HUMANOID LOCOMOTION PLANNING FOR VISUALLY-GUIDED TASKS

Jean-Bernard Hayet, Claudia Esteves, Gustavo Arechavaleta-Servin, Olivier Stasse, Eiichi Yoshida

International Journal of Humanoid Robotics

HUMANOID LOCOMOTION PLANNING FOR VISUALLY-GUIDED TASKS

Jean-Bernard Hayet[1], Claudia Esteves[2], Gustavo Arechavaleta[3], Olivier Stasse[4], Eiichi Yoshida[4]

[1] *Centro de Investigación en Matemáticas, CIMAT. Jalisco s/n. Col. Valenciana. 36240. Guanajuato, Guanajuato. México*
E-mail: jbhayet@cimat.mx
[2] *Departamento de Matemáticas, Universidad de Guanajuato. Jalisco s/n. Col. Valenciana. 36240. Guanajuato, Guanajuato. México*
E-mail: cesteves@cimat.mx
[3] *Robotics and Advanced Manufacturing Group. Centro de Investigación y de Estudios Avanzados del IPN, CINVESTAV. Saltillo, Coah, México*
E-mail: garechav@cinvestav.edu.mx
[4] *CNRS-AIST, JRL (Joint Robotics Laboratory), UMI 3218/CRT, Intelligent Systems Research Institute, AIST Central 2, Umezono 1-1-1, Tsukuba, Ibaraki 305-8568 Japan*
E-mail: {olivier.stasse, e.yoshida}@aist.go.jp

Received Day Month Year
Revised Day Month Year
Accepted Day Month Year

In this work, we propose a landmark-based navigation approach that integrates 1) high-level motion planning capabilities that take into account the landmarks position and visibility and 2) a stack of feasible visual servoing tasks based on footprints to follow. The path planner computes a collision-free path that considers sensory, geometric, and kinematic constraints that are specific to humanoid robots. Based on recent results in Movement Neuroscience that suggest that most humans exhibit nonholonomic constraints when walking in open spaces, the humanoid steering behavior is modeled as a differential-drive wheeled robot. The obtained paths are made of geometric primitives that are the shortest in distance in free spaces. The footprints around the path and the positions of the landmarks to which the gaze must be directed are used within a stack-of-tasks (SoT) framework to compute the whole-body motion of the humanoid. We provide some experiments that verify the effectiveness of the proposed strategy on the HRP-2 platform.

Keywords: nonholonomic motion planning; landmark-based navigation; humanoid motion generation.

1. Introduction

One of the first things to determine whether a humanoid robot is autonomous is its ability to navigate safely around obstacles. Several works during the past decade have tackled this important problem by using in general a two-stage algorithm: First, a motion planner computes a feasible collision-free trajectory for the robot

focusing on the geometric, kinematic and dynamic restrictions of the humanoid and its environment. In a second stage, the computed trajectory is executed, generally by using an open-loop control strategy.

In this work we propose a two-stage landmark-based navigation strategy that, in addition to the geometric, kinematic and dynamic restrictions, considers sensory constraints at the planning stage. Here, we focus on visual sensors and aim to ensure the visibility of visual landmarks in the environment. Landmarks play an important role to control the amount of uncertainty on the robot localization within a navigation framework. In this sense, our planner guarantees that at least one landmark in the environment will be visible to the humanoid at all moments of its navigation task. The problem of maintaining at least one landmark visible at all times while ensuring kinematic and dynamic feasibility is stated as a geometric problem of motion planning with equality and inequality constraints.

The main contributions of this work are:

- To state the landmark-based navigation problem as a motion planning problem that can deal with constraints of different natures (i.e., equalities and inequalities) which are specific to humanoid robots. These constraints appear at different levels in the proposed strategy.
- To provide a set of footprints that guarantee the visibility of at least one landmark and that take the humanoid robot from an initial to a goal configuration if the footprints were to be executed with no drift.
- To integrate and validate the whole strategy on a real humanoid platform.

The remaining of this work is organized as follows: in Section 2 we describe what we believe to be the most relevant works related to our approach. In Section 3, our overall strategy is depicted. In Section 4, we explain the different steps of the global path planner and the way the equality and inequality constraints are handled. Section 5 describes the whole-body motion synthesis strategy and the visual servoing task which guarantees the correct execution of the landmark-gazing task computed by the global planner. Section 6 shows experimental results, both in simulation and on the real humanoid robot HRP-2. Finally, in Section 7 we present our conclusions and future work for this proposal.

2. Related work

During the last few years, several motion planners for humanoids robots have been proposed in the literature. In the same way as for any other mechanism, planners for humanoid robots can be local, which has the advantage that generated motions are more reactive, or global, which allows avoiding local minima. Both of these methods can integrate more or less easily equality or inequality constraints such as collision avoidance, verification of joint limits, stability constraints, etc. An overview of motion planning methods that deal with humanoid locomotion, mostly focusing on the planners experimented on the robot HRP-2 can be found in [29]. Within

these planners, it has been a common practice to use a global planner to ensure the feasibility of the desired task and then a local planner to generate and enforce locally the desired motion for the mechanism. For humanoid robots, this family of approaches have been proposed for different kind of tasks and constraints. For example, in [4], the authors propose a global planner for generating a set of individual footsteps that ensure the feasibility of the locomotion task in rough terrains. An online local pattern generator on the robot H7 enforces the task of following the planned footsteps. In [3], the global planner consists of three stages, where in the first, a coarse path for a simplified model of the robot is obtained and in the last, the footsteps for the local pattern generator are computed. A planner handling contact constraints for locomotion that can involve other parts of the humanoid body for additional stability has been proposed in [6]. The global planner determines the type of contact and the position. The local planner, in the form of a numerical inverse kinematics (IK) solver, enforces the contact and closed-kinematic chain constraints. Another multi-contact constraints motion planner has been described in [5] were the authors have a global posture generator and then a local potential field function driving the motion from one posture to another. This planner has been implemented successfully on the HRP-2 platform. A global planner that also considers closed kinematic chains but for two-hand manipulation while walking is presented in [28]. In that work, the authors use a nonholonomic reduced model of the humanoid to produce smooth collision-free trajectories for the robot and the object. The global planner outputs the footprints and the positions of both hands of the humanoid on the object. These local constraints are handled inside a redundant IK solver. In [30] the global planner outputs a smooth walking path with at the same time the grasping constraints for the humanoid to pivot a large box on a given environment. The kind of local planner used in all of these works, a prioritized stack-of-tasks framework (e.g. [16]), has become a common paradigm to use inside global planners for humanoid robots. These stacks usually include, as tasks with a high priority, collision avoidance or robot stability, and as tasks with lower priority, manipulation or visibility constraints, for example. In order to solve as many tasks in the stack as possible, the global planner is in charge of enabling or disabling tasks and changing their priorities (e.g. [13,15]). In [27] the same task-priority redundancy formalism, this time based on the operational space, has been integrated as a local method embedded in a global motion planner for mobile manipulation tasks. This kind of planners integrating a global supervisor with a local stack-of-tasks have already proven to be effective on real experimentation as for example in [21] and [24]. In [24], visual constraints were integrated within the tasks that had to be solved in the stack. In [17], visual constraints are also handled locally for the task of grasping an object while walking within the pile of tasks. In our previous work [8], we have introduced a global motion planner that considers visibility constraints from a global perspective. In this way, the feasibility of the visual task together with the navigation task can be ensured and, in a local manner, both of these tasks are enforced. In the present paper, our contribution relative to [8] is the possibility of ensuring the visibility of

at least one landmark among many during all the length of the path. In addition, we have tested our algorithm experimentally on the humanoid robot HRP-2.

