# A Hierarchical Framework for Realizing Dynamically-stable Motions of Humanoid Robot in Obstacle-cluttered Environments

Zhaopeng QIU, Adrien ESCANDE, Alain MICAELLI and Thomas ROBERT

***Abstract*—We address in this paper a hierarchical framework for planning and simulating dynamic motions of humanoid robots in cluttered environments. The robot is aimed to realize a dynamic multi-step motion via a series of support (contact or grasp) configurations. In this framework, a global CoM (Center of Mass) trajectory is firstly generated which ensures the balance during the motion; the whole-body collision-free motion planning is then carried out locally for each transition phase between support configurations; finally the generated trajectories (CoM, end-effectors, joints) serve as control references for realizing the dynamic motion. This framework has been tested in a car-ingress scenario.**

## I. INTRODUCTION

### A. Context

The high dimension of degrees of freedom (DoF) of humanoid robots endues them good flexibility to adapt their movements to various environments. However, the high-dimensional DoFs make the motion planning a quite challenging task: as one can imagine, when a humanoid robot realizes a human-like motion in cluttered environments like factories, offices or inside vehicles, it must be able to autonomously self-navigate, maintain dynamic balance, build and remove supports, avoid collisions with obstacles and accomplish manipulations. Unlike simple motions such as standing or walking on even ground, the motions in cluttered environments are usually much more complex: they may be acyclic, dynamic, and associated with various supports.

Realization of a humanoid's dynamic motions in a cluttered environment involves multiple problematics including motion planning, balance maintenance, collision avoidance, motion control, inverse kinematics (IK), etc. Recent progress in researches on these problematics brings more and more exciting achievements, but up to now, there is not yet an universal and efficient approach for realizing humanoid motions in rather cluttered environments. Our study aims to explore an efficient approach to this problem.

This paper is organized as follows: In section II we present the related works on humanoid robot motion planning and whole-body posture computing; In sections III-VI, we present the hierarchical framework as well as the associated approaches and algorithms at each level; The implementation of this framework in a practical scenario and the results are presented in section VII; Finally the conclusion and perspectives concerning this study are given in section VIII.

Zhaopeng QIU and Thomas ROBERT are with the *IFSTTAR - Université Lyon 1, LBMC UMR_T9406, France*; Zhaopeng QIU and Alain MICAELLI are with *CEA LIST, Interactive Simulation Laboratory, France*; Adrien ESCANDE is with the *CNRS-AIST JRL (Joint Robotics Laboratory) UMI3218/CRT, AIST, Tsukuba, Japan*.

## II. RELATED WORK

In recent years, many researchers have contributed their efforts in motion planning for humanoid robots. Numerous methods have been proposed and successfully implemented in digital or real humanoid robots.

Escande presents in [4] an algorithm for planning humanoid robot's contacts in very constrained environments based on inverse kinematics, optimization and graph searching techniques. In this algorithm, a sequence of optimal contacts (between any part of the body and any feasible environment surface) are automatically generated which can navigate the robot to realize its quasi-static collision-free motion in very complex scenarios. Zhang presents in [21] an approach for planning whole-body motion by decomposing the whole-body planning problem into a sequence of low-dimensional problems. A constrained coordination sampling-based planning approach is adopted for solving each sub-problem incrementally under the constraints of collision-freeness and statical balance. Hauser presents in [7] a non-gaited locomotion planner for generating multi-step motions of humanoid robots over uneven and sloped terrains. Numerical IK method is used to satisfy the closed chain kinematics constraint. A PRM (Probabilistic Roadmap) planner is used to plan the transition for each step between two contact configurations. In [7], only static equilibrium is considered, namely limiting the position of the CoM of robot.

The above-mentioned methods have shown good results for planning quasi-static movements, however, they would be no longer valid when dynamic motions are desired as in most situations in the real life. The timing consideration required in the dynamic case brings more difficult balance constraints and higher computational complexity to the planning task.

Kuffner and his colleagues have obtained excellent achievements on a series of topics ([11], [12], [13]) from footstep planning to dynamic motion generation. In [12], they present a planning framework in which the RRT (Rapidly-exploring Random Tree) method is for the first time implemented in humanoid robot motion planning. A collision-free statically-stable motion is firstly generated and it is furthermore converted into a dynamic motion by being zoomed in time. The dynamic balance is guaranteed by filtering the output path via a dynamic filter "AutoBalancer" [9] based on ZMP (Zero-moment point) criterion. Yoshida et al. have presented their planning method [18], [19], [20] for planning the motions of the robot HRP-2. The randomized planning method can adjust the state dimension, namely the controlled DoFs in the RRT algorithm. The dynamic balance

is guaranteed by confining the ZMP inside the support polygon of the robot's feet. The idea of tuning the number of controlled states has inspired our work, but a general method for adjusting the controlled state dimension is not presented in their work. Harada presents in [6] a walking pattern generator based planning method for humanoid robot. Collision avoidance is firstly excluded in the motion generation. Then a PRM method is carried out for each period in which collisions occur.

Despite the numerous works, planning dynamically-stable motion in obstacle-cluttered environments is still an open issue.

## III. OVERVIEW OF THE WHOLE FRAMEWORK

The scheme of the hierarchical framework in our study is shown in Fig. 1. It generates and realizes the humanoid's motion in a cluttered environment via three steps:

1) At the global level, by pre-defining a sequence of support configurations, a global robust CoM trajectory with timing information is firstly generated by an optimization-based method. The humanoid robot maintains its dynamic balance during the motion by tracking this trajectory.

2) At the local level, the whole-body collision-free motion that tracks the imposed CoM trajectory is generated piecewisely for all the transition phases between stances. A local sampling-based method is associated with a flying end-effector (a hand or a foot) for planning locally its trajectory.

