_r)$ its expression in the world frame $(\vec{x}_s(q_r), \vec{y}_s(q_r), \vec{z}_s(q_r))$, a frame attached to the link $l$ such that $\vec{z}_s$ is the outward normal to the surface and $(\vec{x}_s, \vec{y}_s)$ are tangential to it. We also require that $s$ be a convex polygon, the number of vertices of which is $V_s$. The vertices expressed in the local frame attached to the link $l$ are denoted $a_{s,1}, \ldots, a_{s,V_s}$. Their $q_r$-dependent expressions in the world (inertial) frame are denoted $A_{s,1}(q_r), \ldots, A_{s,V_s}(q_r)$.

## 3. Problem Formulation

A contact is established between two surfaces $s_1$ and $s_2$ belonging to two different links $l_1$ and $l_2$ (the entities $r_1$ and $r_2$ can be equal). We denote $c$ such a contact and $E_{\text{ctc}}$ the set of all possible contacts. $(x_c, y_c, \vartheta_c) \in \text{SE}(2)$ denotes the relative position and orientation of the two surfaces $s_1$ and $s_2$ when the contact $c$ is established (see Fig. 2).

A stance $\sigma$ is a finite set of contacts $\sigma = \{c_1, \ldots, c_{n_\sigma}\}$, such that each surface appears at most once. We denote the set of all possible stances $\Sigma$ (which is thus a subset of $2^{E_{\text{ctc}}}$). For every stance $\sigma$ we denote $n_\sigma = \text{card}(\sigma)$ the number of

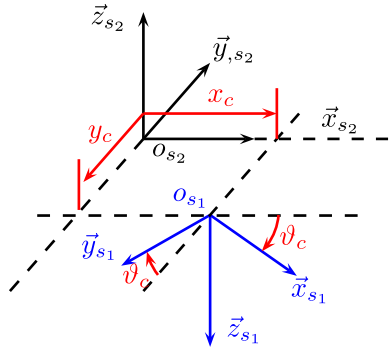



**Figure 2.** Contact parameters.

contacts in $\sigma$. A stance $\sigma'$ is said to be adjacent to $\sigma$ if it differs from $\sigma$ by exactly one contact. We denote $\mathrm{Adj}^+(\sigma)$ the set of stances $\sigma'$ that add one contact to $\sigma$, i.e. $n_{\sigma'} = n_\sigma + 1$ and $\sigma \subset \sigma'$, and we denote $\mathrm{Adj}^-(\sigma)$ the set of stances $\sigma'$ that remove one contact from $\sigma$, i.e. $n_{\sigma'} = n_\sigma - 1$ and $\sigma' \subset \sigma$. Finally $\mathrm{Adj}(\sigma) = \mathrm{Adj}^+(\sigma) \cup \mathrm{Adj}^-(\sigma)$ is the set of all the stances adjacent to $\sigma$. Furthermore, we define on $\mathrm{Adj}^+(\sigma)$ the equivalence relation denoted $\sim_\sigma$, such that $\sigma_1 \sim_\sigma \sigma_2$ if and only if $\sigma_1$ and $\sigma_2$ add respectively the contacts $c_1$ and $c_2$ to $\sigma$ with $c_1$ and $c_2$ being contacts between the same pairs of surfaces (only the relative positions of the contacts $(x_{c_1}, y_{c_1}, \vartheta_{c_1})$ and $(x_{c_2}, y_{c_2}, \vartheta_{c_2})$ might differ). The quotient space is denoted $\mathrm{Adj}^+(\sigma)_{/\sim_\sigma}$. An equivalence class $[\sigma']_{\sim_\sigma}$ is a set of stances that add a given contact $c$ to $\sigma$ for which the position $(x_c, y_c, \vartheta_c)$ varies in $\mathrm{SE}(2)$.

We have now defined the configuration space $\mathcal{C}$ and the stance set $\Sigma$. We define a mapping between the two $p : \mathcal{C} \to \Sigma$ which maps every configuration $q$ of the system to the set of contacts that the system happens to be establishing when put in that configuration. Inversely, for a given $\sigma \in \Sigma$, the inverse kinematics sub-manifold $\mathcal{Q}_\sigma = p^{-1}(\{\sigma\})$ is the set of all configurations that realize the stance $\sigma$. The subset $\mathcal{F}_\sigma \subset \mathcal{Q}_\sigma$ (for "feasible") is made of the configurations that realize the stance and that are subject to physical constraints. These constraints are:

- static equilibrium,
- contact forces within friction cones,
- joint limits,
- torque bounds,
- collision-free.

For two adjacent stances $\sigma$ and $\sigma'$ such that $\mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'} \neq \emptyset$, a configuration $q \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$ will be called a *transition configuration* between $\sigma$ and $\sigma'$.

Such transition configurations satisfy the following property: if $\sigma_{\text{sup}} = \max(\sigma, \sigma')$ and $\sigma_{\text{inf}} = \min(\sigma, \sigma')$, for the order defined by the inclusion relation, then the configuration $q$ is a transition configuration between $\sigma$ and $\sigma'$ if and only if $q$ realizes geometrically the contacts of $\sigma_{\text{sup}}$ ($q \in \mathcal{Q}_{\sigma_{\text{sup}}}$) while satisfying the physical constraints of $\sigma_{\text{inf}}$ ($q \in \mathcal{F}_{\sigma_{\text{inf}}}$), i.e. if and only if $q \in \mathcal{Q}_{\sigma_{\text{sup}}} \cap \mathcal{F}_{\sigma_{\text{inf}}}$.

Finally we can formulate our problem: given an initial and goal stance $\sigma_{\text{init}}$ and $\sigma_{\text{goal}}$ in $\Sigma$, find a sequence of stances $(\sigma_1, \ldots, \sigma_n)$ such that:

- $\sigma_1 = \sigma_{\text{init}}$,
- $\sigma_n = \sigma_{\text{goal}}$,
- $\forall i \in \{1, \ldots, n-1\} \quad \sigma_{i+1} \in \text{Adj}(\sigma_i)$ and $\mathcal{F}_{\sigma_i} \cap \mathcal{F}_{\sigma_{i+1}} \neq \emptyset$.

## 4. Search Algorithm

To explore $\Sigma$ we use a greedy best-first approach [16]. Starting from $\sigma := \sigma_{\text{init}}$, we explore the stances $\sigma'$ that are adjacent to $\sigma$ looking for transition configurations, then we take the most promising one $\sigma^*$, and we reiterate the process by setting $\sigma := \sigma^*$, until finding a $\sigma'$ that is close enough to $\sigma_{\text{goal}}$.

Two data structures are needed in order to write this algorithm: a priority queue $\mathcal{P}$ and an exploration tree $\mathcal{T}$. The priority queue $\mathcal{P}$ is defined through a cost function $f$ that assigns to each pair of stance $\sigma$ and realizing configuration $q$ a cost $f(\sigma, q)$. To define $f$ we use a *guide path* [17] built in a preprocessing step by applying path planning techniques on $\mathcal{C}_{\text{free}}$ [18], the trace of which will be denoted $\mathcal{G}$ (the trace is also known as the image of the path in $\mathcal{C}$). This guide path is constructed through a trade-off between being collision-free and at the same time being close enough to the obstacles as they are possible candidate for contact supports. The cost is defined as the distance of the configuration $q$ to this trace $\mathcal{G}$. The contacts will thus be in priority searched around the guide path. The tree $\mathcal{T}$ basically consists of nodes that are the pair of stances with realizing configuration and edges are drawn between nodes that encode adjacent stances between which a transition configuration exists.

Algorithm 1 gives an overview of the search process. Some technicalities such as avoiding infinite loop by checking that when a new stance is added it does not already belong to $\mathcal{T}$, and backtracking when $\mathcal{P}$ becomes empty before reaching the goal by generating multiple configurations $q \in \mathcal{F}_\sigma \cap \mathcal{Q}_{[\sigma']_{\sim\sigma}}$, are not made explicit for clarity.

**Alrorithm 1** Best-first search
1: push $\sigma_{\text{init}}$ into $\mathcal{P}$ and $\mathcal{T}$
2: **repeat**
3: pop best stance $\sigma$ from $\mathcal{P}$
4: **for** $[\sigma']_{\sim_\sigma} \in \text{Adj}^+(\sigma)_{/\sim_\sigma}$ do
5: call the inverse kinematics-and-statics solver to find $q \in \mathcal{F}_\sigma \cap \mathcal{Q}_{[\sigma']_{\sim_\sigma}}$
6: **if** found $q$ **then**
7: push $\sigma' = p(q)$ into $\mathcal{P}$ and add $\sigma'$ to $\mathcal{T}$ as a son of $\sigma$
8: **end if**
9: **end for**
10: **for** $\sigma' \in \text{Adj}^-(\sigma)$ do
11: call the inverse kinematics-and-statics solver to find $q \in \mathcal{F}_{\sigma'} \cap \mathcal{Q}_\sigma$
12: **if** found $q$ **then**
13: push $\sigma'$ into $\mathcal{P}$ and add $\sigma'$ to $\mathcal{T}$ as a son of $\sigma$
14: **end if**
15: **end for**
16: **until** $\mathcal{T}$ contains a stance close enough to $\sigma_{\text{goal}}$

In Algorithm 1 we make calls to an inverse kinematics-and-statics solver at lines 5 and 11. Let us consider a stance $\sigma = \{c_1, \ldots, c_{n_\sigma}\}$. A stance $\sigma^-$ in $\text{Adj}^-(\sigma)$ will have the form $\sigma^- = \{c_1, \ldots, c_{i_0-1}, c_{i_0+1}, \ldots, c_{n_\sigma}\}$ where we have removed the contact $c_{i_0}$ from $\sigma$, while a stance $\sigma^+$ in $\text{Adj}^+(\sigma)$ will have the form $\sigma^+ = \{c_1, \ldots, c_{n_\sigma}, c_{n_\sigma+1}\}$ where we have added the contact $c_{n_\sigma+1}$ to $\sigma$.