3. Proposed approach

In this work, we propose a motion planner that computes whole-body collision-free walking trajectories for a humanoid robot with the constraint of keeping at least one landmark in sight during all the trajectory. The planner is divided in several stages:

I. First a global motion planner (see Section 4) computes a collision-free path for a reduced model of the system. As a reduced model for the humanoid robot navigating in large environments, we have proposed a differential-drive wheeled robot (DDR). This choice has been done based on recent results in the community of Movement Neuroscience [10] suggesting that most humans exhibit nonholonomic constraints when walking in open spaces [1]. The DDR model is coupled with a limited field-of-view sensor attached to it. Then, the planner computes a collision-free path taking into account the sensor limitations to keep at least one landmark within its field of view during the path at all times. At the end of this stage, the robot gazing profile for the sensor and its footprints are extracted from the path by considering the robot step length and width.

II. Footsteps are then converted into dynamic stable biped locomotion of humanoid through a walking pattern generator. We adopt a method based on the preview control for ZMP [11] for this conversion. This method computes a trajectory for the ZMP from footsteps (position and orientation) to obtain the humanoid CoM reference trajectory for walking motion. This CoM reference trajectory is obtained by a control system based on the cart-table reduced model presented in [11]. The whole-body motion is generated at execution time with a local method based on a Stack-of-Tasks (SoT) approach (see Section 5) where the task of gazing at the landmark is supervised by a visual servoing module that is projected into the null-space of the locomotion task.

The following sections elaborate on each of these stages.

4. Global path planner

Given the modeling of the whole humanoid robot as a *differential-drive robot*, we describe here how to plan 2D trajectories to ensure visibility of at least one landmark all along the computed path. First, we describe precisely the simplified model (Section 4.1) and the constraints that are associated to it. Then we detail the planning algorithm whose overall strategy is described in Fig. 1, which corresponds to : our global planner for the simplified model (detailed in Section 4.3). The blocks in Fig. 1 represent each of the main steps in our strategy, and the section of the paper in which they are described. This planner takes as input the initial and final locations desired for the robot as well as a geometric description of the environment and

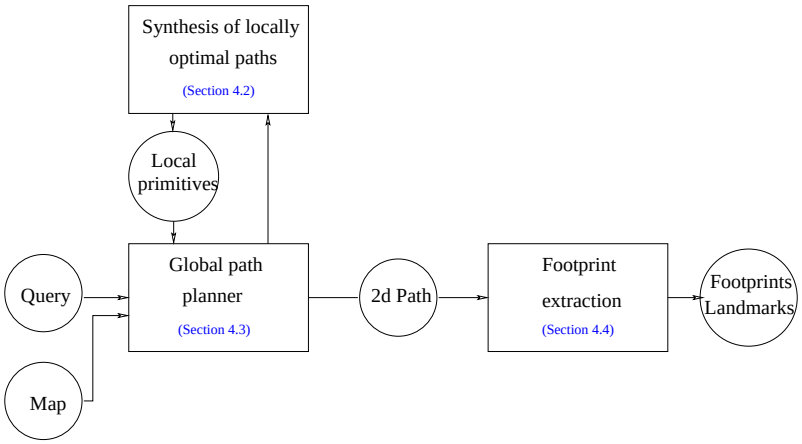


Fig. 1. Global planning strategy for the simplified model : the global planner relies on locally optimal paths synthesis and outputs a 2D path from which a set of footprints and a set of landmark positions (the ones the robot will direct its gaze to) are extracted.

the landmarks positions. Then, a global path is formed recursively using at each step the synthesis of locally optimal paths (described in Section 4.2). From this procedure a 2D path, a concatenation of locally optimal subpaths, is obtained. The output of this strategy are the footprints extracted from this 2D path (Section 4.4), which are used to plan the motion for the whole humanoid, and the landmark that the robot has to maintain in view at each step of the path.

4.1. *Simplified model description and constraints*

The differential-drive simplified model is composed by a subset of the underactuated degrees of freedom of the humanoid kinematic structure. We will denote them by $\mathbf{q}_{ddr} = (x, y, \theta)^T \in SE(2) = \mathcal{C}_{ddr}$ (the configuration of the differential drive simplified model), where (x, y) is the position of the robot on the plane and θ its orientation. We also introduce ϕ, the angle that the sensor makes with respect to the robot orientation. It is not a degree of freedom *per se*, as its value is constrained by the presence of the landmark at the center of the field of view. It should be noted that all the coordinates are relative to the position of the landmark which is currently guiding the robot. This model may capture the human-like walking behavior provided that several constraints expressed on the configuration space $\mathcal{C}_{ddr}$ of the reduced model can be satisfied:

- The human-like walking behavior is characterized by an equality constraint on the tangent bundle of $\mathcal{C}_{ddr}$, i.e. the nonholonomy equation (Equation 1),

$$\dot{x} \sin\theta - \dot{y} \cos\theta = 0. \tag{1}$$

- The landmark visibility gives rise to an equality constraint on $\mathcal{C}_{ddr}$ that links the polar angle corresponding to the robot position, the orientation of the robot and the sensor angle ϕ. These constraints are written in Equation 2.

$$\theta = \arctan(\frac{y}{x}) - \phi + (2k+1)\pi,\ k \in \mathbb{Z}. \tag{2}$$

- The sensor limits can be translated into simple inequalities on the sensor angle, ϕ, by defining bounds (ϕ^-, ϕ^+) such that,

$$\phi^- \leq \phi \leq \phi^+.$$

- Lastly, the polygonal obstacles in the environment generate inequality constraints in $\mathcal{C}_{ddr}$ of two kinds: By causing direct collision with the robot, or by generating shadows for the landmark visibility.

As it has been shown recently, it is possible (1) in the absence of obstacles, to give an analytical expression for shortest length trajectories while handling the first three constraints (nonholonomy, landmark visibility and sensor angular limits), and (2) in the presence of obstacles, to plan collision-free trajectories based on the optimal ones without obstacles, through a recursive, complete scheme. The following paragraphs partly recall the results from recent works in the area [2,9,22] and extend them in several ways.

4.2. *Synthesis of locally optimal paths*

In the absence of obstacles, the optimal trajectories for such a system under the first three constraints above (nonholonomy, landmark visibility, and sensor limits) have been shown to be sets of at most four pieces of segments of logarithmic spirals (referred to as "S"), segments of straight lines (referred to as "L") and in-site rotations (referred to as "R") [2,22]. Logarithmic spirals are those curves that maintain the sensor angle ϕ to a saturated value, so that there are two possible spirals at one point, S^+ (in which $\phi = \phi^+$) and S^- (in which $\phi = \phi^-$). In open spaces, these paths are optimal in terms of Euclidean distance, i.e. they minimize

$$\mathcal{P}^* = \min_{\mathcal{P}} \mathcal{C}(\mathcal{P}), \tag{3}$$

where $\mathcal{C}(\mathcal{P}) = \int_{\mathbf{P}_i}^{\mathbf{P}_\mathbf{f}} ds$ and s is the curvilinear abscissa along $\mathcal{P}$.