3) At the control level, the generated trajectories (CoM, end-effector, joints) serve as control references so that the humanoid can realize the motion by virtue of dynamic controller.

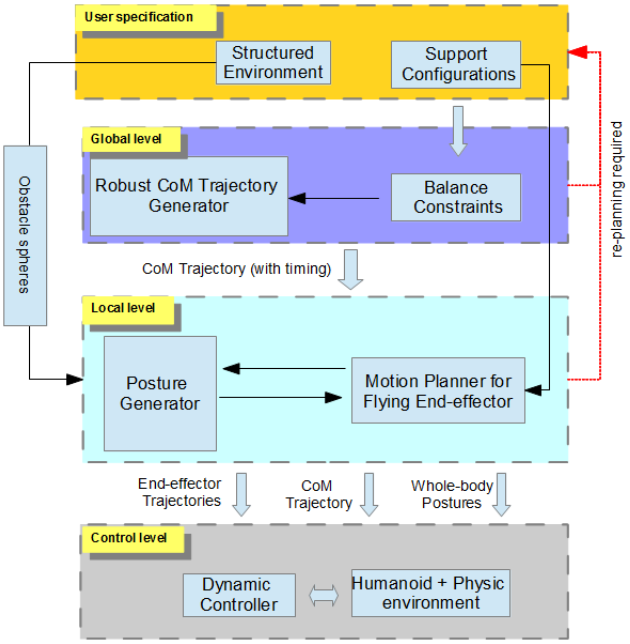


Fig. 1. The hierarchical framework for realizing humanoid's whole-body dynamic motion in cluttered environments.

When applying this framework to realize motions in a desired scenario, we should firstly supply the structured environment including geometric primitives of all the objects in it. A sequence of pre-defined support configurations as well as some kinematic constraints for the motion are also required.

Detailed methodologies at each level of the framework will be presented in the following sections.

## IV. GLOBAL ROBUST COM TRAJECTORY

At the first level, we compute a 3D CoM trajectory that will ensure the humanoid's dynamic balance during its motion. A method based on NURBS (Non-uniform rational B-spline) and optimization technique is explored. The balance criterion adopted in this method has been presented in [15], [8], [1]. It is formulated in our study based on a simplified humanoid model (see Fig. 2).

### A. Balance constraint

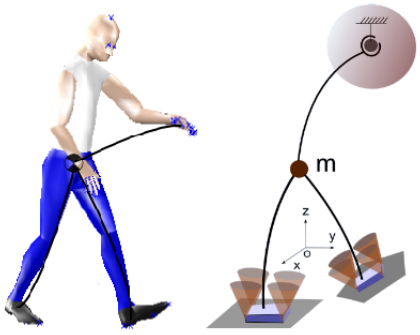


Fig. 2. A mass-point model representing a humanoid robot with all its mass at its CoM: $\mathbf{p} = [x_{com}, y_{com}, z_{com}]^t$. A contact force is limited by the frictional cone. An upper bound is imposed on the magnitude of a grasping force.

The CoM trajectory is expressed as:

$$\mathbf{p}(t) = [x(t), y(t), z(t)]^t, t \in [0, t_f] \tag{1}$$

The balance constraint is expressed as (we denote $\mathbf{p}(t)$ as $\mathbf{p}$ for short):

$$\mathbf{A}_i \mathbf{w}_p = \mathbf{A}_i \begin{bmatrix} m(\ddot{\mathbf{p}} - \mathbf{g}) \\ m\hat{\mathbf{p}}(\ddot{\mathbf{p}} - \mathbf{g}) \end{bmatrix} \leq \mathbf{b}_i \tag{2}$$

where

- $\mathbf{A}_i$ and $\mathbf{b}_i$ are constant which depend only on the $i$-th support configuration;
- $\mathbf{w}_p$ is named as "pseudo wrench" or "generalized wrench" which describes dynamics of the model;
- $\hat{\mathbf{p}}$ is the $3\times3$ *skew-symmetric* matrix of vector $\mathbf{p}$;
- $\mathbf{g}$ is the gravity vector.

Equation (2) defines a *polytope* [5] that is the admissible space for $\mathbf{w}_p$. Suppose that $\mathbf{A}_i$ and $\mathbf{b}_i$ are normalized H-reprentation of this polytope, then the stability margin ([16], [1]) is the smallest distance from point $\mathbf{w}_p$ to facets of this polytope:

$$\xi = min(\mathbf{b}_i - \mathbf{A}_i \mathbf{w}_p) \tag{3}$$

### B. Spline trajectory representation

Suppose that the humanoid realizes its motion via $n_t$ support configurations. We denote a timing vector for its motion:

$$\mathbf{T}_{dist} = [t_0 = 0, t_1, t_2, \cdots, t_{n_t} = t_f] \tag{4}$$

which includes the entering time for each of the $n_t$ support configurations and the total motion duration $t_f$.