- *Case 1*. Finding a configuration $q$ in $\mathcal{F}_\sigma \cap \mathcal{Q}_{[\sigma^+]_{\sim_\sigma}}$ amounts to finding a configuration for which the contacts $c_1, \ldots, c_{n_\sigma}$ are realized at their fixed positions with contact forces being applied at these contacts, and for which the contact $c_{n_\sigma+1}$ is realized without its position being fixed and without applying a contact force at it. The position $(x_{c_{n_\sigma+1}}, y_{c_{n_\sigma+1}}, \vartheta_{c_{n_\sigma+1}})$ of the contact $c_{n_\sigma+1}$ is to be decided by the solver.
- *Case 2*. Finding a configuration $q$ in $\mathcal{Q}_\sigma \cap \mathcal{F}_{\sigma^-}$ amounts to finding a configuration for which all the contacts $c_1, \ldots, c_{n_\sigma}$ are realized at their fixed positions, but with contact forces only applied at the contacts $c_1, \ldots, c_{i_0-1}, c_{i_0+1}, \ldots, c_{n_\sigma}$. No contact forces are applied at the contact $c_{i_0}$.

To find these configurations we use an optimization formulation. We will first explain how to find a configuration $q$ in $\mathcal{F}_\sigma$, i.e. a configuration that realizes the contacts $c_1, \ldots, c_{n_\sigma}$, and for which contact forces are applied at all the contacts. The two cases above will then straightforwardly follow: in the derivations hereunder we just do not need to take into account the contact on which the forces are not applied ($c_{n_\sigma+1}$ in case 1 and $c_{i_0}$ in case 2) in the definition of the contact forces variables and when writing the static equilibrium equation.

## 5. Inverse Kinematics-and-Statics Solver

Our approach is to solve for the configurations and the contact forces simultaneously, as opposed to a rejection sampling approach that would sample random configurations [19–21] and keep only the static equilibrium constraint-satisfying ones [22].

We start from the stance $\sigma = \{c_1, \ldots, c_{n_\sigma}\}$. For each contact $c_i$, let $s_{i1}$ and $s_{i2}$ denote the two surfaces between which the contact is established, $l_{i1}$ and $l_{i2}$ the corresponding indexes of the links to which they belong, and $r_{i1}$ and $r_{i2}$ their respective entities (robot/object/environment).

Without loss of generality (we can reorder $s_{i1}$ and $s_{i2}$), we suppose that the area of the surface $s_{i1}$ is less than the area of the surface $s_{i2}$, so that we can write $s_{i1} \subset s_{i2}$ once the contact is established at the solution. Recall that the vertices of the smaller area surface $s_{i1}$ expressed in the local frame attached to the link $l_{i1}$ are $a_{s_{i1},1}, a_{s_{i1},2}, \ldots, a_{s_{i1},V_{s_1}}$. At each of these vertices we linearize the friction cone between $s_{i1}$ and $s_{i2}$ into a polyhedral cone with a finite number $K$ of vertices, and we denote $U_{i,1}(q_{r_{i1}}), \ldots, U_{i,K}(q_{r_{i1}})$ the $q_{r_{i1}}$-dependent expressions of its unit generators in the world (inertial) frame. The resulting contact force applied by the link $l_{i2}$ on the link $l_{i1}$ at the vertex $a_{s_{i1},j}$ will be a nonnegative linear combination of these generators

$$f_{i,j}(q_{r_{i1}}) = \sum_{k=1}^{K} \lambda_{i,j,k} U_{i,k}(q_{r_{i1}}). \tag{5}$$

Let us now denote $\lambda = (\lambda_{i,j,k})_{i,j,k}$ the vector of all coefficients. The variables of the optimization problem we will write are $(q, \lambda)$.

We now write the constraints of our problem. The joint limit constraint is straightforward

$$\forall r \text{ such that } r \text{ is a robot: } \theta_{r,\min} \leq \theta_r \leq \theta_{r,\max}, \tag{6}$$

as well as the unilateral-force constraint

$$\lambda \geq 0. \tag{7}$$

The following constraints set the relative positions of the links $l_{i1}$ and $l_{i2}$, for all $i \in \{1, \ldots, n_\sigma\}$:

$$\vec{z}_{s_{i1}}(q_{r_{i1}}) + \vec{z}_{s_{i2}}(q_{r_{i2}}) = \vec{0}, \tag{8}$$

$$O_{s_{i1}}(q_{r_{i1}})^T \vec{z}_{s_{i2}}(q_{r_{i2}}) = 0, \tag{9}$$

$$O_{s_{i1}}(q_{r_{i1}})^T \vec{x}_{s_{i2}}(q_{r_{i2}}) = x_{c_i}, \tag{10}$$

$$O_{s_{i1}}(q_{r_{i1}})^T \vec{y}_{s_{i2}}(q_{r_{i2}}) = y_{c_i}, \tag{11}$$

$$\vec{x}_{s_{i1}}(q_{r_{i1}})^T \vec{x}_{s_{i2}}(q_{r_{i2}}) = \cos(\vartheta_{c_i}), \tag{12}$$

$$\vec{x}_{s_{i1}}(q_{r_{i1}})^T \vec{y}_{s_{i2}}(q_{r_{i2}}) = \sin(\vartheta_{c_i}). \tag{13}$$

In case 1 (when adding a contact), the position of the contact $c_{n_\sigma+1}$ is not decided, so the constraints (10) and (11) are relaxed for this contact and left as linear inequality constraints (the linear inequalities defining the polygon shape of the surface $s_{i2}$), and the constraints (12) and (13) are dropped.

Let us now write the static equilibrium constraints. We will write one static equilibrium constraint for every floating-base robot $r$ (so in particular for rigid objects). Let $I$ be the index set $\{1, \ldots, n_\sigma\}$. The set $I$ can be partitioned into three disjoint subsets. $I_1(r) = \{i \mid r_{i1} = r\}$ is the subset of $I$ made of the indices of the contacts in which a surface of $r$ is involved as the smaller area surface; $I_2(r) = \{i | r_{i2} = r\}$ is the subset of $I$ made of the indices of the contacts in which a surface of $r$ is involved as the larger area surface; $I_3(r) = \{i |\ r_{i1} \neq r$ and $r_{i2} \neq r\}$ is the subset of $I$ made of the indices of the contacts which do not involve any surface from $r$. The expression of the moments of the forces applied on $r$ through the contact $c_i$ will depend on whether $i \in I_1(r)$ or $i \in I_2(r)$. In the former case, $r = r_{i1}$, the application points expressed in the local frame of $l_{i1}$ are simply the points $a_{s_{i1},j}$ where $j$ varies. However, in the second case, $r = r_{i2}$, the application points, that are still $a_{s_{i1},j}$, need to be expressed in the local frame of $l_{i2}$ in order to compute the torques resulting from these forces on the joints of $r$. Let us denote these expressions, which depend on $q_{r_{i1}}$ and $q_{r_{i2}}, a'_{s_{i2},j}(q_{r_{i1}}, q_{r_{i2}})$,

$$a'_{s_{i2},j}(q_{r_{i1}}, q_{r_{i2}}) = R_{r_{i2},l_{i2}}(q_{r_{i2}})^T (A_{s_{i1},j}(q_{r_{i1}}) - \Omega_{r_{i2},l_{i2}}(q_{r_{i2}})). \tag{14}$$

Furthermore, let us denote $\Upsilon_{r,l}(q_r, b)$ the following Jacobian matrix, where $b$ is any vector in $\mathbb{R}^3$,

$$\Upsilon_{r,l}(q_r, b) = \frac{\partial [R_{r,0}(q_r)^T ((\Omega_{r,l}(q_r) + R_{r,l}(q_r)b) - \Omega_{r,0}(q_r))]}{\partial \theta_r}. \tag{15}$$

We can now write the static equilibrium equations

$$\sum_{i\in I_1(r)}\sum_{j=1}^{V_{s_{i1}}} f_{i,j} - \sum_{i\in I_2(r)}\sum_{j=1}^{V_{s_{i1}}} f_{i,j} + m_r g = 0, \tag{16}$$

$$\sum_{i\in I_1(r)}\sum_{j=1}^{V_{s_{i1}}} A_{s_{i1},j}\times f_{i,j} - \sum_{i\in I_2(r)}\sum_{j=1}^{V_{s_{i1}}} A_{s_{i1},j}\times f_{i,j} + C_r\times m_r g = 0, \tag{17}$$

$$\begin{aligned}\tau_r = &-\sum_{i\in I_1(r)}\sum_{j=1}^{V_{s_{i1}}} \Upsilon_{r_{i1},l_{i1}}(q_{r_{i1}}, a_{s_{i1},j})^T f_{i,j} \\ &+\sum_{i\in I_2(r)}\sum_{j=1}^{V_{s_{i1}}} \Upsilon_{r_{i2},l_{i2}}(q_{r_{i2}}, a'_{s_{i2},j}(q_{r_{i1}}, q_{r_{i2}}))^T f_{i,j} - \frac{\partial C_r}{\partial \theta_r}^T m_r g,\end{aligned} \tag{18}$$

where $\tau_r \in \mathbb{R}^{\Theta_r}$ is the vector of actuation torques, $C_r(q_r)$ is the position of the center of mass of the robot, $m_r$ is its mass, and $g$ is the gravity vector. Equation (18) allows us to write $\tau_r$ as a function of $(q, \lambda)$ and thus allows us to write the torque limit constraint as

$$\tau_{r,\min} < \tau_r(q,\lambda) < \tau_{r,\max}. \tag{19}$$

Collision-avoidance constraints are set between any two links $l_{\text{nc1}}$ and $l_{\text{nc2}}$ that are not connected by a joint in the kinematic-tree representation of the system. Collision-avoidance between connected links is implicitly considered in the joint limit constraint (6). Let $\mathscr{D}$ be a signed distance between two strictly convex bounding volumes of the links $l_{\text{nc1}}$ and $l_{\text{nc2}}$, 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) [23]. By specifying the corresponding support functions, the enhanced GJK [24] collision-detection algorithm allows us to compute such a continuously differentiable signed distance. The collision-avoidance constraint is thus simply written

$$\mathscr{D}(\mathcal{B}_{\text{nc1}}, \mathcal{B}_{\text{nc2}}) > 0. \tag{20}$$

Figure 3 shows an example of how this constraint generates different resulting configurations corresponding to different local minima.