A full synthesis of these optimal paths has been described in a recent work [22]. For any pair of initial and final configurations $\mathbf{P}_i$ and $\mathbf{P}_f$, it gives the sequence of primitives to execute. We will partly rely on it for the subsequent steps of our algorithm. This synthesis is illustrated by Fig. 2 (left). It depicts, for one starting position and one landmark position, the partition of the plane according to the nature of the shortest path to be done to reach the final point, which can be chosen

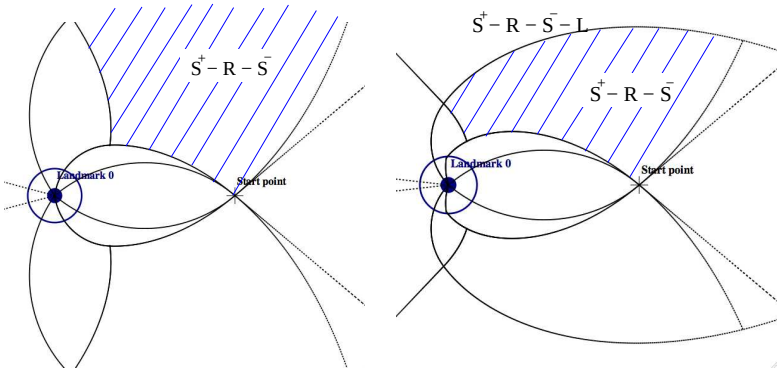


Fig. 2. Shortest path synthesis for DDRs maintaining a landmark in sight, for $Q = 1$ (left) and $Q \approx 2.3$ (right). Each of the region delimited by black curves corresponds to a different kind of shortest trajectory, made of pieces of logarithmic spirals S^+ or S^-, straight lines L and in-site rotation R. The criterion which is optimized here is the one of Eq. 4. Note that the regions where a spiral has to be done backwards (e.g. the one above the starting point, $S^+ - R - S^-$) are reduced in favor of the ones where the backwards part is done in straight line ($S^+ - R - S^- - L$).

anywhere in the plane. An example of a path obtained by this synthesis is given as the red path (light grey) in the upper part of Fig. 3, which is made of four non-zero length primitives.

One of the most noticeable characteristic of the paths computed as described above is the juxtaposition of segments of trajectories that move forwards and segments that move backwards. However, for humanoid robots, *backward motion should be minimized* when possible, as backward motion is generally more insecure, since no sensor feedback allow to detect obstacles, for example. We describe hereafter a first way to penalize backward motion, by modifying the criterion $\mathcal{C}$ from Equation 3.

The new criterion $\mathcal{C}'(\mathcal{P})$ is defined over the set of possible paths $\mathcal{P}$ starting at $\mathbf{P_i}$ and ending at $\mathbf{P_f}$, as follows,

$$\mathcal{C}'(\mathcal{P}) = \int_{\mathbf{P}_i}^{\mathbf{P_f}} q(s)ds, \tag{4}$$

where

$$q(s) = \begin{cases} 1 & \text{if } \dot{x}(s)\cos\theta(s) + \dot{y}(s)\sin\theta(s) > 0, \\ Q & \text{if } \dot{x}(s)\cos\theta(s) + \dot{y}(s)\sin\theta(s) < 0, \end{cases}$$

and where $Q > 1$ is a constant term penalizing backward motion, that acts as a parameter of the algorithm.

The extension of the optimality results to the criterion $\mathcal{C}'(\mathcal{P})$ of Eq. 4 instead of the Euclidean distance is straightforward, as the factor Q only affects: (1) the spatial distribution of the nature of optimal curves and (2) the parameters of the curves involving both backward and forward motion. We spare the algebra to the

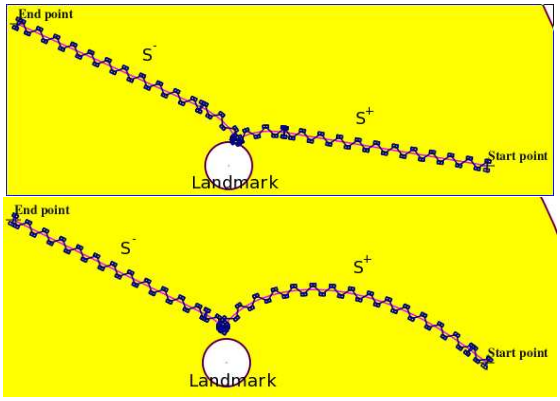


Fig. 3. Effect of the penalizing term Q on the motion primitives for the synthesis of shortest paths. In both cases, the same request is entered, for $Q = 1$ above, $Q > 1$ below. Both of the optimal trajectories are of the type $L-S^+-R-S^--L$, but in the case below, the backward part tends to reduce to the straight line part. In dark blue, footprints generated from the computed trajectory.

reader, and give the resulting plane partition (i.e. the path synthesis), for some $Q > 1$, in Fig. 2, right. As an example, the stripped region on the figure is the one where $S^+ - R - S^-$ trajectories are the shortest. On the right, the stripped region is reduced in favor of the $S^+ - R - S^- - L$ region, i.e. the backward spiral tend to be replaced by a smaller part under $\mathcal{C}'(\mathcal{P})$.

To further illustrate the effect of Q, we give in Fig. 3 two examples of optimal trajectories, without obstacles, for the same pair $(\mathbf{P}_i, \mathbf{P}_f)$. The upper one is the $L-S^+-R-S^--L$ trajectory obtained for $Q = 1$, the lower one is the trajectory *of the same type* obtained for $Q > 1$. Note that the first part of the trajectory (straight line, then spiral S^+) is done *forwards* whereas the second part (spiral S^-, then straight line) is done *backwards*. One can observe that the effect of Q is to reduce the backward part to a simple straight line.

4.3. *Global path planner*

In this part, we use the results that we stated above (that hold in the absence of obstacles) to propose a planning algorithm that handles obstacles (Section 4.3.1). Then, we generalize this algorithm to handle several landmarks (Section 4.3.2).

4.3.1. *Planning among obstacles*

In a recent work [9], we have proposed a planning strategy to generate collision-free paths among obstacles and with the same constraints as before, i.e. that the landmark has to be kept in sight. The algorithm is described in the next few paragraphs. Steps 5 and 6 have been added in another recent work [7].

The idea is to use a recursive scheme proposed in a previous work on car-like systems [14] and adapt it in order to cope with the landmark visibility constraints.