The trajectory is represented by a clamped 3D NURBS of $k$-th degree with a knot vector (defined from 0 to 1):

$$\mathbf{T} = [\underbrace{0, \cdots, 0}_{k+1}, \frac{t_1}{t_f}, \cdots, \frac{t_{n_t-1}}{t_f}, \underbrace{1, \cdots, 1}_{k+1}] \tag{5}$$

and control points (of the same weight):

$$\mathbf{a} = [\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z]^t \tag{6}$$

Both the timing vector and the control points are chosen as variables that will be solved by optimization. The B-spline as well as its derivatives can be expressed as functions of the knots vector and the control points:

$$\mathbf{p}_s^{(d)}(\bar{t}) = f^{(d)}(\mathbf{T}, \mathbf{a}, \bar{t}), d = 0, 1, \cdots, k, \bar{t} \in [0, 1] \tag{7}$$

By scaling in time, the real trajectory in Equation (1) and its derivatives are mapped with the spline trajectories in Equation (7) as:

$$\mathbf{p}^{(d)}(t) = [x^{(d)}(t), y^{(d)}(t), z^{(d)}(t)]^t = \frac{1}{t_f^d}\mathbf{p}_s^{(d)}(\bar{t}),\ t = \bar{t} * t_f \tag{8}$$

### C. Optimization problem statement

We compute the spline using optimization techniques. The CoM trajectory is expected to be stable and robust, thus we choose to maximize the lower bound of its stability margins. Meanwhile, this trajectory should be realist and dynamic, thus we add objectives of minimizing the jerk and the total motion duration. The optimization problem is formulated as follows:

$$\begin{aligned}
Variable:&\ \mathbf{u} = [\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z, t_1, \cdots, t_{n_t}, \xi_{min}]^t \\
Objective:&\ \mathbf{max}(l_b),\ \mathbf{min}(\dddot{\mathbf{p}})\ and\ \mathbf{min}(t_{n_t}) \\
Subject\ to:&\\
&\xi_{min} > 0 \\
&\mathbf{for}\ \ (t_{i-1} \le t < t_i, i \in [1, 2, \cdots, n_t]): \\
&\quad (\mathbf{b}_i - \mathbf{A}_i \mathbf{w}_p(t)) \ge \xi_{min} \\
&\mathbf{A}_{eq}\mathbf{u} = \mathbf{b}_{eq}, \mathbf{A}_{neq}\mathbf{u} \le \mathbf{b}_{neq}, \\
&f_{nl}(\mathbf{u}) \le \mathbf{0}
\end{aligned}$$

where

- $\xi_{min}$ is the lower bound for stability margins throughout the motion;
- $\mathbf{A}_{eq}, \mathbf{b}_{eq}, \mathbf{A}_{neq}$ and $\mathbf{b}_{neq}$ are linear constraint matrices or vectors for imposing initial or final conditions and lower bounds for time durations;
- function $f_{nl}(\mathbf{u})$ defines some non-linear constraints in this problem such as the geometric constraints (e.g. leg length).

Optimization solver calculates the optimal trade-offs among multiple objectives mentioned above. Once the optimization problem is successfully solved, we obtain the CoM trajectory and time durations of all the transition phases between stances.

## V. Local Motion Planning

At the second level of our framework, we aim to plan locally the humanoid's whole-body collision-free motion. The trajectories of end-effectors and joints generated at this level will finally be used at the third level for realizing the planned dynamic motion on a humanoid.

### A. Problem statement

We have obtained at the first level the CoM trajectory and the time distribution among the support configurations. Thereafter, we plan the humanoid's whole-body motion by taking into account the CoM constraint, kinematic constraints of supports and collision avoidance constraints. These constraints vary as the humanoid moves, thus we carry out the planning work locally (for a transition phase instead of the entire motion). The problem at this level can be formulated as:

$$\begin{aligned}
\mathbf{For}:&\ \ t \in [t_{i-1}, t_i],\ i = 1, 2, \ldots, n_t \\
\mathbf{Find}:&\ \ [\mathbf{p}_e(t), \boldsymbol{\theta}_e(t)]\ and\ \mathbf{q}_w(t) \\
s.t.:&\ \ supports\ kinematics,\ CoM\ position\ and\ collision\text{-}freeness\ constraints
\end{aligned}$$

where

- $\mathbf{q}_w(t)$ is a whole-body collision-free posture;
- $[\mathbf{p}_e(t), \boldsymbol{\theta}_e(t)]$ is posture (position-orientation) vector of a controlled end-effector.

### B. Collision-free posture generator

A *posture generator* [3] is used in this work (see Fig. 3). By imposing configurations of one or more bodies as well as the CoM position, the posture generator can compute a whole-body configuration that is collision-free (if it exists) using optimization technique. A SQP (Sequential Quadratic Programming) optimization solver is associated in posture generator for solving the non-linearly constrained problem.

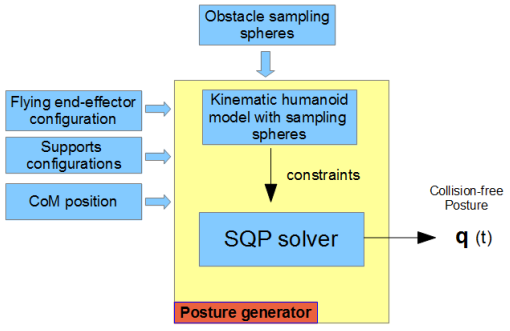


Fig. 3. Posture generator for computing whole-body posture with multiple constraints such as CoM position, body configuration, collision avoidance, etc.

Collision avoidance is integrated in the posture generator as a series of inequality constraints. In order to save the computing time, we adopt a *sphere-sphere* model for collision checking. A series of bounding spheres are sampled for some body segments of the humanoid robot and for the obstacles in the environment (see Fig. 4). To avoid collisions between a body segment and an obstacle, the distance between each pair of their sampled spheres must be greater than zero. Self-collision avoidance for a pair of body segments are defined in the same manner.