We now write the target function we want to optimize, we denote it $\text{obj}(q, \lambda)$. The guide path that was computed through collision-free path planning [17] is made of a number $M$ of milestone configurations $q_{\text{ref},1}, \ldots, q_{\text{ref},M}$ (these are the nodes of the tree or graph structure that was constructed in the PRM [25] or

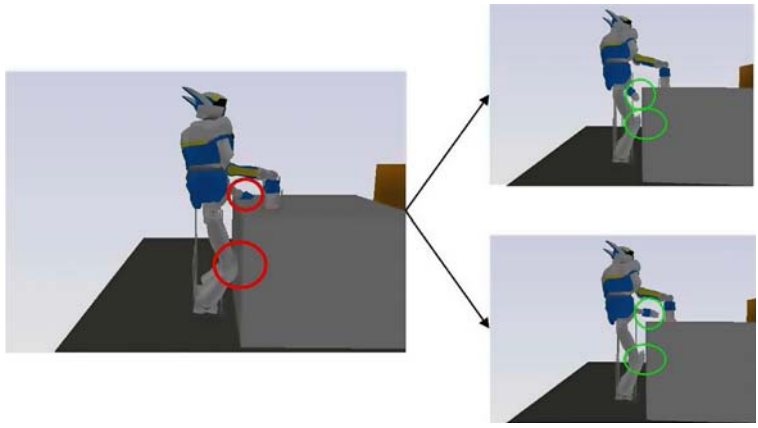


**Figure 3.** Collision-avoidance constraint. The left figure shows a configuration generated without collision-avoidance constraints. The two right figures show two possible solutions corresponding to two different local minima.

RRT [26] process and from which the collision-free path is extracted). Such milestones carry information about major global changes in the configuration of the system $q$ along the guide path. When executing the search Algorithm 1, we maintain a global integer variable $\iota$ that is initialized to 1 and incremented by 1 whenever $q_{\text{ref},\iota}$ is reached (i.e. when the search tree $\mathcal{T}$ contains a configuration close enough to $q_{\text{ref},\iota}$). The search succeeds and Algorithm 1 stops when reaching $q_{\text{ref},M} = q_{\text{goal}}$. The objective function of the inverse kinematics-and-statics solver will be composed of four weighted components:

- a global configuration component

$$\text{obj}_1(q, \lambda) = \| q - q_{\text{ref},\iota} \|^2, \tag{21}$$

- a force minimization component

$$\text{obj}_2(q, \lambda) = \| \lambda \|^2, \tag{22}$$

- a torque minimization component

$$\text{obj}_3(q, \lambda) = \| \tau(q, \lambda) \|^2, \tag{23}$$

- a last component $\text{obj}_4(q, \lambda)$ used when solving case 1 (adding a contact, see Section 4) and that will steer the position of the nonfixed contact $c_{n_\sigma+1}$ to the position of the corresponding contacting links in the currently targeted milestone $q_{\text{ref},\iota}$, this is the component which "pulls" the foot to advance forward when walking for example. The role of this component is to position the contact $c_{n_\sigma+1}$ as close as possible to the final stance while still satisfying

the constraints (8) and (9); it replaces the constraints (10)–(13) which are not enforced in case of the added contact.

$$\text{obj}_4(q,\lambda) = \| O_{s_{n_\sigma+1,1}}(q) - O_{s_{n_\sigma+1,1}}(q_{\text{ref},\iota})\|^2 + \| O_{s_{n_\sigma+1,2}}(q) - O_{s_{n_\sigma+1,2}}(q_{\text{ref},\iota})\|^2. \quad (24)$$

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

$$\text{obj}(q,\lambda) = \begin{cases} \sum_{i=1}^{4} w_i \ \text{obj}_i(q,\lambda) & \text{if case 1,} \\ \sum_{i=1}^{3} w_i \ \text{obj}_i(q,\lambda) & \text{if case 2.} \end{cases} \quad (25)$$

Finally at every call to the inverse kinematics-and-statics problem we solve the following optimization problem

$$\begin{gathered} \min_{(q,\lambda)} \text{obj}(q,\lambda) \\ \text{subject to } (6),(7),(8),(9),(10),(11),(12),(13),(16),(17),(19), \text{ and } (20). \end{gathered} \quad (26)$$

## 6. Results

We applied our planning method to several scenarios that are representative of different tasks involving humanoid robots. These scenarios are

- Biped locomotion with use of arms for support (Fig. 5).
- Dexterous manipulation by a fixed-palm four-finger hand (Fig. 6).
- Dual-arm manipulation (Fig. 7).

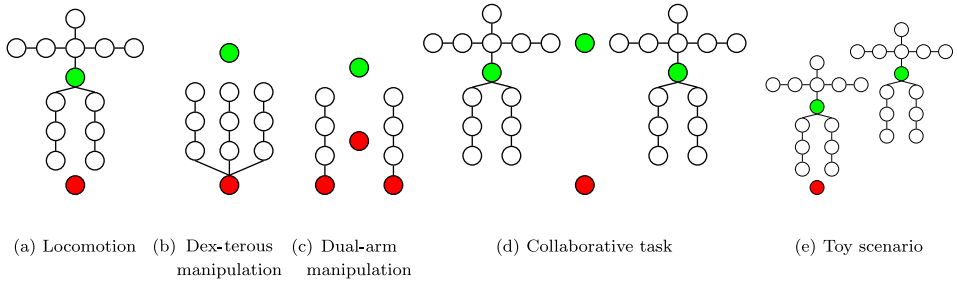



**Figure 4.** Kinematic representations of the systems for each scenario. These representations are simplified for the sake of clarity. The actual model of humanoid robot is HRP-2 [37] and has 30 internal DOFs. The fingers of the dexterous hand and the arms of the dual-arm robot are 6-DOF manipulators. Our implementation is kept generic so that any model of an articulated robot can be used.

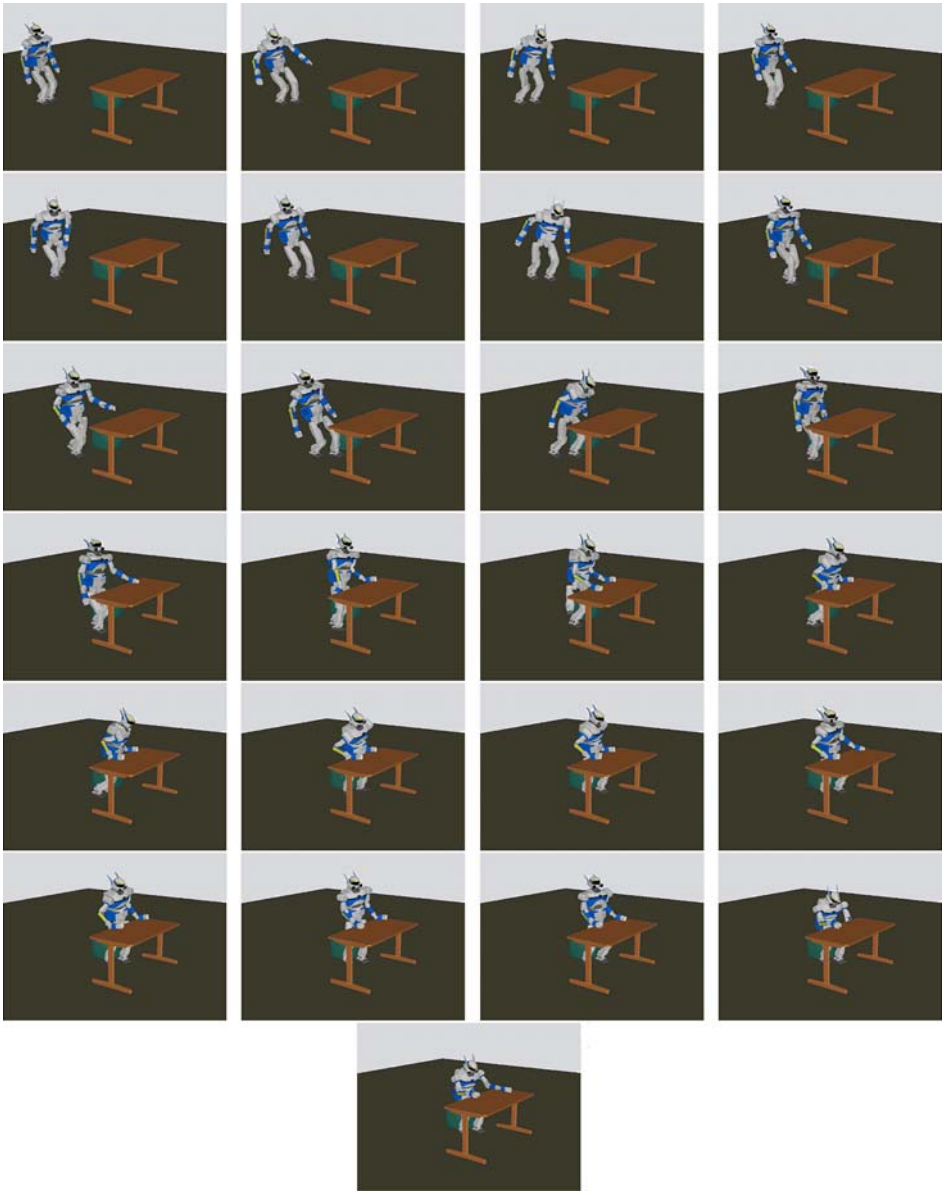


**Figure 5.** Locomotion. The objective is to sit down at the desk. The initial stance is the robot standing about 2 m away from the desk. The final stance is the robot sitting with contacts of its tights with the chair, its feet with the floor, and its forearms with the desk. The motion comprises a first phase in which the robot bipedly walks toward the desk followed by a phase that makes the robot properly sit down.