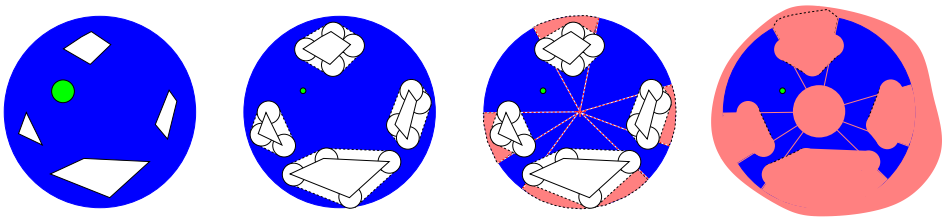


Fig. 4. Construction of $\mathcal{C}_{ddr}^{free}$ in the (x, y, ϕ) space. By dilating physical obstacles in the xy plane to define collision obstacles (white), the circular robot can be reduced to a point (green). Shadows and visibility constraints (Equation 2) define visibility obstacles (3rd from left).

Indeed, one of the effects induced by the presence of the landmarks is that they may generate *shadows* where the visibility is broken. Hence, to get the free space $\mathcal{C}_{ddr}^{free}$, not only the obstacles dilated by the robot shape (that we will suppose circular) have to be removed from the configuration space $\mathcal{C}_{ddr}$ but also the shadows generated by them. Such an explicit building of $\mathcal{C}_{ddr}^{free}$ is illustrated in Fig. 4, and it is the first step of our algorithm. In the image to the right of Fig. 4 show in blue $\mathcal{C}_{ddr}^{free}$ including the dilated obstacles, the shadows from the visibility computation from the landmark and the range limitations of the robot sensor. It should be noted that if the landmark lies on top of an obstacle, that portion of $\mathcal{C}_{ddr}^{obst}$, the obstacle configuration space, would be the union of the dilated zone from the obstacle and the range limits of the landmark.

Then, by computing a roadmap on this representation of $\mathcal{C}_{ddr}^{free}$, we can easily obtain a feasible path for an equivalent point-like *holonomic* system. Steps 1 and 2 of Algorithm 1 hereafter reach this goal and capture the connectivity of the free space $\mathcal{C}_{ddr}^{free}$ with a *Generalized Voronoi Graph* (GVG) $\mathcal{G}$ for a circular robot *without considering its orientation* (i.e., in $\mathbb{R}^2$). Physical obstacles are dilated and merged to the aforementioned shadows. Then (in Step 4 of the algorithm), we try to connect the initial and final configurations $\mathbf{P}_i$ and $\mathbf{P}_f$ using the optimal primitives given by the synthesis. If a collision is found in the newly generated path, it is split recursively into pieces replacing each piece with a shortest path between the previously initial or final configurations and the split point. It should be noted that at this stage the generated collision-free path is not optimal in distance but each piece is locally optimal. As it can be suspected, the algorithm is most efficient when the clearance from the obstacles in the original roadmap is maximized [19,14] as the number of subdivisions will be less. The GVG gives the roadmap that maximizes this clearance. It can also be proven [7] that this subdivision procedure will stop at a certain length of the subpaths, which is function of the distance from the robot to the closest obstacle.

Step 4 of Algorithm 1 produces feasible collision-free paths but that are likely to have useless detours and maneuvers (shortest-length paths in the presence of obstacles are going to be in contact with the obstacles and therefore not collision-free

and insecure for a humanoid robot to execute). An efficient and very simple strategy to implement for path optimization is to take two random points from the feasible path of Step 4 and try to replace the subpath inbetween the two configurations with a shortest-length primitive from the synthesis. If the subpath is collision-free and if it is not of the same length as the previous path, it will be necessarily of a shorter length and can therefore replace the original segment. This procedure is performed until, after a given number of tries no collision-free subpath path, or a length-reducing subpath is found. This optimization strategy does not compromise the completeness of the overall algorithm.

Algorithm 1 Path planning of the DDR reduced model, one landmark

Require: Initial and final configurations $\mathbf{P}_i$ and $\mathbf{P}_f$. Landmark position $\mathbf{L}$.
Ensure: Footprints along a collision-free path $\mathcal{P}$ maintaining the landmark in sight.

1: Build an explicit representation of $\mathcal{C}_{ddr}^{free}$ as the difference of $\mathcal{C}_{ddr}$ and the union of the dilated obstacles with the shadows induced by the landmark visibility;
2: Build the GVG $\mathcal{G}$ on $\mathcal{C}_{ddr}^{free}$; $\mathcal{C}_{ddr}^{free}$ being made of parts of lines or circles, $\mathcal{G}$ is made of parts of lines, parabolas or hyperbolas;
3: Given a starting and a goal configurations $(\mathbf{P}_i, \mathbf{P}_f)$, compute a path $\hat{\mathcal{P}}$ for the *holonomic* system associated to the robot by connecting these locations to $\mathcal{G}$; if not possible, *no nonholonomic path can be found as well*;
4: Recursively connect the starting and ending points with the optimal primitives; if the computed sub-paths are in collision, use the point at middle-path in $\hat{\mathcal{P}}$ as a sub-goal and re-apply the recursive procedure to the two resulting sub-paths;
5: *Optimize* the final path:
6: **for** $i = 1$ to n_i **do**
7: Generate n_s randomly possible shortcuts between the constitutive primitives of the trajectory and apply the one that improves $\mathcal{C}(\mathcal{P})$;
8: **end for**
9: Transform the resulting path *into a set of footprints* for the humanoid robot, i.e., into sets of left and right feet positions (see section 4.4).

This algorithm converges and is complete for the DDR reduced model, in the sense that it will give a trajectory if and only if there is an existing one. Details on the implementation can be found in our previous work [9], in particular the way the GVG edge weights can be set in such manner that either the shortest or the safest paths can be preferred.

A contribution to this algorithm, relative to our previous works [9,7] is that in order to further penalize backward motion, we set weights in the edges $\mathcal{E}$ of the GVG $\mathcal{G}$ that depend (1) on the *length* of the arcs corresponding to this edge, (2) on the *clearance* along the arc and (3) on the proportion of the path that would have to be done backwards between the two endpoints of the arc, if no obstacles were present. The combination of these factors is done in a heuristic way as :

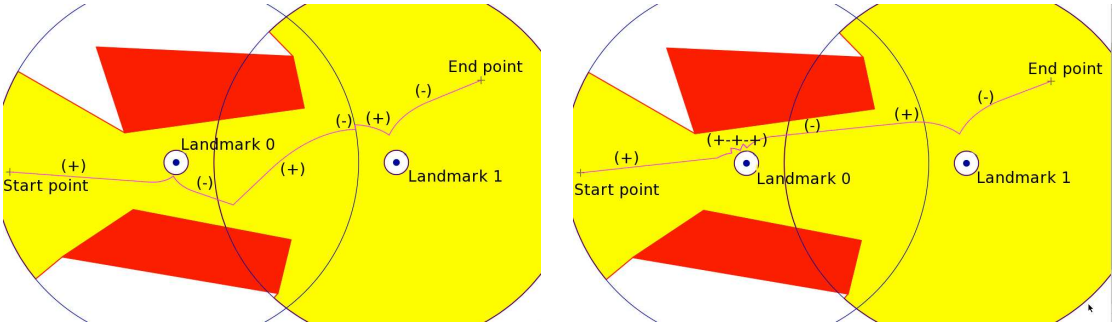


Fig. 5. Including the proportion of forward motion into the graph weights : on the left, the answer to a query that does not include this criterion; on the right, the answer to the same query with the criterion taken into account. In both cases, each primitive is labelled with an (+) for forward motions, or with an (-) for backward motion. Overall, the proportion of the path being done backwards is reduced.

$$w(\mathcal{E}) = \min(\mu, f(\mathcal{E}))\frac{l(\mathcal{E})}{1+\gamma c(\mathcal{E})},$$

where $l(\mathcal{E})$ is the length of the edge, $c(\mathcal{E})$ the minimal clearance along $\mathcal{E}$, and $f(\mathcal{E}) \in [0,1]$ is the ratio of the path along the edge that is done backwards. The constants $\mu > 0$ and γ are user-defined and allow to emphasize one factor over the others. A comparison between two paths, with and without the term in $f(\mathcal{E})$ in the graph weights is shown in Fig. 5.