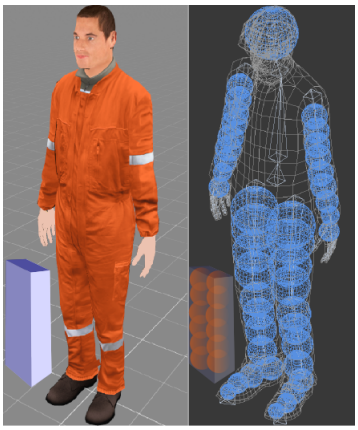

Fig. 4. Collision avoidance: some body segments of the humanoid robot and the obstacles are represented by a series of spheres for collision checking.

### C. Flying end-effectors

We supply two options for solving the IK problem with open chains of flying end-effectors: interpolating method and sampling-based motion planning method.

We recall that the initial and final configurations of a flying end-effector in each of its flight phases are already given in the pre-defined support configuration sequence (see Fig. 5). Meanwhile, timing of each flight phase has also been generated at the first level.

An option for solving the open chain IK problem is to interpolate the flying end-effector's trajectory using polynomial functions. This trajectory then is inputted into the posture generator for computing the whole-body collision-free motion in this flight phase. This method is rapid but can only be used in rather simple cases, such as a small step in a local environment without nearby obstacles.

The second option is to use sampling-based motion planning method. Since the initial and final configurations as well as the timing are known, we choose *Bi-RRT* (Bi-directional Rapidly-exploring Random Trees) method for motion planning in this case. Instead of exploring in humanoid robot's whole-body *C*-space (configuration space), our Bi-RRT method is implemented in the flying end-effector's configuration-time space (*C-t* space) which is a low-dimension (7D) space (see Fig. 6). The time dimension in this method makes it possible to impose the kinematic/kinodynamic constraints. The metric distance in the

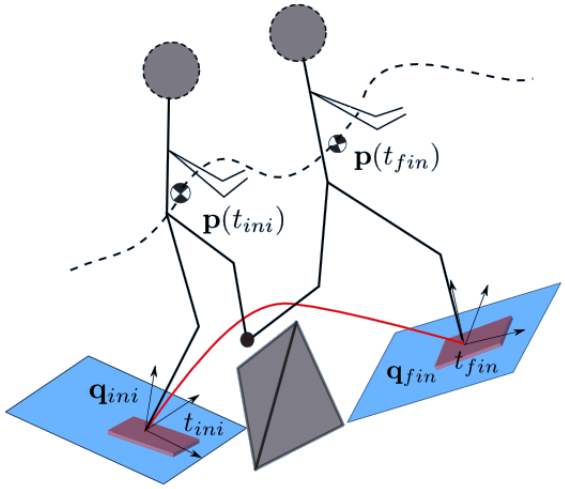


Fig. 5. Illustrating the local planning for a flying foot associated with the imposed CoM trajectory represented by the dashed curve.

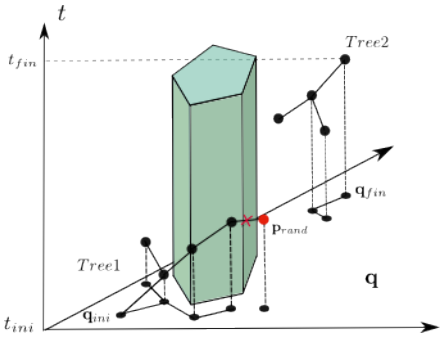


Fig. 6. Bi-RRT in the configuration-time space of an end-effector. The new random point (red) is rejected since the average velocity of the new path segment surpasses the limit value. The $\mathcal{C}$-obstacle sweeps the time interval and results in a hyper-obstacle in the configuration-time space.

sampling space is computed by balancing position, orientation and time dimensions (see Appendix). An upper bound is imposed on the average velocity of the end-effector in order to avoid unrealism in term of brutal changes of whole-body posture. Of course, the two trees must expand in an unique direction along the time axis (time-increasing for the first tree and time-decreasing for the second one).

Algorithm 1 addresses the Bi-RRT method. The two trees are built by initializing their first node with respectively the given configuration as well as its corresponding time. Since the first node in each tree serves as the root, we impose its $parent$ to be null. The two trees then generate random sampling nodes and try to connect with each other within at most $n_{it}$ iterations. Function *RANDOM-POINT()* serves to generate the random sampling point $\mathbf{p}_{rand}$ within a user-defined sampling space. Then function *EXPAND()* (see Algorithm 2) tries to generate a new node towards the sampling point $\mathbf{p}_{rand}$. Function *NEAREST-NODE()* returns the node $\mathbf{p}_{near}$ of a tree that is nearest to $\mathbf{p}_{rand}$. *CHECK-VELOCITY()* checks whether the line connecting $\mathbf{p}_{near}$ and $\mathbf{p}_{rand}$ violates the velocity limit; if not, *LIMIT-METRIC()* generates a new

point (candidate node) $\mathbf{p}_{new}$ lying on the line connecting $\mathbf{p}_{near}$ and $\mathbf{p}_{rand}$ within a limited metric distance with $\mathbf{p}_{near}$. Then function *CHECK-PG()* verifies feasibility of the candidate path segment ($\mathbf{p}_{new}$ to $\mathbf{p}_{near}$): $n_{pg}$ interpolating points at the path segment is inputted one by one into posture generator for imposing the flying end-effector's configuration. If no feasible posture is found, function *EXPAND()* stops and the candidate node $\mathbf{p}_{new}$ is rejected. If function *CHECK-PG()* returns the confirmative signal, $\mathbf{p}_{new}$ is then added into the tree and its $parent$ is labeled as the index of the node $\mathbf{p}_{near}$. Function *GET-INDEX()* returns the index of a node in the node array of the tree.