- Collaborative task (Fig. 8).
- Toy scenario with unrealistic physical limitations to illustrate further theoretic capabilities of the planner (Fig. 9).

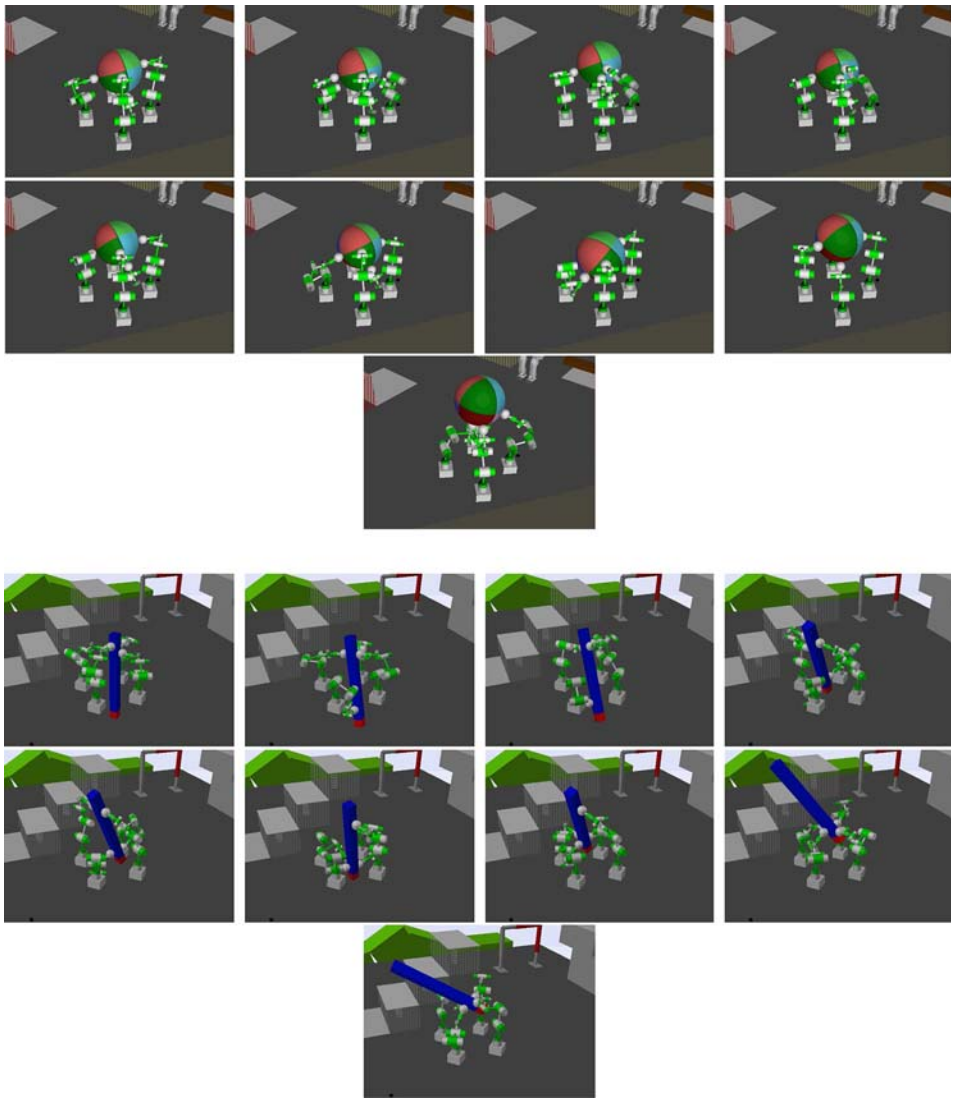


**Figure 6.** Dexterous manipulation. The objective in the first experiment is to rotate the ball upside down. The initial stance is the four fingers holding the ball from its "southern hemisphere." The goal stance is the four fingers holding the ball from its "northern hemisphere." Both stances are 4-contact stances. The main difficulty resides in keeping the force closure condition at every step. This is implicitly taken into account by the way the inverse kinematics-and-statics problem is formulated. In the second experiment the objective is to hold the pen from the red end.

Figure 4 shows the kinematic-tree representations of these systems.

The following figures show the obtained sequence in each case, using the IPOPT [27] software as a non-linear constrained optimization solver.

We have made some re-adjustment of the algorithm and inverse solver in two of these scenarios. In the dexterous manipulation problem the spherical surface is approxi-

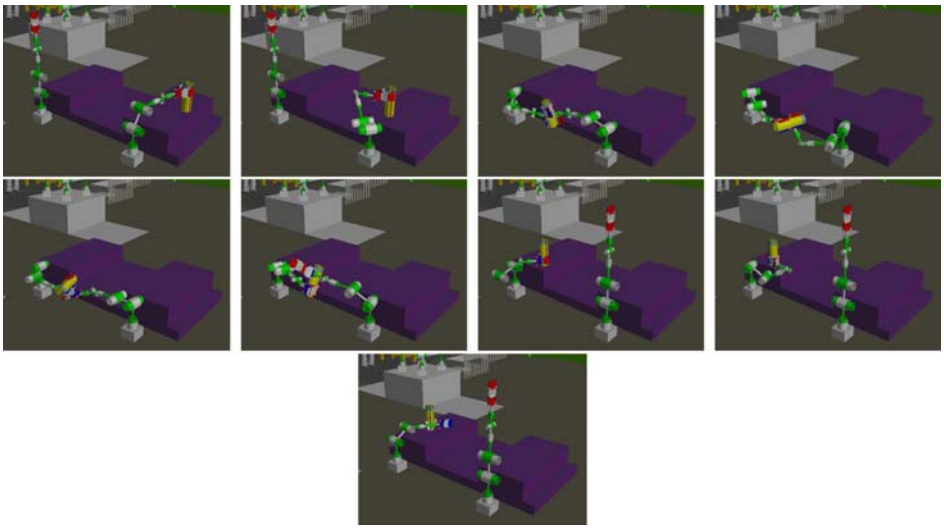


**Figure 7.** Dual arm manipulation. The objective is to bring the cylinder from the first platform to the second platform. The second platform is outside the workspace of the first arm. The planner finds a solution in which it needs to transfer the cylinder to the second arm. The initial stance is made of a contact between one of the planar surfaces of the cylinder and the first platform. The final stance is made of a contact between the same planar 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.

mated by an 20-face polyhedron, and in the collaborative task scenario the contacts between the hands of the robots and the handles of the table are tagged as being *bilateral*, which translates in the inverse solver by replacing the friction cones for this contact by $\mathbb{R}^3$ and dropping the corresponding components of the constraint (7).

The friction coefficient was set to 1 for all the scenarios. Although this might seem to be over-simplification, it does not really influence the feasibility of the stances when the contacts are horizontal or almost horizontal as it turns out to be the case in most of the presented scenarios. Two scenarios in which the friction coefficient is particularly influential are the dexterous and the dual-arm manipulation scenarios. In the latter, the grasping is performed by means of two frictional unilateral contacts rather than one bilateral contact as it is coarsely considered in the collaborative carrying scenario for the grasping of the handles of the table by the grippers of the robots. Since this coefficient would strongly depend on the materials (cover of the fingers, the grippers, the ball, the cylinder) we assumed that a coefficient of 1 would be representative though too small coefficients result in infeasible problems.

As for computation time, the range is in the order of 10–20 min, performed on a 3.06 GHz Pentium IV system with 1 GB of RAM memory. The executable program was compiled with a Microsoft Visual C++ 8.0 compiler from a C++ implementation of the algorithm.

