2014 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS 2014)
September 14-18, 2014, Chicago, IL, USA

Task-Oriented Whole-Body Planning for Humanoids based on Hybrid Motion Generation

Marco Cognetti, Pouya Mohammadi, Giuseppe Oriolo, Marilena Vendittelli

Abstract— **This paper considers the problem of planning the motion of a humanoid robot that must execute a manipulation task, possibly requiring stepping, in environments cluttered by obstacles. The proposed method explores the submanifold of the configuration space that is admissible with respect to the assigned task and at the same time satisfies other constraints, including humanoid equilibrium. The exploration tree is expanded using a hybrid scheme that simultaneously generates footsteps and whole-body motions. The algorithm has been implemented for the humanoid robot NAO and validated through planning experiments and dynamic playback in V-REP.**

I. Introduction

Humanoids have been the subject of intensive research activity in the last two decades. The final objective is to develop highly versatile robotic platforms that can effectively assist — or even replace — humans in their daily activities.

In the past, many researchers have focused on the design of control models and algorithms for achieving stable biped locomotion. Due to their highly redundant kinematic structure, however, humanoids are capable to achieve other complex tasks in addition to locomotion, the most relevant being manipulation. To fully express their versatility, they must be able to plan and perform whole-body motions in unstructured environments that are populated by obstacles.

Simultaneous execution of multiple tasks can be tackled at the kinematic control level recurring to the task-priority framework [1], in which it is also possible to encode the collision-free requirement [2]. A clever application of this technique for humanoid footstep generation in manipulation tasks is presented in [3]. However, kinematic control is inherently local and only suited for generating reactive behaviors. In fact, it has no backtracking capabilities and may fail to find a solution in complex cases. If the geometry of the scene is known, a more deliberative approach consists in planning in advance a collision-free motion realizing the assigned task.

Motion planning for humanoids is particularly challenging for a number of reasons. The first is the high number of degrees of freedom. The second is that a humanoid robot is not a free-flying system in its configuration space — motion must be generated appropriately. Finally, the requirement that the robot maintains equilibrium, either static or dynamic, typically constrains the trajectory of the robot center of mass, thus reducing the dimension of the planning space. As a consequence, even the basic instance of the motion planning problem (i.e., find a collision-free motion between two rest configurations) is usually approached by introducing simplifications at various levels of the problem formulation.

For example, one may simplify the environment by taking into account collisions only at the footstep [4], [5], [6] or at the leg [7] level. A somewhat dual approach consists in finding first a collision-free path for a simplified geometric model of the humanoid, such as its bounding volume, and then approximating this path with a feasible locomotion trajectory. This technique, used in [8] to animate digital characters and in [9] to plan dynamically feasible motions of a humanoid, requires reshaping the path if the feasible trajectory is found to be in collision with obstacles. The whole configuration space of a humanoid is considered in [10] to plan first collision-free, statically stable motions that are then converted to dynamically stable collision-free trajectories; in this work, however, the robot does not perform stepping motions. The method in [11] represents one of the few motion planning methods that simultaneously generates footsteps and whole-body motions.

In this paper, we are interested in task-constrained motion planning, i.e., finding collision-free motions for a humanoid that is assigned a certain task (e.g., a manipulation action) whose execution may require stepping. Related literature shows three main approaches to this problem: (i) separate locomotion from task execution [12], [13]; (ii) integrate them with a two-phase planner which first computes a collision-free, statically stable paths for a free-flying humanoid base, and then approximates it with a dynamically stable walking motion [14]; (iii) achieve acyclic locomotion and task execution through whole-body contact planning [15].

As [14] and [15], our approach does not separate locomotion from task execution. In particular, using the task-constrained motion planning framework developed in [16], the proposed method explores the submanifold of the configuration space that is admissible with respect to tasks and possible other constraints, including humanoid equilibrium. Expansion of the search tree within the constrained manifold is obtained through a hybrid scheme that generates simultaneously feet positions and whole-body motions, which are validated by collision checks.

The fact that in the proposed planner steps and whole-body motions are simultaneously generated using a hybrid motion model has some relevant consequences: for example, stepping over obstacles becomes possible, while it is not contemplated in [14]. Also, both statically and dynamically balanced walking can be generated, while in [14] collisions

The authors are with the Dipartimento di Ingegneria Informatica, Automatica e Gestionale, Sapienza Università di Roma, via Ariosto 25, 00185 Roma, Italy. E-mail: {cognetti, oriolo, vendittelli}@diag.uniroma1.it, pouya.mohammadi@gmail.com. This work is supported by the EU FP7 ICT-287513 SAPHARI project.

found in the second step can only be avoided by resorting to a dynamic walk. As for [15], that approach does not allow to consider task paths nor does it easily generalize to allow dynamic walking. Finally, both in [14] and in [15], task execution is not exact along arcs joining configurations in the search tree.

Summarizing, the contribution of the proposed work is twofold: first, feasible motions are generated in a single phase; second, walking emerges naturally from the solution of the planning problem as driven by the assigned task. See the concluding section for some additional comments.