**Algorithm 1** GENERATE-RRT($\mathbf{q}_{ini}, t_{ini}, \mathbf{q}_{fin}, t_{fin}$)

$Tree1.\mathbf{p} = (\mathbf{q}_{ini}, t_{ini}), Tree2.\mathbf{p} = (\mathbf{q}_{fin}, t_{fin})$
$Tree1.parent = 0, Tree2.parent = 0$
**for** $i = 1$ **to** $n_{it}$ **do**
  $\mathbf{p}_{rand} = [\mathbf{q}_{rand}, t_{rand}] \leftarrow$ RANDOM-POINT()
  EXPAND($Tree1$, $\mathbf{p}_{rand}$)
  **if** IS-CONNECTED($Tree1$, $Tree2$) **then**
    break
  **end if**
  EXPAND($Tree2$, $\mathbf{p}_{rand}$)
  **if** IS-CONNECTED($Tree1$, $Tree2$) **then**
    break
  **end if**
**end for**

**Algorithm 2** EXPAND($Tree$, $\mathbf{p}_{rand}$)

$\mathbf{p}_{near} \leftarrow$ NEAREST-NODE($Tree$, $\mathbf{p}_{rand}$)
**if** CHECK-VELOCITY($\mathbf{p}_{near}, \mathbf{p}_{new}$) $= TRUE$ **then**
  $\mathbf{p}_{new} \leftarrow$ LIMIT-METRIC($\mathbf{p}_{near}, \mathbf{p}_{rand}, d_{lim}$)
  **for** $i = 1$ **to** $n_{pg}$ **do**
    $e \leftarrow$ CHECK-PG($Tree.\mathbf{p}_{near}, \mathbf{p}_{new}$)
    **if** $e$ = FALSE **then**
      Break
    **end if**
  **end for**
  $Tree.\mathbf{p}$.PUSHBACK($\mathbf{p}_{new}$)
  $Tree.parent$.PUSHBACK(GET-INDEX($\mathbf{p}_{near}$))
**end if**

The two trees connect to each other when the function *IS-CONNECTED()* returns a confirmative signal. This function examines the nearest nodes pair between two trees. When their distance is smaller than a threshold value, it verifies the velocity constraint and the feasibility of the path segment connecting this pair of nodes. If valid, the algorithm connects the two trees, stops the iterations and returns the solution path.

A smoothing algorithm is carried out for the generated path in a iterative way. It samples randomly two points lying on two different segments of the path and tests the new segment connecting the two sampling points using functions *CHECK-VELOCITY()* and *CHECK-PG()*. The path is updated by replacing all portions between the two points with a new valid segment. This smoothing operation is carried out for a desired number of iteration. The smoothed trajectory as well as the accordingly generated whole-body posture are then exported to motion control level.

## VI. Dynamic simulation: execution of the generated motion

Once the work of the first two levels has been successfully carried out, one should have obtained a set of trajectories of the CoM, the controlled end-effectors and the joints. All these data are thereafter used as motion references for executing the generated movement on a humanoid via dynamic controllers. Control strategies and techniques in this context have been broadly presented in many studies ([10], [17], [14], [2]). Multiple tasks should be defined for the controller in the dynamic simulation. These tasks result in multiple objectives for optimization solver of the controller which are weighted according to their priorities. In our work, several main tasks are defined including (in order of their priorities):

**Center of mass**: the dynamic balance is ensured by the global CoM trajectory. Thus, the CoM of the humanoid should track precisely its pre-defined robust trajectory, including the position, velocity and acceleration references;

**End-effector**: the posture of a flying end-effector should track its trajectory generated by local planners, including the position, orientation, translational and angular velocities, and accelerations;

**Supports**: a contact or a grasp should be activated or deactivated during the movement according to the motion strategy. The wrenches applied at a contact or grasp are taken into account in the motion control strategy;

**Posture**: the joints are also controlled and the reference posture is exported to control the dynamic motion at the third level of our framework.

## VII. Case study: application and results

This section presents the implementation of our framework in a complex car-ingress scenario.

An example motion of this scenario is shown in Fig. 7 which is a reconstructed motion based on real human motion data recorded in Motion Capture experiments. In this implementation, we aim to realize the car-ingress motion in the same environment via the same sequence of support configurations as in this example. The geometric features of objects in this structured environment are listed in Table I. The sampled spheres of obstacles (see Fig. 10) in this environment are automatically generated beforehand. The sequence and placements of supports in this motion are listed in Table II and Table III.

### A. Global CoM trajectory

After its buttock touches the seat, the humanoid will be very safe from losing balance. Thus at the first level,

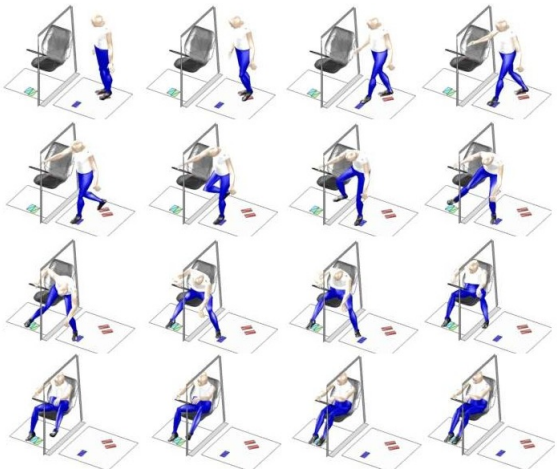

Fig. 7. A reconstructed car-ingress motion using Motion Capture data. One can see the motion strategy in car-ingress scenario.