Handling moving obstacles Humanoid robots are expected to work on human populated environments. If a moving obstacle, such as a person traversing the robot working area, is found while the robot executes its path two different strategies can be followed:

- If the moving obstacle is blocking the computed trajectory, the robot can stop or reduce its velocity depending on the distance to the obstacle and wait for the obstacle to pass.
- If the moving obstacle is not blocking the robot path but occluding the landmark, the robot continues performing its trajectory keeping an estimate of the landmark position. When the obstacle passes and the landmark is seen again, this estimate is corrected.

If the moving obstacle remains blocking the path or occluding the landmark for a given time interval, it is included to the robot map and a new path is computed.

These strategies are not currently integrated into our planning algorithm but they will be implemented as future work.

4.3.2. *Handling several landmarks*

Our next improvement upon the preliminary version of this work [8] has been the generalization of the previously mentioned method to the case of several landmarks, which is more extensively discussed in [7]. This has been done in such a way that, along the generated path, at least one landmark remains visible at all times. Because of this, navigation is facilitated in the sense that, as the computed paths are followed, we can be sure in advance that landmarks can be relied on for visual localization. The implemented idea, that is detailed in a recently submitted paper [7], is to generate a set of different roadmaps as shown above, for each landmark $\mathbf{L}_k$, and in which the same process as in Algorithm 1 is applied. Now, the difference is that for each pair of landmarks $(\mathbf{L}_k, \mathbf{L}_l)$, we also compute geometrically the sub-set of $\mathbb{R}^2$ where a switch of the gaze from landmark $\mathbf{L}_k$ to $\mathbf{L}_l$ is safe, i.e. where both are visible and none of them is lost while switching from one to the other. A set of nodes is extracted from the GVGs of these regions that allows the different individual roadmaps to be connected. As a result, we obtain a forest covering the union of the visibility areas for each landmark, and we use it in the same way as described above to generate collision-free paths. The algorithm is summed up in Algorithm 2.

An example of the steps and results of this extended algorithm is depicted on Fig. 6. First, in Fig. 6(a), the polyhedral environment with its obstacles and a set of 3 landmarks visibility regions covering it are shown. Then, the union of the visibility areas is shown in Fig. 6(b), and the graph resulting from the union of individual roadmaps, as described above, is shown in Fig. 6(c). Connector nodes, that connect individual roadmaps, are easily recognizable. Last, Fig. 6(d) shows the computed path resulting for a particular query and the footprint generation, explained hereafter, computed on it.

Algorithm 2 Path planning of the DDR reduced model, several landmarks.

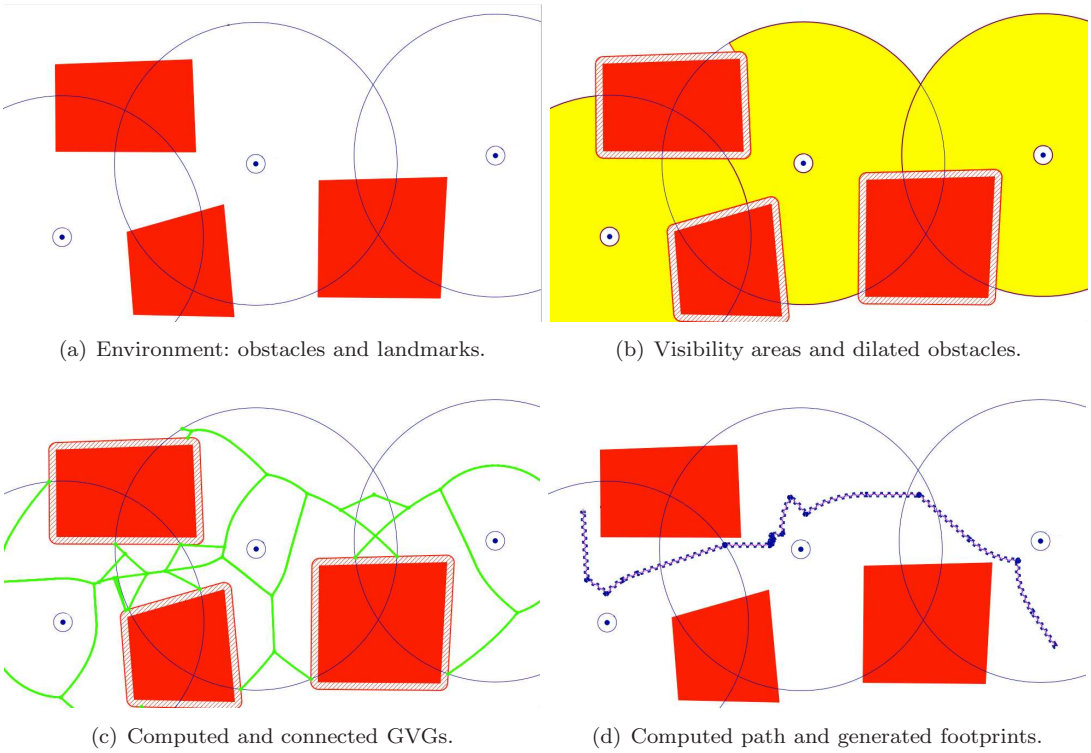
Require: Initial and final configurations $\mathbf{P}_i$ and $\mathbf{P}_f$, a set of landmarks $\mathbf{L}_k$.

Ensure: Footprints along a collision-free path $\mathcal{P}$ maintaining *at least* one landmark in sight and the sequence of corresponding landmarks to observe.

1: Build $\mathcal{C}^{free}_{ddr,i}$ and a roadmap (GVG) $\mathcal{G}_i$ for each landmark i as described in Algorithm 1;

2: Determine geometrically the sets $\mathcal{S}_{kl}$ where both landmarks $\mathbf{L}_k$ and $\mathbf{L}_l$ are visible and where a switch from gazing at one from the other is possible. Connect the GVGs $\mathcal{G}_i$ through the nodes of the GVGs of $\mathcal{S}_{kl}$;

3: Apply the recursive algorithm of Algorithm 1, optimize the solution if it exists and generate footprints.



(a) Environment: obstacles and landmarks.

(b) Visibility areas and dilated obstacles.

(c) Computed and connected GVGs.