The paper is organized as follows. In the next section, we introduce a motion model for the humanoid and provide a rigorous formulation of the addressed problem. The hybrid motion generation mechanism that is at the core of our method is presented in Section III, while a randomized planner that uses such mechanism is described in Section IV. Planning experiments performed in V-REP for the NAO humanoid robot are reported in Section V. Some comments on future work and possible extensions conclude the paper.

II. Problem Formulation

The goal of this section is to provide a clear formulation of our planning problem. To this end, we shall preliminarily introduce a convenient motion model for a humanoid robot.

A. Humanoid Motion Model

Planning for a humanoid requires an appropriate definition of the configuration space. Denote by n the number of articulation variables (joint angles) of the humanoid. In general, to specify the configuration of a free-flying humanoid one should assign the pose (position and orientation) of one body, such as the torso, and the n values of the joint angles. Since in this paper we are interested in planning motions in which the robot has always at least one foot on the ground (no jumping), a configuration may be simply defined as follows

$$\boldsymbol{q} = \begin{pmatrix} \boldsymbol{q}_{\mathrm{spt}} \\ \boldsymbol{q}_{\mathrm{jnt}} \end{pmatrix},$$

where $\boldsymbol{q}_{\mathrm{spt}} = (x_{\mathrm{spt}} \quad y_{\mathrm{spt}} \quad \theta_{\mathrm{spt}})^T \in SE(2)$ is the planar pose of the support foot in a world frame and $\boldsymbol{q}_{\mathrm{jnt}} \in \mathcal{C}_{\mathrm{jnt}}$ is the n-vector of joint angles, with the associated joint space $\mathcal{C}_{\mathrm{jnt}}$. The humanoid configuration space $SE(2) \times \mathcal{C}_{\mathrm{jnt}}$ has therefore dimension $n+3$.

The above definition of configuration requires that the support foot is always identified, even when the humanoid is in a double support phase (both feet on the ground). Our planner shall then arbitrarily define the support foot at the starting configuration, which is typically in double support. Whenever a step is planned, the swing foot moves to a different location on the ground, which is defined as the new pose of the support foot, as shown in Figure 1. The identity of the support foot therefore changes at every step.

The partitioning of the configuration vector $\boldsymbol{q}$ reflects the different way in which the coordinates evolve. In particular, the support foot pose $\boldsymbol{q}_{\mathrm{spt}}$ undergoes discrete displacements achieved through steps, whereas the joint coordinates $\boldsymbol{q}_{\mathrm{jnt}}$ can be changed with continuity. This specific mechanism is well represented by the following *hybrid* motion model

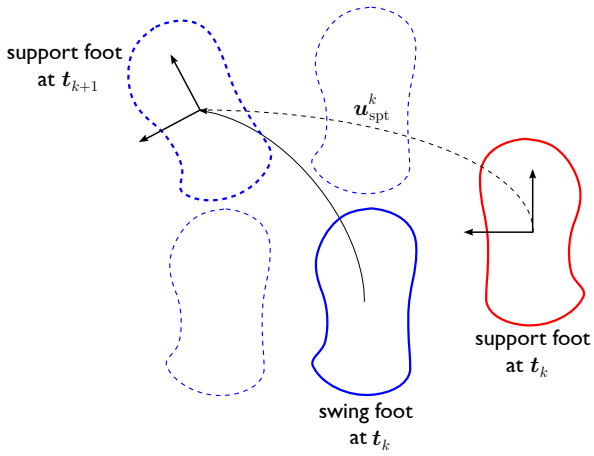

Fig. 1. Support foot displacement after a step initiated at t_k. The swing foot moves from its initial pose to the selected final pose (thick dashed blue), and becomes the new support foot. The support foot displacement $\boldsymbol{u}^k_{\mathrm{spt}}$ is defined accordingly. Other possible final poses for the swing foot are also shown (light dashed blue).

$$\boldsymbol{q}^{k+1}_{\mathrm{spt}} = \boldsymbol{q}^k_{\mathrm{spt}} + \boldsymbol{A}(\boldsymbol{q}^k_{\mathrm{spt}})\boldsymbol{u}^k_{\mathrm{spt}} \tag{1}$$

$$\dot{\boldsymbol{q}}_{\mathrm{jnt}}(t) = \boldsymbol{v}_{\mathrm{jnt}}(t) \qquad t \in [t_k, t_{k+1}]. \tag{2}$$

This model describes the evolution of the robot configuration within a generic time interval $[t_k, t_{k+1}]$ in which the robot performs a *stepping motion*, i.e., a whole-body motion that produces also a displacement of the support foot (see Figure 1). In particular:

- $\boldsymbol{q}^k_{\mathrm{spt}} = \boldsymbol{q}_{\mathrm{spt}}(t_k)$ and $\boldsymbol{q}^{k+1}_{\mathrm{spt}} = \boldsymbol{q}_{\mathrm{spt}}(t_{k+1})$ are the poses of the support foot respectively at the start and the end of the time interval;
- $\boldsymbol{A}(\boldsymbol{q}^k_{\mathrm{spt}})$ is the homogeneous transformation matrix from the support foot frame at t_k to the world frame;
- $\boldsymbol{u}^k_{\mathrm{spt}}$ is the pose displacement of the support foot expressed in its frame;
- $\boldsymbol{v}_{\mathrm{jnt}}$ is the velocity command for the humanoid joints.