| Environmental element | Geometric primitives |
|---|---|
| H point | [0, 0, 0] |
| Ground | $z$ = -485$mm$ |
| Floor | $z$ = -231$mm$ |
| Sill | $y$ = -402$mm$, $z$ = -230$mm$ |
| Roof | $y$ = -193$mm$, $z$ = 933$mm$ |
| Steering wheel | $d$ = 360$mm$, $\mathbf{c}$ = [-402, -10, 409] ($mm$), $\theta_{incli} = 55.0°$ |
| Rear pillar | $x$ = 236$mm$, $\theta_{incli} = 80.7°$ |
| Front pillar | $x$ = -941$mm$ |
| Seat | A set of vertices |

TABLE I
ENVIRONMENT PRIMITIVES SPECIFICATION IN THE CAR-INGRESS EXAMPLE

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Left foot | 1 | - | 1 | 1 | 1 | 1 | 1 | - | - | - | 1 | 1 |
| Right foot | 1 | 1 | 1 | - | - | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Left hand | - | - | - | - | - | - | - | - | - | - | - | 1 |
| Right hand | - | - | - | - | 1 | 1 | 1 | 1 | - | 1 | 1 | 1 |
| Buttock | - | - | - | - | - | - | 1 | 1 | 1 | 1 | 1 | 1 |

TABLE II
SEQUENCE OF SUPPORT CONFIGURATIONS: "1" INDICATES A VALID SUPPORT

we generate the CoM trajectory for the period from the beginning of the motion to the instant when the buttock touches the seat. Accordingly, the first 6 support configurations (gray columns in Table II) are used for computing the balance constraints (polytopes) for the 6 transition phases ($n_t$ = 6). A geometric constraint is imposed which limits the distance from CoM to a foot contact center within $1.08m$. The minimum time durations for transitions phases are set to be $[0.15, 0.4, 0.1, 0.5, 0.6, 0.6](s)$. The above-mentioned limits are determined according to experimental data.

The CoM trajectory generation is carried out in Matlab. A 5-th order B-spline which has 17 knots and 11 control points is used to represent the CoM trajectory. The global CoM trajectory (see Fig. 8) has been successfully generated within 3 minutes (on a workstation with Xeon 3.4GHz and 8GB of RAM, same for following levels). The results are given in Tabel IV. This trajectory has a lower bound of $19.4N \cdot m$ for stability margins throughout the movement.

| Support | Position ($m$) | Orientation (Euler angles $x$-$y$-$z$) |
|---|---|---|
| Left foot | [0.1912, -1.0636, -0.4852]<br>[-0.2397, -0.7371, -0.4852]<br>[-0.6512, -0.1054, -0.2317] | [0, 0, 50°]<br>[0, 0, 90°]<br>[0, 0, 90°] |
| Right foot | [0.2834, -0.9368, -0.4852]<br>[-0.6235, -0.0119, -0.2317] | [0, 0, 50°]<br>[0, 0, 90°] |
| Right hand | $\mathbf{p}_{ini}$:[0.3720, -0.8323, 0.2697]<br>$\mathbf{p}_{grasp}$:[-0.4600, -0.0555, 0.5688] | –<br>– |

TABLE III
SUPPORT PLACEMENTS

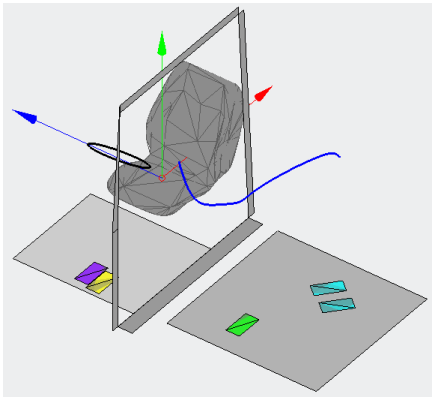

Fig. 8. Generated CoM trajectory illustrated in the structured environment.

| Variable | Value |
|---|---|
| $\mathbf{a}_x(m)$ | [ 0.2350, 0.2350, 0.2492, 0.2763, 0.1825, -0.3005, -0.2687, -0.2684, -0.3894, -0.1642, -0.0454] |
| $\mathbf{a}_y(m)$ | [-1.0112, -1.0112, -0.9860, -0.9350, -0.8688, -0.6894, -0.7915, -0.5306, -0.5296, -0.2661, -0.1463] |
| $\mathbf{a}_z(m)$ | [0.4659, 0.4659, 0.4635, 0.4505, 0.4260, 0.3694, 0.3299, 0.2706, 0.2336, 0.2153, 0.2104] |
| $knots$ | [0, 0, 0, 0, 0, 0, 0.1574, 0.2451, 0.4536, 0.5632, 0.8684, 1, 1, 1, 1, 1, 1] |
| $\mathbf{T}_{dist}$ $(s)$ | [0, 0.7176, 1.1176, 2.0679, 2.5679, 3.9592, 4.5592] |
| $\xi_{min}$ | $19.4N \cdot m$ |

TABLE IV
RESULTS OF THE GENERATED COM TRAJECTORY

### B. Local planning

The right hand is supposed to move along a straight line between its initial position and the grasp position at the steering wheel. Thus its trajectory is generated by interpolating between the two position which is then a 2nd-order polynomial function of time. The first step of the left foot is generated by interpolation with trigonometric functions.

Bi-RRT method is applied for two step phases in which the humanoid steps both feet into the car. During the left foot step phase, the buttock is always in contact with the seat, thus the CoM constraint is not taken into account in the

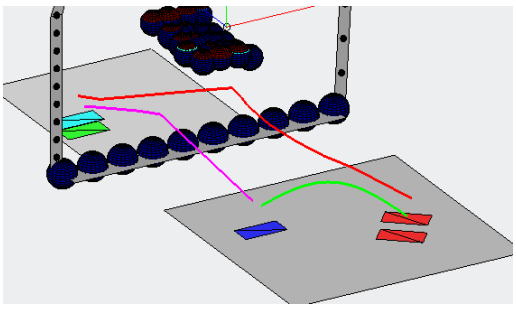

Fig. 9. Feet trajectories generated at the second level for foot flight phases. The body frame of a foot is chosen at its ankle joint. Generated feet orientations are not shown in the figure.