(d) Computed path and generated footprints.

Fig. 6. Computing path among obstacles observing at least one landmark among several ones. Landmarks are depicted as dark dots. The final nonholonomic path is made of non-trivial primitives (pieces of lines or spirals), and in-site rotations.

4.4. *Footprint extraction*

Once a path has been computed through Algorithm 2, the next step is to produce a set of footprints that will serve as input for the locomotion controller. We used a purely geometrical way to do so : given the specification of the humanoid robot (inter-feet distance and desired step length, in particular), we first set the feet aligned with the initial configuration, and then generate the different steps until a given distance from the end of the current part trajectory is attained (i.e., we stop the robot at non-differentiable points to perform in-site rotations).

5. Local whole-body motion generator

As a local motion planner, we have used the method proposed in [17]. To generate the whole-body motion of the humanoid robot, the first step is to obtain, from the footsteps extracted from the global planner, the desired stable trajectory of the center of mass (CoM) of the robot. This stable trajectory is used in a second step as the highest priority task inside the stack-of-tasks (SoT) formalism proposed in [17]

and that we use here to enforce the visibility and geometric constraints that we know to be feasible because of the global planner of Section 4. The local method is able to generate dynamically stable biped walking motion that always maintains the ZMP inside the support polygon formed by the foot (feet).

5.1. *Pattern Generator*

This method uses the locomotion pattern generator from [11] whose input is a set of footprints compliant with the robot stepping parameters, as in this case the footprints resulting from the global VDDR planner from Section 4. From these set of footsteps and the stepping period, the reference ZMP trajectory is derived and tracked using a preview control of a simplified 3D linear inverted pendulum. The real multi-body dynamics of the robot are considered within a second stage of the preview controller to obtain a more accurate trajectory of the CoM. The CoM reference trajectory is the output of the pattern generator.

5.2. *Stack-of-Tasks*

The local whole-body motion generator is a hierarchical inverse kinematics solver as proposed in [17]. Here, only two behaviors are involved in the local motion generation: locomotion and visual constraints enforcement. The main idea is to use the redundancy formalism [23,20] to compute a control law using a series of tasks, the one with the least priority projected into the null space of the one with one level above of priority and so on, until the highest level of priority is reached.

A kinematic task is represented by an error function $\mathbf{e}(\mathbf{q}) = \mathbf{x}_d - f(\mathbf{q})$ where $\mathbf{x}_d \in \mathbb{R}^m$ stands for the desired operational location in an m-dimensional space (i.e. position and orientation); $\mathbf{q} \in \mathbb{R}^n$ is the robot configuration in an n-dimensional configuration space and $\mathbf{x} = f(\mathbf{q})$ is either a point or location attached to the robot expressed in terms of its configuration. The exponential convergence of $\mathbf{e}(\mathbf{q})$ is achieved by means of $\dot{\mathbf{e}}(\mathbf{q}) = -\alpha\mathbf{e}(\mathbf{q})$ where the linear system of equalities:

$$J(\mathbf{q})\dot{\mathbf{q}} = -\alpha\mathbf{e}(\mathbf{q}) \tag{5}$$

is satisfied, with $J(\mathbf{q}) \in \mathbb{R}^{m\times n}$ being the Jacobian of the task. Thus, when $m < n$, a hierarchical structure of tasks under the form $\{\mathbf{e}_1(\mathbf{q})\ \mathbf{e}_2(\mathbf{q})\ \ldots\ \mathbf{e}_p(\mathbf{q})\}$ can be defined on the $(n-m)$-dimensional null-space. It contains p tasks ordered with decreasing priority, where to each task i is associated a Jacobian $J_i(\mathbf{q})$. The problem to be solved recursively for all $\mathbf{e}_i(\mathbf{q})$ is then formulated as:

$$\begin{aligned} &\min_{\dot{\mathbf{q}}_i\in\mathbb{R}^n,\mathbf{w}\in\mathbb{R}^m} \tfrac{1}{2}\parallel \mathbf{w} \parallel^2 + \tfrac{1}{2}\parallel k_i\dot{\mathbf{q}}_i \parallel^2, \\ &\text{s.t.} \qquad J_i(\mathbf{q})Q_{i-1}(\mathbf{q})\dot{\mathbf{q}}_i - (\dot{\mathbf{e}}_i(\mathbf{q}) - J_i(\mathbf{q})\dot{\mathbf{q}}_{i-1}) = \mathbf{w} \end{aligned} \tag{6}$$

where $\dot{\mathbf{q}}_i$ are the successive versions of the vector of velocities, k_i a scalar controlling the regularization on the vector of velocities (see after), the matrix $Q_l(\mathbf{q}) = (I - J_l^+(\mathbf{q})J_l(\mathbf{q})) \in \mathbb{R}^{n\times n}$ stands for the orthogonal projection of $J_l(\mathbf{q})$, and $J^+ = J^T(JJ^T)^{-1}$ is the Moore-Penrose inverse of matrix J. This means that $J(\mathbf{q})$ and $Q(\mathbf{q})$ are orthogonal complements. The solution of (6) is given by the following recursive computation of the articular velocity as proposed in [23]:

$$\begin{cases} \dot{\mathbf{q}}_0 = 0 \\ \dot{\mathbf{q}}_i = \dot{\mathbf{q}}_{i-1} + (\hat{J}_i(\mathbf{q})^{+_{k_i}}(\dot{\mathbf{e}}_i - J_i(\mathbf{q})\dot{\mathbf{q}}_{i-1}), \quad \text{for } i = 1 \dots p \end{cases} \tag{7}$$

where $\hat{J}^{+k} = J^T(JJ^T + k^2 I)^{-1}$ is a singularly robust pseudo-inversion of J with a factor k regulating this operation [20]. In our work, the tasks with the highest priorities are related to the locomotion behavior by specifying the position of both feet and the reference CoM trajectory to be reached. The task Jacobian of the CoM is computed as in [25]. As all of the DOFs of the humanoid lower body, including the underactuated DOFs are used for locomotion, only those of the upper body remain available for defining additional tasks, in this case the enforcement of visual constraints. Inside the visual enforcement task, three subtasks are included:

1) A *visual servoing* task, which goal is to minimize the error between the current value of the ith landmark (which position is given as input by Algorithm 2) and the desired value resulting from the VDDR planner.
2) A *centering* task to center the landmark in the image and stabilize the camera motion.
3) A *joint limit avoidance* task.

In Section 4, it is explained that the paths obtained with the VDDR model are paths where the visual sensor is saturated. When we apply this to the humanoid robot we profit from the upper body redundancy to avoid joint saturation by using the neck, chest and waist degrees of freedom for keeping the landmark in sight.

More details on how to compute each of these tasks can be found in [17].

6. Experimental results

In this section we present some results of applying our strategy, first on simulations, then through two scenarios with the humanoid robot HRP-2 from the French-Japanese Joint Robotics Laboratory at Tsukuba. In all the experiments below, the planning part following the algorithms presented in section 4.3 has been implemented in C++ with widespread software libraries. In particular, the CGAL library was used for the 2D planner from which we benefited in particular to implement Voronoi diagrams.