The discrete-time nature of eq. (1) comes from the fact that a step requires a nonzero time $T_k = t_{k+1} - t_k$ to be completed. Joint variables can change instantaneously, as implied by the continuous-time integrator dynamics (2).

It should be emphasized that in (1–2) the support foot displacement $\boldsymbol{u}^k_{\mathrm{spt}}$ and the joint velocity profile $\boldsymbol{v}_{\mathrm{jnt}}(t)$ in $[t_k, t_{k+1}]$ are not independent. In fact, all humanoid motions are generated at the joint level. In particular, any pose displacement of the support foot from t_k to t_{k+1} will be the result of the motion of the swing leg during the time interval. This will be appropriately handled by our motion generation scheme (see Section III).

The above model is obviously valid also for *non-stepping motions*, i.e., motions in which the robot changes its internal posture without moving its feet. In that case, $\boldsymbol{u}^k_{\mathrm{spt}} = \boldsymbol{0}$ and the support foot does not move; the duration T_k of the motion may then be arbitrary.

In the following, we will use model (1–2) to describe any *elementary* (i.e., stepping or non-stepping) humanoid motion.

B. Task-Oriented Planning

Having defined a motion model for the humanoid, we can turn our attention to the assigned task.

In this paper, we will consider a manipulation task, simply defined as a trajectory (position and possibly orientation) for one of the hands of the humanoid. Tasks that may be described in this way include, e.g, opening a door, turning a valve or picking up an object. See the concluding section for comments on possible generalizations of this formulation.

Collect the task variables in vector $\boldsymbol{y}$, which takes values in an appropriate space. The task coordinates are related to configuration coordinates by a forward kinematic map

$$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{q}_{\rm spt}, \boldsymbol{q}_{\rm jnt}).$$

Suppose that a desired task trajectory $\boldsymbol{y}^*(t)$, $t \in [t_i, t_f]$, is assigned. It is assumed that the initial configuration $\boldsymbol{q}(t_i)$ of the robot is given, and that $\boldsymbol{f}(\boldsymbol{q}_{\rm spt}(t_i), \boldsymbol{q}_{\rm jnt}(t_i)) = \boldsymbol{y}^*(t_i)$.

Our planning problem consists in finding a *feasible* (in a sense to be explained below) humanoid motion over $[t_i, t_f]$ that realizes the assigned task while avoiding collisions with workspace obstacles, whose geometry is known in advance. In general, a solution will consist of a sequence of elementary motions, either stepping or non-stepping, fully described by the hybrid model (1–2) and starting at $t_1 = t_i$.

Generation of an appropriate sequence among the many possible is left to the planner; in general, this also means choosing the duration T_k of each elementary motion. For simplicity, and to keep notation light, we consider all durations to be equal, i.e., $T_k = T$ for all k and $t_f - t_i = N \cdot T$. This assumption is not necessary and can be removed to allow for elementary motions of different duration.

As already mentioned, once a motion is determined at the joint level, the resulting sequence of steps (and in particular, the placements $\boldsymbol{q}^k_{\rm spt}$ of the support foot, for $k = 1, 2, \ldots$) is completely determined. Therefore, a solution consists of a trajectory $\boldsymbol{q}_{\rm jnt}(t)$, $t \in [t_i, t_f]$, such that:

1) The assigned task trajectory is realized; that is,
$$\boldsymbol{y}(t) = \boldsymbol{f}(\boldsymbol{q}^k_{\rm spt}, \boldsymbol{q}_{\rm jnt}(t)) = \boldsymbol{y}^*(t),$$
for all $t \in [t_k, t_{k+1}]$, $k \in [1, \ldots, N]$.
2) Collisions with workspace obstacles and self-collisions are avoided.
3) Position and velocity limits on the robot joints, respectively in the form $\boldsymbol{q}_{\rm jnt,m} < \boldsymbol{q}_{\rm jnt} < \boldsymbol{q}_{\rm jnt,M}$ and $\boldsymbol{v}_{\rm jnt,m} < \boldsymbol{v}_{\rm jnt} < \boldsymbol{v}_{\rm jnt,M}$, are verified.
4) The robot is in equilibrium at all times.

Requirements 3 and 4 express what we mean by *feasibility* of the humanoid motion. In particular, the last may be declined differently, depending on the desired kind of equilibrium. We will come back to this in the next section.

III. Motion Generation

Our planner, which works in an iterative fashion, makes use of a *motion generator*. At each iteration, the motion generator is invoked to produce a feasible, collision-free elementary motion that realizes a portion of the assigned task trajectory.