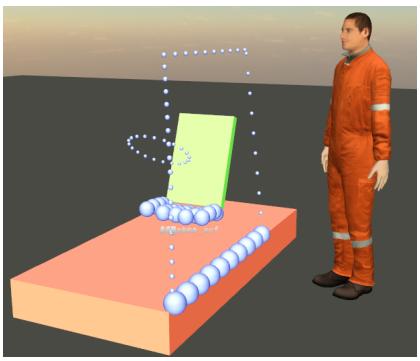

Fig. 10. Structured virtual environment and the humanoid for motion simulation in a car-ingress scenario under XDE®. The humanoid is in its initial stance for realizing the planned car-ingress motion.

posture generator. In several trials, the Bi-RRT can always successfully find feasible paths in about 3-10 minutes for both the two step phases. An example of exported smoothed trajectories for the two feet is shown in Fig. 9. All the trajectories (CoM, feet, right hand, joints) as well as the velocities and accelerations are saved at a sampling time of $0.001s$.

The planning time seems rather long for only an one-step transition. Two reasons can be argued for this problem: 1) collision avoidance adds hundreds of non-linear inequality constraints in SQP solver in the posture generator, which increases significantly the computing complexity; 2) the feasible zone is narrow regarding the sampling space in Bi-RRT method (it needs more than 100 iterations for generating a path with less than 10 nodes), thus most of the planning time is consumed for verifying non-feasible path segments with posture generator.

### C. Motion simulation

The motion execution at the control level is realized using XDE® software developed by CEA-LIST. The structured virtual environment as well as the humanoid in this scenario is shown in Fig. 10. The humanoid consists of 19 body segments and it has 45 degrees of freedom. An impedance controller is associated with the humanoid to actuate its movement. Trajectories generated at the first two levels are imported for defining a series of tracking tasks. The time step of simulation is chosen as $5ms$. Some clips of the motion simulation are shown in Fig. 11. The virtual humanoid robot realizes successfully the car-ingress motion, which validates the methods in our framework.

## VIII. CONCLUSION

In this paper, we present a hierarchical framework for planning dynamic collision-free motions for humanoid robots moving in a cluttered environment. In order to avoid the time-consuming planning task in high-dimensional configuration space of the robot, we decompose and carry out the motion planning task in two steps (first two levels of the framework): at the global level, based on a simplified model, a robust dynamically stable CoM trajectory is generated beforehand which will be used for ensuring the dynamic balance during the movement; then at the local level, whole-body collision-free movement that tracks the global robust CoM trajectory is generated locally. The CoM trajectory, the end-effector trajectories and the whole-body postures are used as tracking references at the third level of the framework for motion simulation. A humanoid robot can realize the generated dynamic motion with help of dynamic controller. We have successfully applied this framework for simulating a car-ingress motion.

This framework shows advantage in the following aspects:

*1) General support types:* In this framework, dynamic balance can be guaranteed for various kinds of supports: planar contact, non-planar contact, grasp. Accordingly, this framework can be applied for realizing humanoid's motions in more complex environments and more general scenarii.

*2) Planning efficiency:* Instead of carrying out time-consuming planning in a global scope, we generate the motion with local planning. In this way, the computational complexity of the planning task can be significantly decreased since it can be executed in a lower dimensional space (for one end-effector instead of the whole body).

In the future, we should improve this work in the following aspects:

- Generation of the sequence of support configurations;
- Whole-body considerations in the generation of the CoM trajectory;
- Improve the naturalness of the final whole-body motion.

## IX. ACKNOWLEDGMENTS

This study is part of a PhD thesis co-funded by two institutes: CEA and IFSTTAR.

## APPENDIX

### METRIC DISTANCE IN BI-RRT METHOD

In Bi-RRT method, each sampled point in an end-effector's $\mathcal{C}$-$t$ space is a 7D vector:

$$\mathbf{p} = [\mathbf{g}, \boldsymbol{\theta}, t] = [x, y, z, \alpha, \beta, \gamma, t] \tag{9}$$

which consists of its position, its orientation and the time.

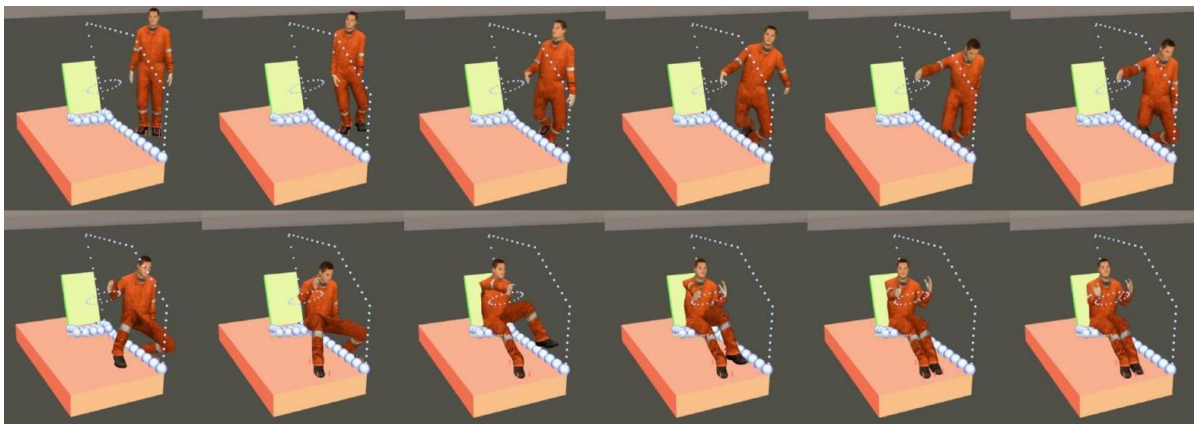

Fig. 11. Clips of the dynamic motion of humanoid in simulation of a car-ingress scenario