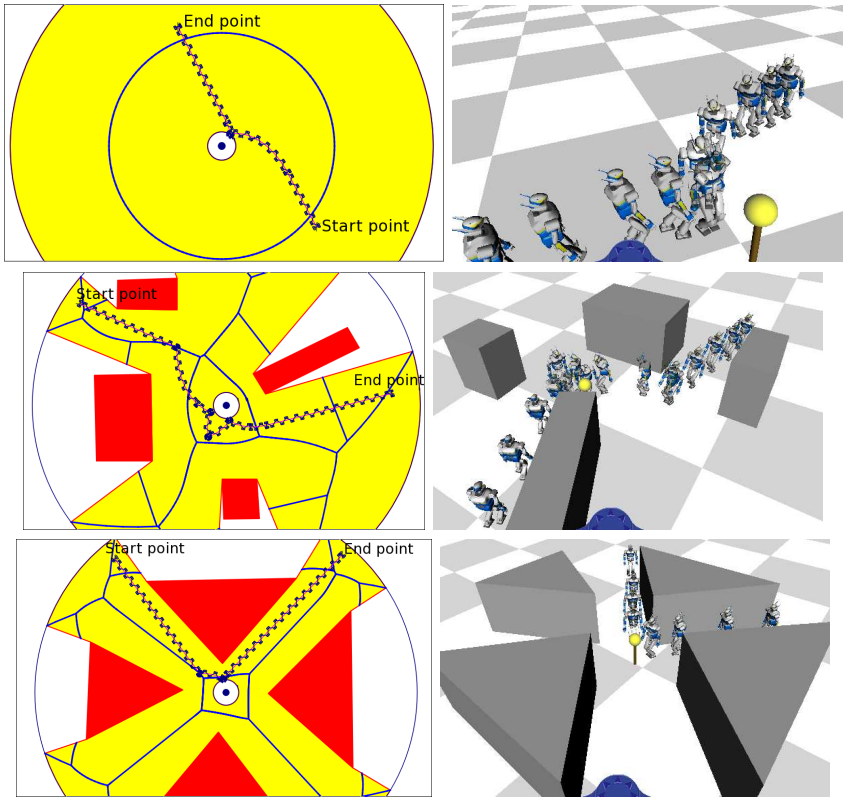


Fig. 7. Three examples of scenarios with (2nd and 3rd rows) or without (1st row) obstacles. The left column shows, in all cases, the trajectories computed by the 2D planner that allows for sensor restrictions to be met all along the trajectories. The spiral and line parts are quite distinguishable; the blue structures are the edges of the underlying GVG. The right column displays several configurations of the real trajectory by the humanoid. The landmark is represented by a yellow ball.

6.1. *Simulations*

In this series of simulations, we used the pattern generation, including dynamics simulation and motion controllers, from the Open-HRP platform [12], and our own implementation of inverse kinematics for orienting the robot towards the landmark.

In a first scenario (first row of Fig. 7), we computed a feasible path (which is optimal), with one landmark to maintain in sight and without obstacles using all the elements described in Section 4. On the left side, the computed 2D path is shown, with the generated footprints in dark blue and the Voronoi diagram in blue. On the right side, the trajectory made by the simulated robot is shown, with the landmark represented by the yellow sphere. Notice how the robot keeps the landmark in sight during the whole motion, which could be an important element in surveillance tasks, for example.

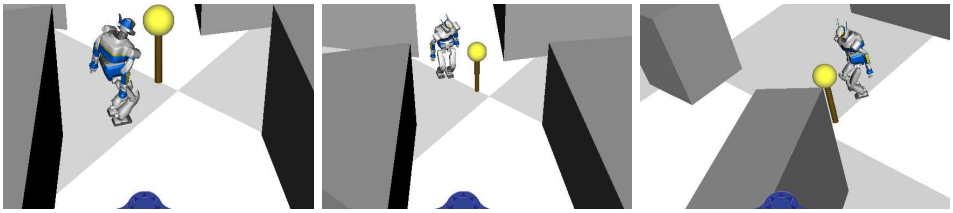


Fig. 8. Individual configurations of the HRP-2 robot extracted from the trajectories above. Note that the inverse kinematics tend to use many of the degrees of freedom of the body upper part to set the gaze onto the landmark, e.g. in the left frame.

On the second and third rows two other scenarios are depicted, with still one landmark to maintain in sight but this time including a set of polygonal obstacles which have to be avoided by the robot. Because of the algorithm design, the collision avoidance is done for a bounding cylinder around the robot, so that no 3D collision-avoidance has been explicitly done. As in the previous case, the 2D path image shows the footprints computed around the obstacles as well as the Voronoi diagram for each of the environments. Note how the final footprints path has some of its intermediary points on the Voronoi diagram, which correspond to the recursion steps in the algorithm 2 planning paths among obstacles. Again, in the right column, we show the simulated trajectory for the virtual humanoid robot. Fig. 8 shows a close-up of some configurations near the landmark with the robot gaze directed to it.

The one-landmark scenarios of Figures 7 and 8 have already been presented in [8]. Here, they are presented to have a complete panorama of the results obtained with our algorithms for one and several landmarks.

A second scenario, this time with two landmarks, is shown in Fig. 9. The top row on this Figure shows the initial and the final configurations given as input to compute the path. The global path is constructed recursively using Algorithm 2 in order to ensure the visibility of at least one landmark (yellow spheres) at all times. The second and third rows of Fig. 9 show two configurations near the first landmark (a close-up on the left and a view of the complete environment on the right) where the robot has to maneuver to go around the landmark without losing sight of it. These maneuvers could eventually be changed into a different walking pattern for the humanoid robot, i.e. it could be executed using side steps, a pattern which we did not explore on the present work. The fourth row in Fig. 9 shows the moment where the switching of landmarks is executed. These motion depends on the amplitude of the viewing field of the simulated robot. The figure on the left shows the moment when the robot is able to see both landmarks and the figure on the right shows when the robot has already switched from one landmark to the other. In the bottom row of Figure 9 (left) a configuration near the end of the path is shown and on the right an example configuration along the path using the robot