Due to the hybrid nature of the humanoid model, motion for the two parts of the configuration $\boldsymbol{q}$ is generated using two interleaved procedures. The first (*step generation*) decides if and where to displace the support foot, and produces an associated trajectory for both the swing foot and the center of mass of the humanoid. The second (*joint motion generation*) computes joint velocity commands so as to realize these associated trajectories and, at the same time, comply with the assigned manipulation task within the considered interval.

A. Step Generation

Step generation is invoked with reference to a humanoid configuration $\boldsymbol{q}^k = (\boldsymbol{q}^k_{\rm spt}\ \ \boldsymbol{q}^k_{\rm int})^T$ at time t_k.

First, a displacement $\boldsymbol{u}^k_{\rm spt}$ is chosen for the support foot from the following set of *step primitives*

$$\boldsymbol{0} \cup \left\{ \begin{pmatrix} \pm\alpha\,\delta_x \\ d_{\min} + \beta\,\delta_y \\ \pm\gamma\,\delta_\theta \end{pmatrix}, \alpha, \beta, \gamma \in \{0, 1, \ldots, M\} \right\}$$

where $\boldsymbol{0}$ is the null displacement corresponding to a non-stepping motion. In any other case, $\boldsymbol{u}^k_{\rm spt}$ is the linear combination of three basic displacements: a forward displacement of length δ_x, a lateral displacement of length δ_y (to which one should add $d_{\min}$, the given minimum lateral distance between the feet), and a rotation by an angle δ_θ. The value of M determines the size of the maximum displacement.

To explore the space of possible solutions, the choice of $\boldsymbol{u}^k_{\rm spt}$ in the set of primitives may be performed either randomly or based on an appropriate heuristic. For example, minimizing the angle between the final orientation of the support foot and the Cartesian tangent to the assigned task trajectory at t_k would lead to privileging foot placements that are locally aligned with the manipulation task.

Once $\boldsymbol{u}^k_{\rm spt}$ has been chosen, the new pose of the support foot may be computed using (1). At this point, if $\boldsymbol{u}^k_{\rm spt} \neq \boldsymbol{0}$ a *Stepping Pattern Generator*[1] (SPG for short) is invoked. This is an external module that takes as input the current configuration $\boldsymbol{q}^k$ and the chosen displacement $\boldsymbol{u}^k_{\rm spt}$ of the support foot, and produces as output:

1) a reference trajectory $\boldsymbol{z}^*_{\rm swg}$ in $[t_k, t_{k+1}]$ for the swing foot (position and orientation);
2) a reference trajectory $\boldsymbol{z}^*_{\rm com}$ in $[t_k, t_{k+1}]$ for the center of mass of the humanoid.

The SPG will guarantee that the robot always maintains equilibrium by appropriately assigning $\boldsymbol{z}^*_{\rm com}$. For illustration, static equilibrium is considered in this paper; i.e., we use an SPG that assigns the trajectory of the center of mass in such a way that its ground projection is always contained in the support polygon. Dynamic walking can be achieved using an SPG that relies on the Zero Moment Point concept [17].

If $\boldsymbol{u}^k_{\rm spt} = \boldsymbol{0}$ (a non-stepping motion has been chosen) the SPG is not invoked. The reference trajectory for the swing foot in $[t_k, t_{k+1}]$ is simply $\boldsymbol{z}^*_{\rm swg}(t) \equiv \boldsymbol{z}_{\rm swg}(t_k)$, whereas the equilibrium constraint will be directly taken into account during joint motion generation.

[1] We do not discuss in detail the structure of this module, which is usually available as part of the basic software suite of humanoid robots.

B. Joint Motion Generation

Joint motion generation starts from $\boldsymbol{q}^k = (\boldsymbol{q}^k_{\rm spt}\ \boldsymbol{q}^k_{\rm jnt})^T$ at t_k and realizes the portion of the assigned task trajectory $\boldsymbol{y}^*(t)$ between t_k and t_{k+1}, plus the reference trajectory $\boldsymbol{z}^*_{\rm swg}$ for the swing foot and (if $\boldsymbol{u}^k_{\rm spt} \neq \boldsymbol{0}$) the reference trajectory $\boldsymbol{z}^*_{\rm com}$ for the center of mass in the same interval.

Assume $\boldsymbol{u}^k_{\rm spt} \neq \boldsymbol{0}$. For convenience, define the augmented task vector $\boldsymbol{y}_a = (\boldsymbol{y}^T\, \boldsymbol{z}^T_{\rm swg}\, \boldsymbol{z}^T_{\rm com})^T$ and let $\boldsymbol{J}_a$ be the Jacobian matrix of $\boldsymbol{y}_a$ with respect to $\boldsymbol{q}_{\rm jnt}$. Note that $\boldsymbol{q}^k_{\rm spt}$, which is constant throughout the interval $[t_k, t_{k+1})$, determines the placement of the base of the kinematic chain. Define the augmented task error as $\boldsymbol{e} = \boldsymbol{y}^*_a - \boldsymbol{y}_a$, where $\boldsymbol{y}^*_a(t)$ is the reference value of the augmented task between t_k and t_{k+1}.