Relying on $Rodrigues'\ rotation\ formula$, we define the metric distance between two points $\mathbf{p}_1$ and $\mathbf{p}_2$ in the $\mathcal{C}$-$t$ space as:

$$d(\mathbf{p}_1, \mathbf{p}_2) = \sqrt{\|\mathbf{g}_1 - \mathbf{g}_2\|^2 + \lambda_1 \|log(\mathbf{R}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2))\|^2 + \lambda_2 (t_1 - t_2)^2} \tag{10}$$

where:

- $\lambda_1$ and $\lambda_2$ are scalars for balancing the relative weights among translation, rotation and time metrics.

In order to limit the translational and rotational velocity, we need to compute the change rate of the configuration (can be regarded as the slope regarding the time axis):

$$v(\mathbf{p}_1, \mathbf{p}_2) = \frac{\|\mathbf{g}_1 - \mathbf{g}_2\|}{|t_1 - t_2|} \tag{11}$$

$$\omega(\mathbf{p}_1, \mathbf{p}_2) = \frac{\|log(\mathbf{R}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2))\|}{|t_1 - t_2|} \tag{12}$$

## REFERENCES


[1] S. Barthélemy and P. Bidaud. Stability measure of postural dynamic equilibrium based on residual radius. *Advances in Robot Kinematics: Analysis and Design*, pages 399–407, 2008.

[2] C. Collette, A. Micaelli, C. Andriot, and P. Lemerle. Dynamic balance control of humanoids for multiple grasps and non coplanar frictional contacts. In *Humanoid Robots, 2007 7th IEEE-RAS International Conference on*, pages 81–88. IEEE, 2007.

[3] A. Escande, A. Kheddar, and S. Miossec. Planning support contact-points for humanoid robots and experiments on hrp-2. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 2974–2979. IEEE, 2006.

[4] A. Escande, A. Kheddar, S. Miossec, and S. Garsault. Planning support contact-points for acyclic motions and experiments on hrp-2. In *Experimental Robotics*, pages 293–302. Springer, 2009.

[5] B. Grunbaum and GC Shephard. Convex polytopes. 1967.

[6] K. Harada, S. Hattori, H. Hirukawa, M. Morisawa, S. Kajita, and E. Yoshida. Motion planning for walking pattern generation of humanoid. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 4227–4233. IEEE, 2007.

[7] K. Hauser, T. Bretl, and J.C. Latombe. Non-gaited humanoid locomotion planning. In *Humanoid Robots, 2005 5th IEEE-RAS International Conference on*, pages 7–12. IEEE, 2005.

[8] H. Hirukawa, S. Hattori, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, and M. Morisawa. A universal stability criterion of the foot contact of legged robots-adios zmp. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1976–1983. IEEE, 2006.

[9] S. Kagami, F. Kanehiro, Y. Tamiya, M. Inaba, and H. Inoue. Autobalancer: An online dynamic balance compensation scheme for humanoid robots. In *Proc. Int. Workshop Alg. Found. Robot.(WAFR)*, 2000.

[10] O. Khatib, L. Sentis, J. Park, and J. Warren. Whole body dynamic behavior and control of human-like robots. *International Journal of Humanoid Robotics*, 1(1):29–43, 2004.

[11] J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue. Online footstep planning for humanoid robots. In *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 1, pages 932–937. IEEE, 2003.

[12] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue. Motion planning for humanoid robots under obstacle and dynamic balance constraints. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 1, pages 692–698. IEEE, 2001.

[13] J.J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue. Dynamically-stable motion planning for humanoid robots. *Autonomous Robots*, 12(1):105–118, 2002.

[14] J. Park and O. Khatib. Contact consistent control framework for humanoid robots. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1963–1969. IEEE, 2006.

[15] Z. Qiu, A. Escande, A. Micaelli, and T. Robert. Human motions analysis and simulation based on a general criterion of stability. In *First International Symposium on Digital Human Modeling*. IEA, 2011.

[16] T. Robert, Z. Qiu, J. Causse, A. Escande, and A. Micaelli. A dynamic stability analysis of the sit-to-stand transfer. In *ISB 2011*. ISB, 2011.

[17] L. Sentis and O. Khatib. Synthesis of whole-body behaviors through hierarchical control of behavioral primitives. *International Journal of Humanoid Robotics*, 2(4):505–518, 2005.

[18] E. Yoshida. Humanoid motion planning using multi-level dof exploitation based on randomized method. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3378–3383. IEEE, 2005.

[19] E. Yoshida, I. Belousov, C. Esteves, and J.P. Laumond. Humanoid motion planning for dynamic tasks. In *Humanoid Robots, 2005 5th IEEE-RAS International Conference on*, pages 1–6. IEEE, 2005.

[20] E. Yoshida, C. Esteves, T. Sakaguchi, J.P. Laumond, and K. Yokoi. Smooth collision avoidance: Practical issues in dynamic humanoid motion. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 827–832. IEEE, 2006.

[21] L. Zhang, J. Pan, and D. Manocha. Motion planning of human-like robots using constrained coordination. In *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, pages 188–195. IEEE, 2009.