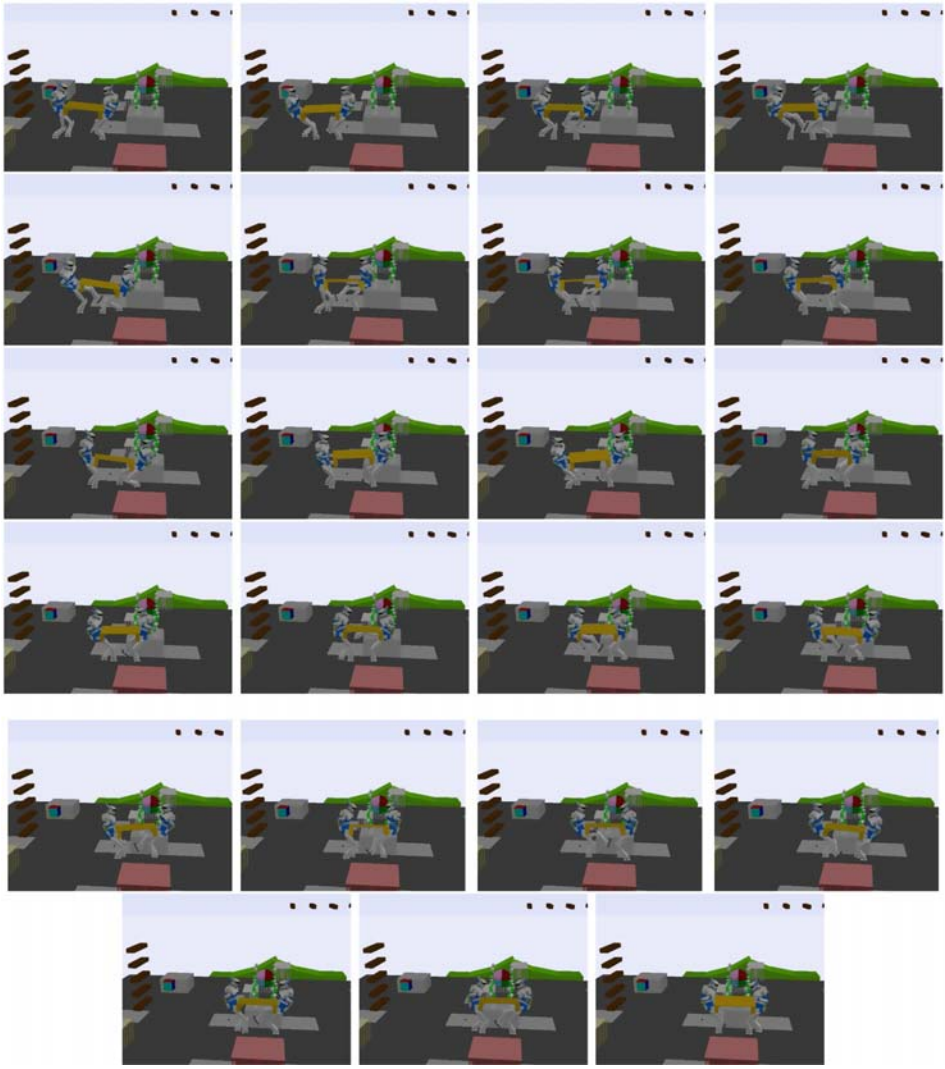


**Figure 8.** Collaborative task. The objective is for the two robots to transport the table together. The contacts between the hands of the robots and the handles of the table are bilateral contacts, which means that they are not subject to unilaterality constraints. They are also required not to be removed throughout the search. So all the searched stances contain at least these four contacts. Other contacts are made and broken between the feet of the robots and the floor. The initial stance and final stances are 1.5 m away from each other. The motion can be compared to a quadruped gait.

## 7. Dynamics and Real-World Application

The output of Algorithm 1 is not a motion, but rather a sequence of key postures that encode the contact transitions of the system if it were to realize a motion that goes from the initial to the goal stance. We now discuss the full dynamic motion generation taking this output sequence as an input of a motion generation tool.

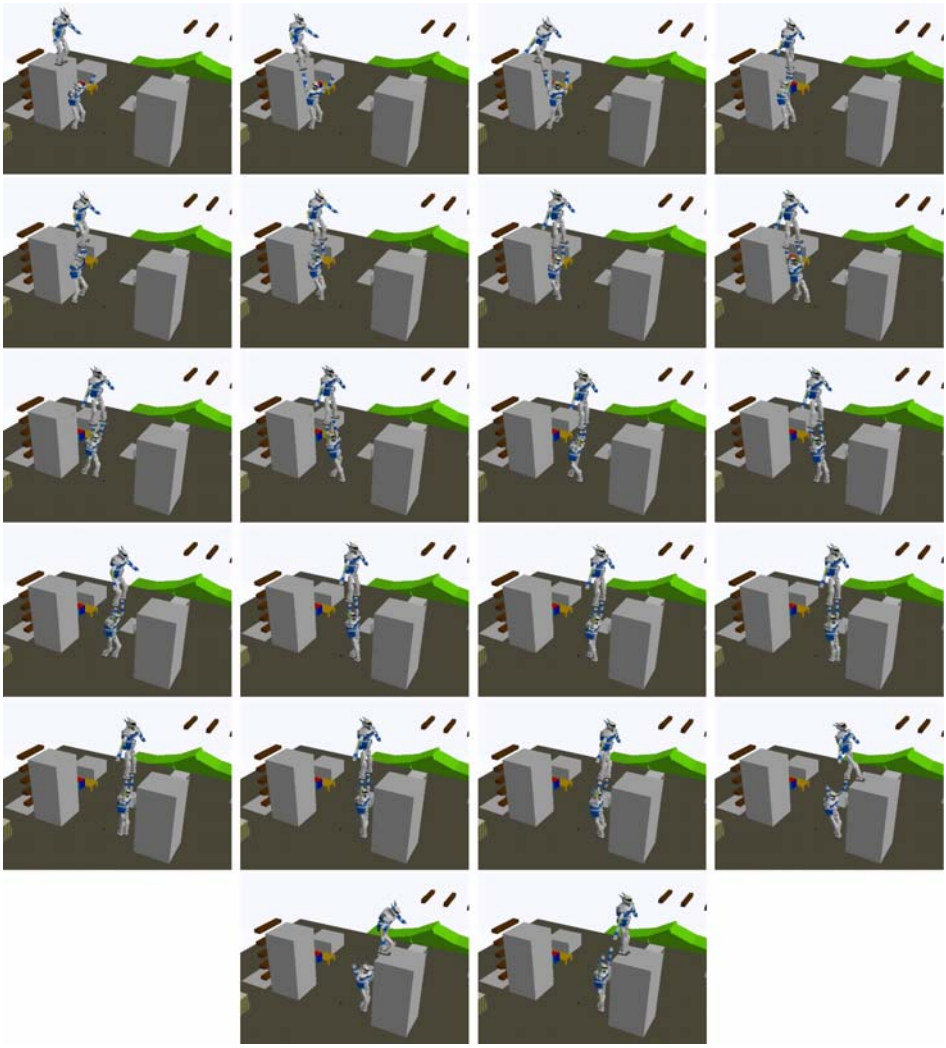


**Figure 9.** Toy scenario. The objective is for the robot to get from the first platform to the second platform. This cannot be achieved without using the other robot as a support. The planner finds such a solution. Contacts are created and broken between the hands of the supporting robot and the feet 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.

The previous contact-before-motion planning works [1,2] relied in their second stage, the continuous motion planning, on geometric path planning techniques [16,18]. The motion generated is thus restricted to be quasi-static, and would benefit from a post-processing dynamic filtering stage [28,29].

We have investigated two alternative approaches to directly generate dynamic motion, presented, respectively, in [30,31].

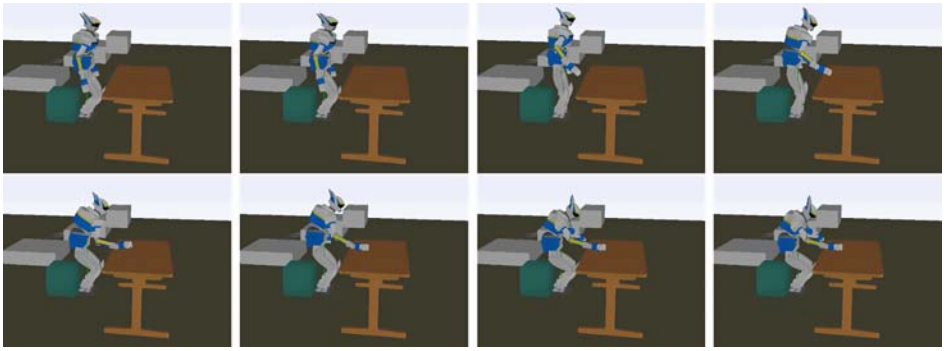


**Figure 10.** Screenshots captured from the sitting dynamic simulation using a feedback-based multi-objective controller.

The first one [30] is based on a closed-loop control law that combines a finite-state machine with a multi-objective controller [32] to generate the motion in simulation. The multi-objective controller solves at every step of the simulation a quadratic program (QP) that minimizes a weighted sum of quadratic objectives for the COM, the links of the robot, or the whole posture, subject to linear constraints expressing the dynamics equation of motion of the robot and the non-sliding contact constraints. Collision-avoidance is treated using heuristics on the objectives for the motion of the contact-adding link, but eventhough it is not strictly expressed as a constraint the generated motion is robust to undesired collision events thanks to its feedback nature. Figure 10 shows captured frames from an example motion generated through this method. We refer the reader to [30] for full details and videos.

The second method [31] is based on an open-loop nonlinear optimization on the full motion. The solver finds a motion law $t \mapsto q(t)$ optimizing a performance index (duration of the motion, energy consumed through actuation) subject to dynamics and contact constraints. Figure 11 shows captured frames of a motion generated using this method. Full videos can be found in [33].

## 8. Conclusion

We presented an algorithm that extended previous contact-before-planning methods to more general systems made of multiple entities. The algorithm was written as a best-first search with an inverse kinematics-and-statics solver as a key component. Many classes of locomotion and manipulation problems of humanoid robots can be solved through the modeling of the system as a centralized multi-agent system. This algorithm is one component along a long chain of stages, starting from the sensing and modeling of the environment, then to the planning of a rough path that takes obstacles into account as possible contact supports, followed by the execution of the present algorithm, before finally feeding its result to either an offline motion generation tool that will be executed on the

**Figure 11.** Screenshots captured from the sitting real world experiment using an open-loop full-motion optimization method.

robot with adequate trajectory tracking system, or is directly used by an online control law. Each of these many stages requires computation times that range from minutes to hours, making the whole framework impractical for now as such for achieving real-time sensing, planning, and control. This is a limitation generally found in state-of-the-art path planning and trajectory optimization in general. However, our contribution in this work that is the generality in terms of possible systems and tasks – one formulation fits all – constitutes one step toward full autonomy for generic systems and tasks centered around the humanoid robot.

*Acknowledgements*

This work was partially supported by the Japan Society for the Promotion of Science (JSPS) Grant-in-Aid for Scientific Research (B), 22300071, 2010, and by the FP7 RoboHow.Cog project www.robohow.eu.