Joint motion is generated as

$$\boldsymbol{v}_{\rm jnt} = \boldsymbol{J}^\dagger_a(\boldsymbol{q}_{\rm jnt})\,(\dot{\boldsymbol{y}}^*_a + \boldsymbol{K}\boldsymbol{e}) + (\boldsymbol{I} - \boldsymbol{J}^\dagger_a(\boldsymbol{q}_{\rm jnt})\boldsymbol{J}_a(\boldsymbol{q}_{\rm jnt}))\boldsymbol{w}, \quad (3)$$

where $\boldsymbol{J}^\dagger_a$ is the pseudoinverse of $\boldsymbol{J}_a$, $\boldsymbol{K}$ is a positive definite gain matrix and $\boldsymbol{w}$ is an n-vector that may be arbitrarily specified without affecting execution of the augmented task. In fact, since $\boldsymbol{I} - \boldsymbol{J}^\dagger_a\boldsymbol{J}_a$ is a projection matrix in the null space of $\boldsymbol{J}_a$, substituting (3) in (2) yields $\dot{\boldsymbol{e}} = -\boldsymbol{K}\,\boldsymbol{e}$, i.e., the augmented task reference trajectory is exponentially stable.

To explore the space of possible solutions, we set for the whole $[t_k, t_{k+1}]$ interval

$$\boldsymbol{w} = \boldsymbol{w}_{\rm rnd}, \quad (4)$$

where $\boldsymbol{w}_{\rm rnd}$ is a bounded-norm randomly generated n-vector.

Suppose now that $\boldsymbol{u}^k_{\rm spt} = \boldsymbol{0}$, i.e., a non-stepping motion has been chosen. In this case, the augmented task vector $\boldsymbol{y}_a$ does not contain $\boldsymbol{z}_{\rm com}$. Equation (3) is still used, with the appropriate definition of $\boldsymbol{y}_a$, but vector $\boldsymbol{w}$ is now chosen as

$$\boldsymbol{w} = -\eta \cdot \nabla_{\boldsymbol{q}_{\rm jnt}} H(\boldsymbol{q}_{\rm jnt}) + \boldsymbol{w}_{\rm rnd}, \quad \eta > 0, \quad (5)$$

where $H(\boldsymbol{q}_{\rm jnt})$ is the squared distance between the centroid of the support polygon and the projection of the center of mass on the ground. Inclusion of the first term, which would move $\boldsymbol{q}_{\rm jnt}$ in the direction of the antigradient of H, is aimed at keeping the center of mass as close as possible to the center of the support polygon, thereby preferring robot configurations that are statically further from instability.

During the integration of (3–4), or (3–5), collision avoidance is continuously checked, together with position and velocity limits for the joints. For a non-stepping motion, equilibrium is also verified. If any of these conditions is violated, the current motion generation is prematurely terminated. Otherwise, integration stops when t_{k+1} is reached. At this point, a feasible joint motion $\boldsymbol{q}_{\rm jnt}(t)$, $t \in [t_k, t_{k+1}]$ has been generated and can be used by the planner.

IV. Planner Overview

The proposed planner builds a tree in the task-constrained configuration space

$$\mathcal{C}_{\rm task} = \{\boldsymbol{q} \in \mathcal{C} : \boldsymbol{f}(\boldsymbol{q}_{\rm spt}, \boldsymbol{q}_{\rm jnt}) = \boldsymbol{y}^*(t), t \in [t_i, t_f]\}$$

with the root at the initial configuration $\boldsymbol{q}(t_i)$. Nodes are configurations of the humanoid, and arcs represent elementary whole-body motions (stepping or non-stepping) that join adjacent nodes and have been verified to be feasible. We emphasize that both nodes and arcs are completely contained in $\mathcal{C}_{\rm task}$. Every node is associated to a time instant t_k, $k = 1, \ldots, N+1$; the tree structure means that multiple nodes may be associated to the same t_k.

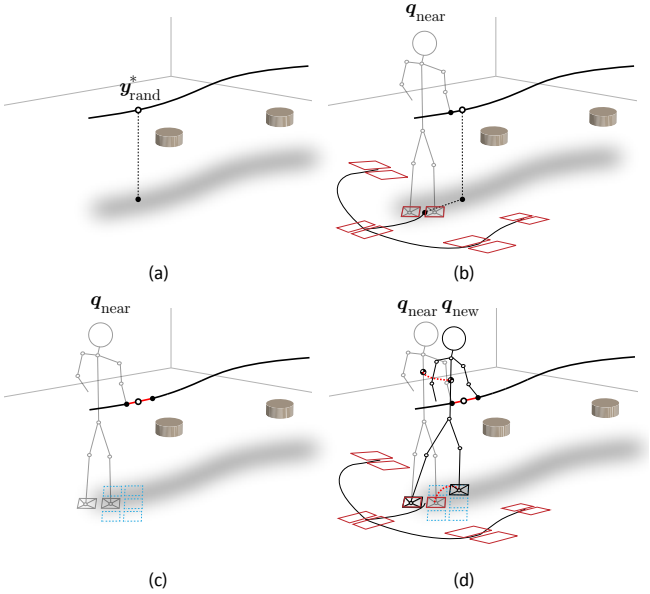

Fig. 2. An iteration of the proposed planner. (a) A random sample is chosen from the assigned task trajectory and its projection on the ground is computed. (b) A configuration $\boldsymbol{q}_{\rm near}$ is extracted from those in the tree using a probability inversely proportional to the distance between such projection and the midpoint between the feet. (c) The portion of the assigned task trajectory starting from $\boldsymbol{q}_{\rm near}$ is extracted (red) and the step generator is called to choose a support foot displacement among the set of primitives (dashed blue). (d) The SPG produces reference trajectories for the swing foot and the center of mass (both dashed red) and the joint motion generator is called to produce a feasible elementary motion. Once this is validated, its final configuration $\boldsymbol{q}_{\rm new}$ is added to the tree.

The generic iteration of the planner, which is illustrated in Figure 2, implements an RRT-like strategy.

First, a random sample $\boldsymbol{y}^*_{\rm rand}$ is chosen from the assigned task trajectory, and its projection on the ground is computed (Figure 2a). Then, a configuration $\boldsymbol{q}_{\rm near}$ is extracted from the current tree using a probability inversely proportional to the Euclidean distance between such projection and the midpoint between the feet (Figure 2b). The idea behind this metric is that motion generation from configurations at which this distance is large is more prone to failure, because the robot is likely to be close to its joint limits or to losing equilibrium.

Once $\boldsymbol{q}_{\rm near}$ has been identified with the associated time, say, t_k, the portion of $\boldsymbol{y}^*(t)$ between t_k and t_{k+1} is acquired and the motion generator is called. As explained in the previous section, this module will first choose a displacement for the support foot in the set of primitives (Figure 2c), and then invoke the SPG to produce reference trajectories for the swing foot and (if $\boldsymbol{u}^k \neq \boldsymbol{0}$) the center of mass. Finally, the joint motion generator computes a joint motion that realizes the assigned task portion as well as these reference trajectories over $[t_k, t_{k+1}]$. If this motion is successfully validated, the final humanoid configuration $\boldsymbol{q}_{\rm new}$ is added to the tree (Figure 2d); otherwise, the procedure is repeated.

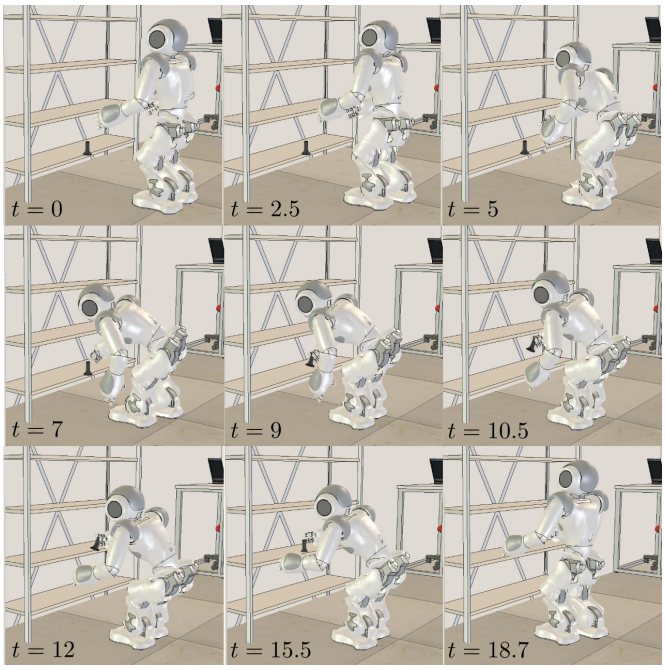

Fig. 3. Pick and place: snapshots from a solution.

V. Planning Experiments

The proposed planner has been implemented in V-REP (a robot simulator developed by Coppelia Robotics) on a MacBook Pro dual-core running at 2.66 GHz. The chosen robotic platform is NAO by Aldebaran Robotics. This small humanoid has 5 degrees of freedom in each leg, 5 in each arm, 1 in the pelvis, 2 in the neck, and 1 in each finger. Joint limits were taken from the official documentation.

We consider two planning scenarios. In the first, the robot must pick up an object from the lower shelf of a bookcase and place it on an upper shelf. To this end, its right hand is assigned a suitable trajectory lasting 18 s. The duration of elementary motions is $T = 2$ s. For step generation, uniform probability is used to extract support foot displacements from the set of primitives, in which $\delta_x = 0.03$ m, $d_{\min} = 0.1$ m, $\delta_y = 0.01$ m, $\delta_\theta = 7.5°$ and $M = 2$. For joint motion generation, we use $\boldsymbol{K} = \mathrm{diag}\{2, 2, 2\}$ in eq. (3) and $\eta = 1.6$ in eq. (5), while $\boldsymbol{w}_{\mathrm{rnd}}$ in eqs. (4-5) is generated using uniform probability and a norm limit at 0.4 rad/sec. Numerical integration of joint velocities is performed with a 4-th order Runge-Kutta algorithm using a step of 0.05 s.