## References

1. A. Escande, A. Kheddar and S.Miossec, Planning support contact-points for humanoid robots and experiments on HRP-2, in: *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Beijing, pp. 2974–2979 (2006).
2. K. Hauser, T. Bretl and J.-C. Latombe, Non-gaited humanoid locomotion planning, in: *Proc. IEEE-RAS Int. Conf. on Humanoid Robots,* Tsukuba, pp. 7–12 (2005).
3. J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba and H. Inoue, Footstep planning among obstacles for biped robots, in: *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Maui, HI, pp. 500–505 (2001).
4. J. Chestnutt, J. Kuffner, K. Nishiwaki and S. Kagami, Planning biped navigation strategies in complex environments, in: *Proc. IEEE-RAS Int. Conf. on Humanoid Robots*, Munich, 2003.
5. J. Chestnutt, M. Lau, J. Kuffner, G. Cheung, J. Hodgins and T. Kanade, Footstep planning for the Honda ASIMO humanoid, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Barcelona, pp. 629–634 (2005).
6. N. Torkos and M. van de Panne, Footprint-based quadruped motion synthesis, in: *Proc. Graphics Interface*, Vancouver, pp. 151–160 (1998).
7. E. Kokkevis, D. Metaxas and N. I. Badler, Autonomous animation and control of four-legged animals, in: *Proc. Graphics Interface, Quebec*, pp. 10–17 (1995).

8. K. Hauser and J.-C. Latombe, Multi-modal motion planning in non-expansive spaces, *Int. J. Robotics Res.* **29**(7), 897–915 (June 2010).
9. A. Escande, A. Kheddar, S. Miossec and S. Garsault, Planning support contact-points for acyclic motions and experiments on HRP-2, in: *Proc. Int. Symp. on Experimental Robotics*, Athens, pp. 293–302 (2008).
10. M. Erdmann and T. Lozano-Perez, On multiple moving objects, *Algorithmica* **2**, 477–521 (1987).
11. J. van den Berg and M. Overmars, Prioritized motion planning for multiple robots, in: *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Edmonton, pp. 430–435 (2005).
12. T. Simeon, S. Leroy and J.-P. Laumond, Path coordination for multiple mobile robots: a resolution complete algorithm, *IEEE Trans. Robotics Automat.* **18**, 42–49 (2002).
13. K. Bouyarmane and A. Kheddar, Multi-contact stances planning for multiple agents, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Shanghai, pp. 5546–5353 (2011).
14. K. Bouyarmane and A. Kheddar, Static multi-contact inverse problem for multiple humanoid robots and manipulated objects, in: *Proc. IEEE-RAS Int. Conf. on Humanoid Robots*, Nashville, TN, pp. 8–13 (2010).
15. K. Bouyarmane and A. Kheddar, Multi-contacts stances planning for humanoid locomotion and manipulation, in: *Proc. 28th Ann. Conf. of the Robotics Society of Japan*, Nagoya (2010).
16. J. C. Latombe, *Robot Motion Planning*. Kluwer Academic (1991).
17. K. Bouyarmane, A. Escande, F. Lamiraux and A. Kheddar, Potential field guide for multicontact humanoid motion planning, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Kobe, pp. 1165–1170 (2009).
18. S.M. LaValle, *Planning Algorithms*. Cambridge University Press, Cambridge, UK (2006).
19. S. M. LaValle, J. Yakey and L. Kavraki, A probabilistic roadmap approach for systems with closed kinematic chains, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Detroit, MI, pp. 1671–1676 (1999).
20. L. Han and N. M. Amato, A kinematics-based probabilistic roadmap method for closed chain systems, in: *Proc. Workshop on Algorithmic Foundations of Robotics*, Hanover, NH, pp. 233–246 (2000).
21. J. Cortes, T. Siméon and J.-P. Laumond, A random loop generator for planning the motions of closed kinematic chains using PRM methods, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Washington, DC, pp. 2141–2146 (2002).
22. T. Bretl and S. Lall, Testing static equilibrium for legged robots, *IEEE Trans. Robotics* **24**(4), 794–807 (2008).
23. M. Benallegue, A. Escande, S. Miossec and A. Kheddar, Fast $C^1$ proximity queries using support mapping of sphere-torus-patches bounding volumes, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Kobe, pp. 483–488 (2009).
24. E. G. Gilbert, D. W. Johnson and S. S. Keerthi, A fast procedure for computing the distance between complex objects in three-dimensional space, *IEEE Trans. Robotics Automat.* **4,** 193–203 (1988).
25. L. E. Kavraki, P. Svestka, J. Claude Latombe and M. H. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, *IEEE Trans. Robotics Automat.* **12**, 566–580 (1996).
26. S. LaValle, J. Kuffner, Rapidly-exploring random trees: progress and prospects, in: B. Donald, K. Lynch, and D. Rus (Eds.), *Algorithmic and Computational Robotics: New Direction*, A. K. Peters, Wellesley, pp. 293–308 (2001).

27. A. Wachter and L. T. Biegler, On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, *Math. Prog.* **106**, 25–57 (2006).
28. K. Yamane and Y. Nakamura, Dynamics filter – concept and implementation of online motion generator for human figures, *IEEE Trans. Robotics* **19**, 421–432 (2003).
29. J. J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba and H. Inoue, Dynamically-stable motion planning for humanoid robots, *Autonom. Robots* **12**, 105–118 (2002).
30. K. Bouyarmane and A. Kheddar, Using a multi-objective controller to synthesize simulated humanoid robot motion with changing contact configuration, in: *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, San Fransisco, CA, pp. 4414–4419 (2011).
31. S. Lengagne, P. Mathieu, A. Kheddar, and E. Yoshida, Generation of dynamic multi-contact motions: 2D case studies, in: *Proc. IEEE-RAS Int. Conf. on Humanoid Robots*, Nashville, TN, pp. 14–20 (2010).
32. Y. Abe, M. da Silva and J. Popovic, Multiobjective control with frictional contacts, in: *Proc. ACM SIGGRAPH/Eurographics Symp. on Computer Animation*, Aire-la-Ville, pp. 249–258 (2007).
33. S. Lengagne and K. Bouyarmane, Robots learn to walk like a senior citizen. http://www.newscientist.com/article/dn19901-robots-learn-to-walk-like-a-senior-citizen.html (December 2010).
34. M. W. Spong, S. Hutchinson and M. Vidyasagar, *Robot Modeling and Control*. Wiley, Hoboken, NJ (2005).
35. H. Bruyninckx and J. D. Schutter, Symbolic differentiation of the velocity mapping for a serial kinematic chain, *Mech. Machine Theory* **31**(2), 135–148 (1996).
36. O. Lefebvre, F. Lamiraux and D. Bonnafous, Fast computation of robot–obstacle interactions in nonholonomic trajectory deformation, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Barcelona, pp. 4612–4617 (2005).
37. K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi and T. Isozumi, Humanoid Robot HRP-2, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Barcelona, pp. 1083–1090 (2004).

## Appendix A. Gradient Derivations

### *A.1. Geometric Gradients*

The problem (26) is a constrained nonlinear optimization problem. To solve it we need to compute at least the gradients of the objective function and constraints. The gradients with respect to $\lambda$ do not require particular attention. The gradients with respect to $q$, however, need careful derivations that we are going to detail. These gradients (as well as the computation of (15)) are all down to the computation of the three following analytical Jacobian matrices

$$\frac{\partial \Omega_{r,l}(q_r)}{\partial q_r}, \tag{27}$$

$$\frac{\partial [R_{r,l}(q_r)u]}{\partial q_r}, \tag{28}$$

$$\frac{\partial [R_{r,l}(q_r)^T u]}{\partial q_r}, \tag{29}$$

where $u$ is any arbitrary vector of $\mathbb{R}^3$. We consider the most general case in which $q_r = (\xi_r, \phi_r, \theta_r)$, the other cases follow straightforwardly.

To compute these Jacobian matrices we make use of the algorithmically-computable linear and angular *Geometric Jacobians* [34] $J^{\text{lin}}_{r,l}$ and $J^{\text{ang}}_{r,l}$ defined as

$$\dot{\xi}^0_{l,r} = J^{\text{lin}}_{r,l}\dot{\theta}_r, \tag{30}$$

$$\omega^0_{r,l} = J^{\text{ang}}_{r,l}\dot{\theta}_r, \tag{31}$$

where $\dot{\xi}^0_{l,r}$ and $\omega^0_{r,l}$ are, respectively, the root-frame-expressed linear and angular velocities of the link $(r, l)$ relative to root link. If $\phi = (\alpha, \beta, \gamma, \delta)$ is a unit quaternion, then let

$$\begin{aligned}\rho(\phi) &= \rho(\alpha, \beta, \gamma, \delta) \\ &= \begin{pmatrix} 2(\alpha^2+\beta^2)-1 & 2(\beta\gamma-\alpha\delta) & 2(\beta\delta+\alpha\gamma) \\ 2(\beta\gamma+\alpha\delta) & 2(\alpha^2+\gamma^2)-1 & 2(\gamma\delta-\alpha\beta) \\ 2(\beta\delta-\alpha\gamma) & 2(\gamma\delta+\alpha\beta) & 2(\alpha^2+\delta^2)-1 \end{pmatrix},\end{aligned} \tag{32}$$

be the corresponding orientation matrix. We can show that

$$\frac{\partial \Omega_{r,l}}{\partial q_r} = \left[\mathbf{1}_{3\times 3}\middle|\left(\frac{\partial \rho}{\partial \alpha}\Omega^0_{r,l}, \frac{\partial \rho}{\partial \beta}\Omega^0_{r,l}, \frac{\partial \rho}{\partial \gamma}\Omega^0_{r,l}, \frac{\partial \rho}{\partial \delta}\Omega^0_{r,l}\right)\middle| R_{r,0}J^{\text{lin}}_{r,l}\right], \tag{33}$$

$$\frac{\partial [R_{r,l}u]}{\partial q_r} = \left[\mathbf{0}_{3\times 3}\middle|\left(\frac{\partial \rho}{\partial \alpha}R^0_{r,l}u, \frac{\partial \rho}{\partial \beta}R^0_{r,l}u, \frac{\partial \rho}{\partial \gamma}R^0_{r,l}u, \frac{\partial \rho}{\partial \delta}R^0_{r,l}u\right)\middle| - R_{r,0}\left[(R^0_{r,l}u)\times\right]J^{\text{ang}}_{r,l}\right], \tag{34}$$

$$\frac{\partial [R^T_{r,l}u]}{\partial q_r} = \left[\mathbf{0}_{3\times 3}\middle|\left(R^{0\,T}_{r,l}\frac{\partial \rho^T}{\partial \alpha}u, R^{0\,T}_{r,l}\frac{\partial \rho^T}{\partial \beta}u, R^{0\,T}_{r,l}\frac{\partial \rho^T}{\partial \gamma}u, R^{0\,T}_{r,l}\frac{\partial \rho^T}{\partial \delta}u\right)\middle| R^{0\,T}_{r,l}[(R^T_{r,0}u)\times]J^{\text{ang}}_{r,l}\right], \tag{35}$$

where the notation $[v\times]$ is used to denote the skew symmetric matrix that corresponds the cross product by the vector $v$ of $\mathbb{R}^3$. The partial derivatives of $\rho$ are expressed at $\phi_r$.