A few snapshots from a solution computed by our planner are shown in Fig. 3. The robot moves from its start configuration at $t = 0$ s and it has to take several steps before grasping the object at $t = 9$ s and placing it on the upper shelf at $t = 15.5$ s. In this case, the planner has chosen to perform the actual pick and place operation entirely in double support; note also how collisions with the bookcase are carefully avoided. Once the object is released, the robot takes a last step forward and successfully completes its task at $t = 18.0$. After this, the robot performs a brief self-motion for achieving the best possible final posture from the viewpoint of static balance.

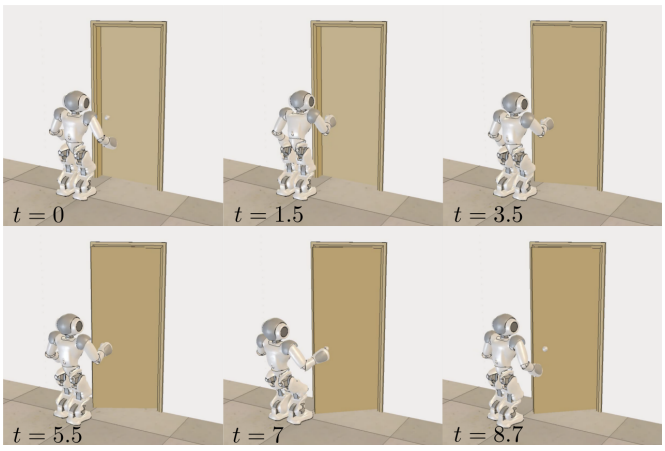

Fig. 4. Opening a door: snapshots from a solution.

In the second problem, the robot must open a door by grasping its knob and moving it along an arc of 45°. The duration of this task is 8 s, while T was set at 1.6 s in this case. All other parameters used by the planner are set to the same values as in the previous experiment.

Figure 4 shows a selection of snapshots from a solution found by our planner for this problem. At the start, NAO approaches the knob with its right hand. Knob grasping occurs at $t = 1.5$ s. Since its starting configuration is very close to the door, the robot takes a few backward steps between $t = 1.5$ s and $t = 7$ s in order to be able to open the door without colliding with it. Note how NAO continues opening the door while stepping back. At $t = 7$ s, the door opening phase ends. Then, the robot releases the knob and takes one more backward step. Also in this case, the plan is concluded by a self-motion.

Table I collects some data that are related to the planner performance in both problems, i.e., the time needed by the planner to generate a solution, the number of nodes contained in the final tree and the mean value of the norm of the task error throughout the duration of the task (not only at nodes). Since our planner is randomized, these data are averaged over 10 executions of the planner for each problem.

Note how both the execution time and the tree size are larger for the first problem than for the second. The reason for this is twofold. First, the task duration for the pick and place scenario is longer. Second, the geometry of that scene requires the robot to stretch its arm to reach the object on the lower shelf. As a consequence, many candidate configurations are discarded by the planner for violating the joint limits or colliding with the bookcase, and therefore planning time increases. In both cases task accuracy is very high and insensitive to the complexity of the problem.

To further validate the proposed method, we have performed a dynamic playback of the planned motions in V-REP; i.e., we have used the generated joint trajectories as reference signals for the joint-level robot controllers and enabled a full physical simulation of the humanoid, including multibody dynamics and interaction with the environment

data	pick and place	opening a door
execution time (s)	5.41	2.61
tree size (# nodes)	81.6	55
mean task error (m)	$4.44{\cdot}10^{-4}$	$4.2761{\cdot}10^{-4}$

TABLE I
PLANNER PERFORMANCE

(foot contact, object grasping and releasing). The obtained NAO motions, shown in the video accompanying this paper, are virtually identical to the reference motions; in particular, this confirms that static equilibrium is effectively achieved.

VI. CONCLUSIONS

In this paper, we have considered the problem of planning the motion of a humanoid robot that must execute a manipulation task, possibly requiring stepping, in an environment cluttered by obstacles. The proposed method explores the submanifold of the configuration space that is admissible with respect to the assigned task and at the same time satisfies other constraints, including humanoid equilibrium. The exploration tree is expanded using a hybrid scheme that simultaneously generates footsteps and whole-body motions. The algorithm has been implemented for the humanoid robot NAO and validated through dynamic playback in the V-REP environment.

A distinctive feature of the developed planner is that, differently from the existing literature, its application is not limited to 'regulation' tasks, i.e., tasks defined only in terms of a final desired value, such as a grasp posture for one hand. Indeed, by handling tasks that are specified through actual trajectories, we can allow the robot to perform more complicated operations, such as opening a door. On the other hand, if only $\boldsymbol{y}^*(t_f)$ is assigned, a suitable *approach trajectory* in the task space can be easily designed and incorporated in the assigned task to recover our problem setting. Another possibility is to simply set $\boldsymbol{y}^*(t) = \boldsymbol{y}^*(t_f)$ for all t; in this case, the motion generation scheme (3) will naturally produce as approach trajectory a linear motion in the task space with exponential convergence speed.

Another interesting aspect of our approach is its independence from the Stepping Pattern Generator, which is in fact assumed to be external to the planner. This means that both statically and dynamically balanced walking can be embedded in our plan, without the need of any post-processing phase. An actual Walking Pattern Generator may be also included to produce longer sequences of stepping motions directly; this can be desirable when the assigned task implicitly requires long transfers ('open the door at the end of the corridor'). We are currently working on this extension.

Although in this paper we have essentially focused on manipulation tasks, we emphasize that the proposed framework is general and therefore can be applied also to tasks of different nature, such as navigation, observation, and so on. Clearly, the heuristic used by the motion generator and the metric adopted in the planner should be adjusted to the specific nature of the task.

Other directions for further research include the following:

- handle *moving* obstacles, along the lines of [18];
- plan in the presence of torque limits, using a second-order motion generation scheme as done in [19];
- accept composite tasks that require physical interaction with the environment and therefore specify both motion and force profiles.

REFERENCES

[1] B. Siciliano and J.-J. Slotine, "A general framework for managing multiple tasks in highly redundant robotic systems," in *Fifth International Conference on Advanced Robotics*, 1991, pp. 1211–1216.

[2] O. Kanoun, F. Lamiraux, and P. B. Wieber, "Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task," *IEEE Trans. on Robotics*, vol. 27, no. 4, pp. 785–792, 2011.

[3] O. Kanoun, J.-P. Laumond, and E. Yoshida, "Planning foot placements for a humanoid robot: A problem of inverse kinematics," *Int. J. of Robotics Research*, vol. 30, no. 4, pp. 476–485, 2011.

[4] J. Kuffner, J.J., K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue, "Footstep planning among obstacles for biped robots," in *2001 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2001, pp. 500–505.

[5] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue, "Motion planning for humanoid robots," in *Proc. 11th Int. Symp. of Robotics Research (ISRR 2003)*, 2003.

[6] J. Chestnutt, M. Lau, G. Cheung, J. Kuffner, J. Hodgins, and T. Kanade, "Footstep planning for the honda ASIMO humanoid," in *2005 IEEE Int. Conf. on Robotics and Automation*, 2005, pp. 629–634.

[7] N. Perrin, O. Stasse, L. Baudouin, F. Lamiraux, and E. Yoshida, "Fast humanoid robot collision-free footstep planning using swept volume approximations," *IEEE Trans. on Robotics*, vol. 28, no. 2, pp. 427–439, 2012.

[8] J. Pettré, J.-P. Laumond, and T. Siméon, "A 2-stages locomotion planner for digital actors," in *2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2003, pp. 258–264.

[9] E. Yoshida, I. Belousov, C. Esteves, and J.-P. Laumond, "Humanoid motion planning for dynamic tasks," in *2005 5th IEEE-RAS Int. Conf. on Humanoid Robots*, 2005, pp. 1–6.

[10] J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue, "Dynamically-stable motion planning for humanoid robots," *Autonomous Robots*, vol. 12, pp. 105–118, 2002.

[11] K. Harada, M. Morisawa, K. Miura, S. Nakaoka, K. Fujiwara, K. Kaneko, and S. Kajita, "Kinodynamic gait planning for full-body humanoid robots," in *2008 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2008, pp. 1544–1550.

[12] F. Burget, A. Hornung, and M. Bennewitz, "Whole-body motion planning for manipulation of articulated objects," in *2013 IEEE Int. Conf. on Robotics and Automation*, 2013.

[13] K. Hauser and V. Ng-Thow-Hing, "Randomized multi-modal motion planning for a humanoid robot manipulation task," *Int. J. of Robotics Research*, vol. 30, no. 6, pp. 678–698, 2011.

[14] S. Dalibard, A. El Khoury, F. Lamiraux, A. Nakhaei, M. Taïx, and J.-P. Laumond, "Dynamic walking and whole-body motion planning for humanoid robots: An integrated approach," *Int. J. of Robotics Research*, vol. 32, no. 9–10, pp. 1089–1103, 2013.

[15] K. Bouyarmane and A. Kheddar, "Humanoid robot locomotion and manipulation step planning," *Advanced Robotics*, vol. 26, no. 10, pp. 1099–1126, 2012.

[16] G. Oriolo and M. Vendittelli, "A control-based approach to task-constrained motion planning," in *2009 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2009, pp. 297–302.

[17] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," in *2003 IEEE Int. Conf. on Robotics and Automation*, 2003, pp. 1620–1626.

[18] M. Cefalo, G. Oriolo, and M. Vendittelli, "Task-constrained motion planning with moving obstacles," in *2013 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2013, pp. 5758–5763.

[19] M. Cefalo and G. Oriolo, "Dynamically feasible task-constrained motion planning with moving obstacles," in *2014 IEEE Int. Conf. on Robotics and Automation*, 2014.