### A.2. Torque Gradients

Let $\mu \in \{1, \ldots, \Theta_r\}$ and $\Upsilon^{\mu}_{r,l}$ denote the $\mu$th column of $\Upsilon_{r,l}$ defined in (15). In order to compute the gradient of the constraint (19), we need to compute the partial derivatives of the mappings

$$\Upsilon_1 : q_{r_{i1}} \mapsto \Upsilon^{\mu}_{r_{i1},l_{i1}}(q_{r_{i1}}, a_{s_{i1},j}), \tag{36}$$

$$\Upsilon_2 : (q_{r_{i1}}, q_{r_{i2}}) \mapsto \Upsilon^{\mu}_{r_{i2},l_{i2}}(q_{r_{i2}}, a'_{s_{i2},j}(q_{r_{i1}}, q_{r_{i2}})). \tag{37}$$

Let $D_1 \Upsilon^{\mu}_{r,l}$ and $D_2 \Upsilon^{\mu}_{r,l}$ denote the partial derivatives of $(q_r, b) \mapsto \Upsilon^{\mu}_{r,l}(q_r, b)$ with respect to $q_r$ and $b$, respectively (recall that $b$ is an arbitrary $\mathbb{R}^3$ variable). We have

$$\frac{\partial \Upsilon_1}{\partial q_{r_{i1}}} = D_1 \Upsilon^{\mu}_{r_{i1},l_{i1}}(q_{r_{i1}}, a_{s_{i1},j}), \tag{38}$$

$$\frac{\partial \Upsilon_2}{\partial q_{r_{i1}}} = D_2 \Upsilon^{\mu}_{r_{i2},l_{i2}}(q_{r_{i2}}, a'_{s_{i2},j}) \frac{\partial a'_{s_{i2},j}(q_{r_{i1}}, q_{r_{i2}})}{\partial q_{r_{i1}}}, \tag{39}$$

$$\frac{\partial \Upsilon_2}{\partial q_{r_{i2}}} = D_1 \Upsilon^{\mu}_{r_{i2},l_{i2}}(q_{r_{i2}}, a'_{s_{i2},j}) + D_2 \Upsilon^{\mu}_{r_{i2},l_{i2}}(q_{r_{i2}}, a'_{s_{i2},j}) \frac{\partial a'_{s_{i2},j}(q_{r_{i1}}, q_{r_{i2}})}{\partial q_{r_{i2}}}. \tag{40}$$

In these expressions, the two derivatives

$$\frac{\partial a'_{s_{i2},j}(q_{r_{i1}}, q_{r_{i2}})}{\partial q_{r_{i1}}}, \frac{\partial a'_{s_{i2},j}(q_{r_{i1}}, q_{r_{i2}})}{\partial q_{r_{i2}}} \tag{41}$$

can be computed directly using the three Jacobian matrices (27)–(29). Indeed from (14) we have, denoting temporarily $v = A_{s_{i1},j}(q_{r_{i1}}) - \Omega_{r_{i2},l_{i2}}(q_{r_{i2}})$,

$$\frac{\partial a'_{s_{i2},j}(q_{r_{i1}}, q_{r_{i2}})}{\partial q_{r_{i1}}} = R_{r_{i2},l_{i2}}(q_{r_{i2}})^T \frac{\partial A_{s_{i1},j}(q_{r_{i1}})}{\partial q_{r_{i1}}}, \tag{42}$$

$$\frac{\partial a'_{s_{i2},j}(q_{r_{i1}}, q_{r_{i2}})}{\partial q_{r_{i2}}} = \frac{\partial [R_{r_{i2},l_{i2}}(q_{r_{i2}})^T v]}{\partial q_{r_{i2}}} - R_{r_{i2},l_{i2}}(q_{r_{i2}})^T \frac{\partial \Omega_{r_{i2},l_{i2}}(q_{r_{i2}})}{\partial q_{r_{i2}}}. \tag{43}$$

and $D_2 \Upsilon^{\mu}_{r,l}$ require a last derivation effort. As for the former, we can write, denoting temporarily $\zeta_\mu$ and $\psi_\mu$ the $\mu$th column of $J^{\text{lin}}_{r,l}$ and $J^{\text{ang}}_{r,l}$, respectively,

$$D_1 \Upsilon^{\mu}_{r,l}(q_r, b) = \left[ \mathbf{0}_{3\times 3} \middle| \left( \frac{\partial \rho}{\partial \alpha}\zeta_\mu(b), \frac{\partial \rho}{\partial \beta}\zeta_\mu(b), \frac{\partial \rho}{\partial \gamma}\zeta_\mu(b), \frac{\partial \rho}{\partial \delta}\zeta_\mu(b) \right) \middle| R_{r,0} \left( \frac{\partial \zeta_\mu(b)}{\partial \theta_{r,v}} \right)_{v\in\{1,\ldots,\Theta_r\}} \right], \tag{44}$$

where

$$\zeta_\mu(b) = \zeta_\mu + \psi_\mu \times [R^0_{r,l} b] \tag{45}$$

is the $\mu$th column of the linear Geometric Jacobian expressed at the point $b$, and, by generalizing the result of [35] to floating-base mechanisms,

$$\frac{\partial \zeta_\mu(b)}{\partial \theta_{r,v}} = \begin{cases} \psi_v \times \zeta_\mu(b) & \text{if } v < \mu, \\ \psi_\mu \times \zeta_\mu(b) & \text{if } v = \mu, \\ \psi_\mu \times \zeta_v(b) & \text{if } v > \mu. \end{cases} \tag{46}$$

Finally, as for $D_2 \Upsilon^{\mu}_{r,l}$, we have

$$D_2 \Upsilon^{\mu}_{r,l} = R_{r,0}[(\psi_\mu)\times] R^0_{r,l}. \tag{47}$$

### *A.3. Distance Gradients*

Let us compute the gradient of the constraint (20). Let $P_1$ and $P_2$ be, respectively, the closest points on $\mathcal{B}_{\text{nc1}}$ and $\mathcal{B}_{\text{nc2}}$, such that $\mathscr{D}(\mathcal{B}_{\text{nc1}}, \mathcal{B}_{\text{nc2}}) = ||P_2 - P_1||$ if there is no collision and the farthest points such that $\mathscr{D}(\mathcal{B}_{\text{nc1}}, \mathcal{B}_{\text{nc2}}) = -||P_2 - P_1||$ 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 [36] makes the computation of this gradient straightforward by considering the Jacobians at the points $P_{1_{\in \mathcal{B}_{\text{nc1}}}}$ and $P_{2_{\in \mathcal{B}_{\text{nc2}}}}$ that are rigidly attached to $\mathcal{B}_{\text{nc1}}$ and $\mathcal{B}_{\text{nc2}}$, respectively, and coincide with $P_1$ and $P_2$ in the configuration $q$ at which we are computing the gradient. So we can write

$$\frac{\partial \mathscr{D}(\mathcal{B}_{\text{nc1}}, \mathcal{B}_{\text{nc2}})}{\partial q} = \begin{cases} \frac{(P_1 - P_2)}{||P_1 - P_2||}^T \left( \frac{\partial P_{1_{\in \mathcal{B}_{\text{nc1}}}}}{\partial q_{r_1}} - \frac{\partial P_{2_{\in \mathcal{B}_{\text{nc2}}}}}{\partial q_{r_2}} \right) & \text{if there is no collision,} \\ -\frac{(P_1 - P_2)}{||P_1 - P_2||}^T \left( \frac{\partial P_{1_{\in \mathcal{B}_{\text{nc1}}}}}{\partial q_{r_1}} - \frac{\partial P_{2_{\in \mathcal{B}_{\text{nc2}}}}}{\partial q_{r_2}} \right) & \text{if there is interpenetration.} \end{cases} \tag{48}$$

## Appendix B. Complexity Analysis

Let us denote $B$ the branching factor of the search tree $\mathcal{T}$ constructed in Algorithm 1. $B$ is dominated by the total number of possible surface pairs that can constitute a contact. To count the total number of such possible surface pairs, we first recall that a surface $(r', l', s')$ can be in contact with any other surface not belonging to the same link $l'$. Thus the number of surfaces with which $(r', l', s')$ can be in contact is

$$\sum_{r\neq r'}\sum_{l} S_{r,l} + \sum_{l\neq l'} S_{r',l}, \tag{49}$$

and the total number of possible surface pairs that can be in contact is thus

$$\sum_{r',l'}\left[S_{r',l'}\left(\sum_{r\neq r'}\sum_{l} S_{r,l} + \sum_{l\neq l'} S_{r',l}\right)\right] \leq \left[\sum_{r,l} S_{r,l}\right]^2. \tag{50}$$

Let $n_S = \sum_{r,l} S_{r,l}$ denote the total number of surfaces, we can thus write

$$B = \mathcal{O}(n_S^2). \tag{51}$$

Let now $h$ denote the number of steps in the final solution. $h$ is the height of the tree $\mathcal{T}$. The worst case complexity of Algorithm 1 is when the tree is full. If $C_w$ denotes the worst case complexity, then we have

$$C_w = \mathcal{O}(B^h). \tag{52}$$

We now need to estimate $h$. The notation $||\sigma_{\text{init}} - \sigma_{\text{goal}}||$ will here refer to an Euclidean distance between two arbitrary points belonging, respectively, to the initial and goal stance print, and $d$ a characteristic dimension of the system of robots (for instance the length of the leg of a humanoid robot). We can write a domination relation for $h$ as

$$h = \mathcal{O}\left(\frac{||\sigma_{\text{init}} - \sigma_{\text{goal}}||}{d}\right). \tag{53}$$

Finally from (49), (51), and (52), we can express the worst case complexity in terms of the input data of the problem

$$C_w = \mathcal{O}\left(n_S^{2\left[||\sigma_{\text{init}} - \sigma_{\text{goal}}||/d\right]}\right). \tag{54}$$

In practice, the search tree is rarely full. More than that, it is often very sparse. Therefore the complexity that expresses more accurately the average behavior of Algorithm 4 would be the best case complexity, that we denote $C_b$.

In this case we have a linear relation

$$C_b = \mathcal{O}(hB), \tag{55}$$

and therefore

$$C_b = \mathcal{O}\left(\frac{||\sigma_{\text{init}} - \sigma_{\text{goal}}||}{d} n_S^2\right). \tag{56}$$

## Appendix C. Notation Tables

**Table 1.** Systems

| | |
|---|---|
| $N$ | number of entities (robots and objects) constituting the system |
| $r \in \{0, \ldots, N\}$ | index of the entity, or the environment (case $r = 0$) |
| $q_r \in \mathcal{C}_r$ | configuration and configuration space of the entity $r$ |
| $q \in \mathcal{C}$ | configuration and configuration space of the whole system |
| $L_r$ | number of rigid links in the entity $r$ |
| $l \in \{0, \ldots, L_r - 1\}$ | index of the link in the entity $r$ ($l = 0$ is the root link) |
| $\Theta_r$ | number of actuated joints of the entity $r$ (robot) |
| $\theta_r \in \mathbb{R}^{\Theta_r}$ | joint angle vector of the entity $r$ (robot) |
| $\xi_r$ | position of the root link of the entity $r$ |
| $\phi_r$ | orientation (unit quaternion) of the root link of the entity $r$ |
| $\Omega_{r,l}$ (resp. $\Omega^0_{r,l}$) | global position of link $l$ of the entity $r$ expressed in the global (resp. root link) frame |
| $R_{r,l}$ (resp. $R^0_{r,l}$) | global orientation matrix of link $l$ of the entity $r$ expressed in the global (resp. root link) frame |

**Table 2.** Contact surfaces and contacts

| | |
|---|---|
| $S_{r,l}$ | number of contact surfaces on link $l$ of entity $r$ |
| $s \in \{1, \ldots, S_{r,l}\}$ | index of the surface in link $l$ of entity $r$ |
| $(O_s, \vec{x}_s, \vec{y}_s, \vec{z}_s)$ | frame corresponding to the surface $s$ |
| $V_s$ | number of vertices of the polygonal shape of the surface $s$ |
| $a_{s,\cdot}$ (resp. $A_{s,\cdot}$) | vertex of the surface $s$ expressed in the link's local (resp. the global) frame |
| $c$ | contact between surfaces $s_1$ and $s_2$ |
| $(x_c, y_c, \vartheta_c)$ | relative position and orientation of the two surfaces in contact |

**Table 3.** Stances and stance set

| | |
|---|---|
| $E_{\text{ctc}}$ | set of all contacts |
| $\sigma$ | stance (set of contacts, subset of $E_{\text{ctc}}$) |
| $n_\sigma$ | number of contacts in stance $\sigma$ |
| $\mathcal{Q}_\sigma$ | submainfold consisting of the configurations that geometrically realize $\sigma$ |
| $\mathcal{F}_\sigma$ | subset of $\mathcal{Q}_\sigma$ made of physically feasible configurations |
| $\Sigma$ | set of all stances (subset of $2^{E_{\text{ctc}}}$) |
| $p$ | direct kinematics mapping from $\mathcal{C}$ to $\Sigma$ |
| $\text{Adj}^+(\sigma)$ | set of stances adjacent to $\sigma$ by adding one contact |
| $\text{Adj}^-(\sigma)$ | set of stances adjacent to $\sigma$ by removing one contact |
| $\text{Adj}(\sigma)$ | set of stance adjacent to sigma (union of $\text{Adj}^+(\sigma)$ and $\text{Adj}^-(\sigma)$) |
| $\sim_\sigma$ | equivalence relation on $\text{Adj}^+(\sigma)$ erasing the specification of the position of the added contact |
| $[\sigma']_{\sim_\sigma}$ | equivalence class of which $\sigma'$is a representative |

**Table 4.** Statics and IK solver parameters

| | |
|---|---|
| $c_i$ | $i$th contact of the stance $\sigma$ |
| $s_{i1}$ (resp. $s_{i2}$) | smaller area (resp. larger area) surface of the surfaces constituting $c_i$ |
| $f_{i,j}$ | contact force applied by $l_{i2}$ on $l_{i1}$ at $a_{s_{i1},j}$ |
| $a'_{s_{i2},}$ | position of $a_{s_{i1},}$ expressed in the local frame of $s_{i2}$'s link |
| $m_r$ | mass of the entity $r$ |
| $C_r$ | position of the center of mass of the entity $r$ |
| $g$ | gravity vector |
| $\tau_r \in \mathbb{R}^{\Theta_r}$ | actuation torques of entity $r$ |
| $I$ | the index set $\{1, \ldots, n_\sigma\}$ |
| $I_1(r)$ | set of indices of the contacts to which $r$ contributes with a smaller area surface |
| $I_2(r)$ | set of indices of the contacts to which $r$ contributes with a larger area surface |
| $I_3(r)$ | set of indices of the contacts not involving any surface from $r$ |
| $K$ | number of generators of the linearized friction cones |
| $U_{i,k}$ | $k$th generator of the linearized friction cone at $a_{s_{i1}}$ |
| $\lambda_{i,j,k}$ | $U_{i,k}$'s coordinate of $f_{i,j}$ |
| obj | objective function to optimize by the inverse kinematics-and-statics solver |
| $\text{obj}_i$ | $i$th component of obj |
| $w_i$ | weight of objective $\text{obj}_i$ |

**Table 5.** Search algorithm

| | |
|---|---|
| $\mathcal{T}$ | search tree |
| $\mathcal{P}$ | priority queue |
| $B$ | branching factor of $\mathcal{T}$ |
| $h$ | number of steps (height of $\mathcal{T}$) |
| $d$ | characteristic dimension of the system (e.g. length of leg of a humanoid) |
| $n_S$ | total number of contact surfaces of the system |
| $C_w$ | worst-case complexity |
| $C_b$ | best-case complexity |

**Table 6.** Gradient computations

| | |
|---|---|
| $\rho$ | mapping from unit quaternions to rotation matrices |
| $(\alpha, \beta, \gamma, \delta)$ | four components of a quaternion |
| $\Upsilon$ | modified Jacobian matrix |
| $\Upsilon_1$, $\Upsilon_2$ | mapping from the configurations to $\Upsilon$ |
| $D_1$, $D_2$ | partial derivative operators |
| $\mathcal{B}_{\text{nc1}}$, $\mathcal{B}_{\text{nc2}}$ | bounding volumes of two links that necessitate a collision-avoidance constraint |
| $\mathscr{D}(\mathcal{B}_{\text{nc1}}, \mathcal{B}_{\text{nc2}})$ | signed distance between the two bounding volumes |
| $P_1$, $P_2$ | witness point |
| $J^{\text{lin}}_{r,l}$ | linear Geometric Jacobian at the origin of link $l$ of entity $r$ |
| $J^{\text{ang}}_{r,l}$ | angular Geometric Jacobian of link $l$ of entity $r$ |
| $\xi^0_{r,l}$ | linear velocity of link $l$ of entity $r$ relative to the entity's root link |
| $\omega^0_{r,l}$ | angular velocity of link $l$ of entity $r$ relative to the entity's root link |
| $\zeta_\mu$ | $\mu$th column of $J^{\text{lin}}_{r,l}$ |
| $\psi_\mu$ | $\mu$th column of $J^{\text{ang}}_{r,l}$ |

## About the Authors

**Karim Bouyarmane** graduated as Ingénieur (equi. MSc) from both Ecole Polytechnique (at Palaiseau, France) and Ecole nationale supérieure des Mines de Paris in 2007 and 2008, respectively. He received the PhD diploma from the Montpellier 2 University in 2011 after spending his three-year PhD research program at AIST in Tsukuba, Japan. He is currently holding a JSPS postdoctoral fellowship position at ATR Computational Neuroscience Laboratories in Kyoto, Japan.

**Abderrahmane Kheddar** received the ingénieur degree from the Institut National d'Informatique (INI), Algiers, the DEA (Master by research), and PhD degree in robotics, both from the University of Paris 6, France. He is presently Directeur de Recherche at CNRS and the director of the CNRS-AIST Joint Robotic Laboratory, UMI3218/CRT, Tsukuba, Japan. He is also leading a new team called "Interactive Digital Humans" IDH that he created at CNRS-UM2 LIRMM at Montpellier, France. His research interests include haptics and humanoids. He is a founding member and a senior advisor of the IEEE/RAS chapter on haptics and is with the editorial board of the IEEE Transactions on Robotics and the Journal of Intelligent and Robotic Systems; he was a founding member of the IEEE Transactions on Haptics and served in its editorial board during three years (2007–2010). He cochaired the 2001 IEEE RO-MAN workshop, chaired the 2006 EuroHaptics conference for its first edition in France, and organized several conference workshops (IROS 2004, RO-MAN 2009, RO-MAN 2010, RSS 2010). He is a member of the IEEE. He has more than 160 scientific publications as book chapters, journals, and major international conferences. The New Scientist and the Reuters Press Agency advertized his recent work in multi-contact planning.