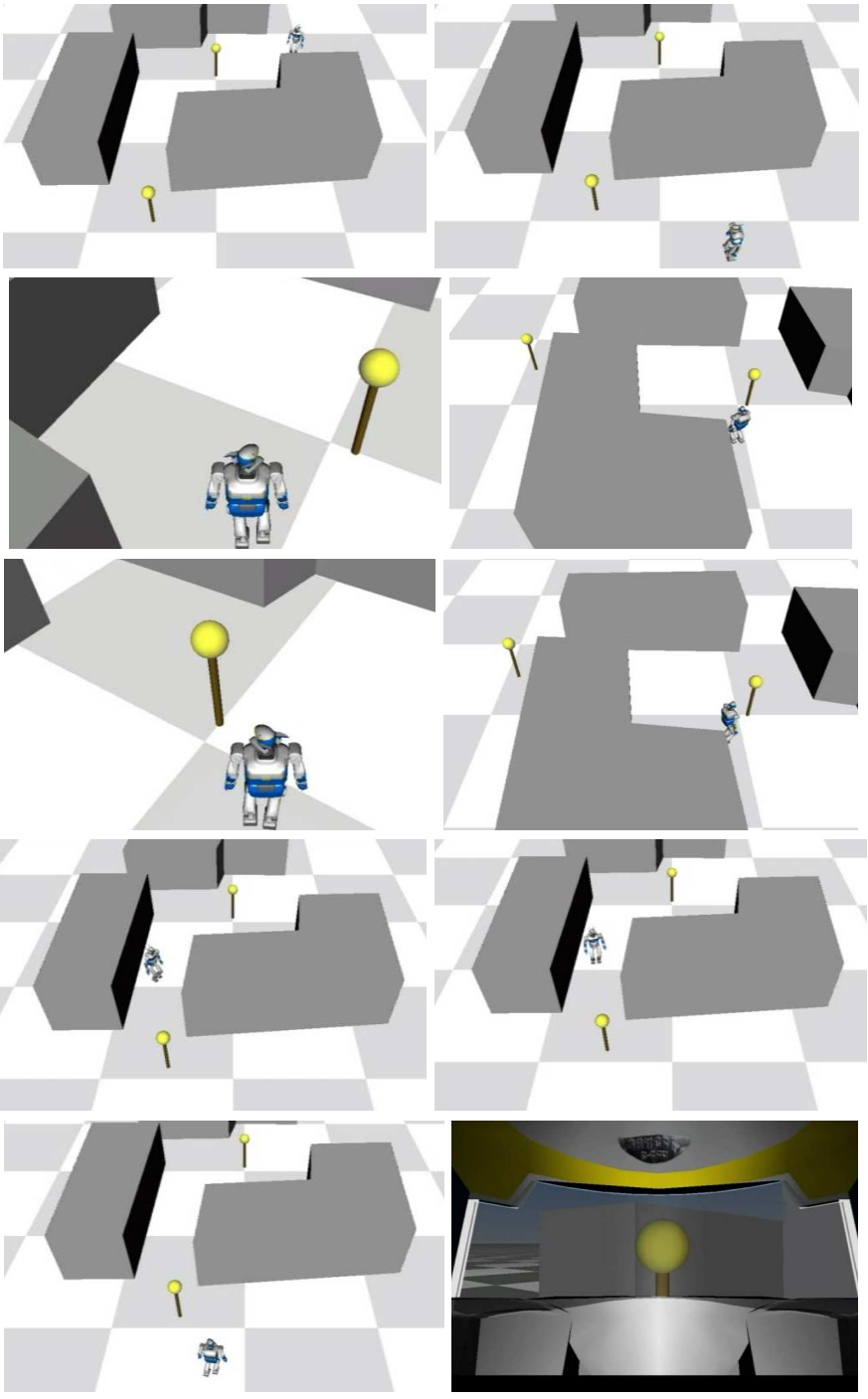


Fig. 9. Two-landmark scenario with obstacles. Top row: initial (left) and final (right) configurations. Second and third rows: The robot surrounds the landmark to execute the path without losing sight of it. Fourth row: The switching moment from viewing the first landmark to viewing the second one. Last row: on the left, a configuration near the end of the path; On the right, a view from the simulated robot camera to the landmark.

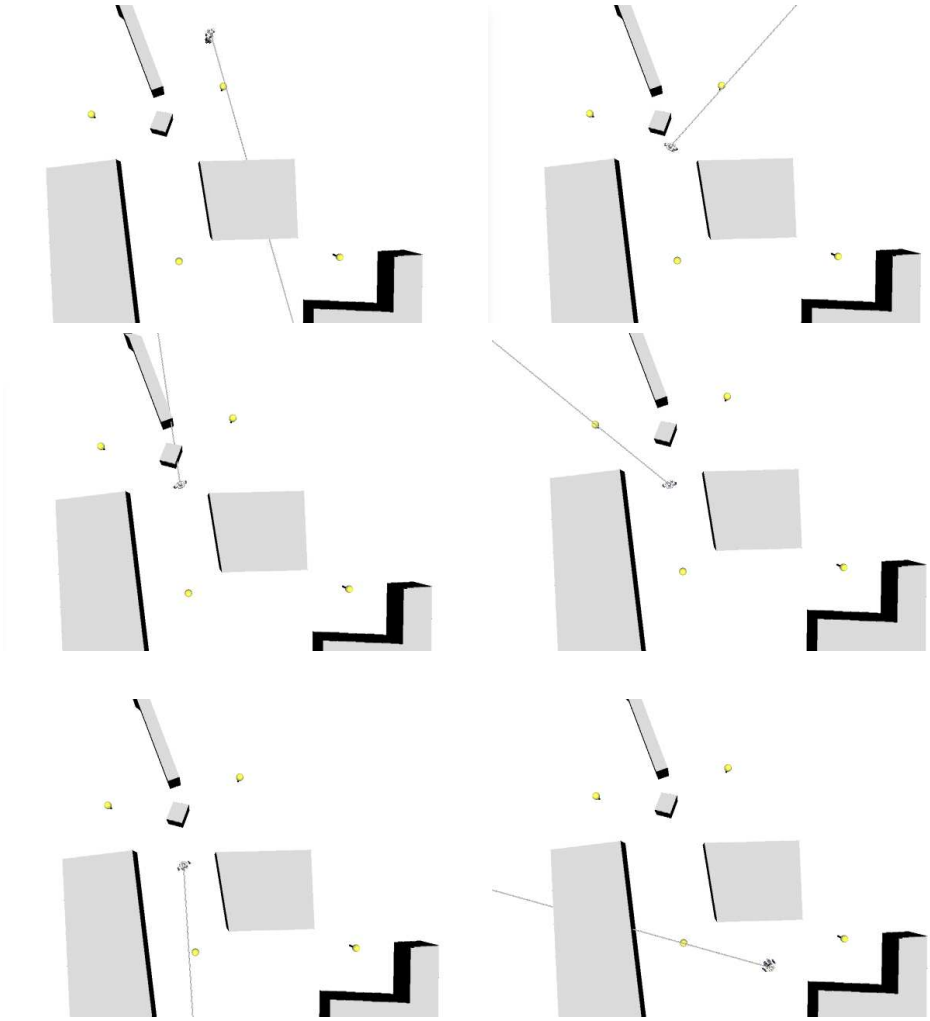


Fig. 10. Top view of a four-landmark scenario with obstacles. Some configurations of the execution of the path. The light ray indicates the gazing direction (the center of the field of view) of the robot to describe which landmark is seen at each shown configuration.

view is displayed.

The last simulated scenario, depicted in Fig. 10 describes, from left to right and from top to bottom, the execution of a 4-landmark trajectory with obstacles. The light ray on the figure describes the robot gazing direction, which is the center of the robot field of view. The first and last image show the initial and final configurations on the path. The left figure on the middle row shows the switching point between the first and the second landmark along the trajectory. Here the field of view is

large enough so that the robot can keen in sight the two upper landmarks at the same time. Three landmarks are seen by the robot during the execution of the path.

6.2. *Results with HRP-2*

We ported the planning algorithm on the HRP-2 platform, at the CNRS-AIST Joint French-Japanese Robotics Laboratory in Tsukuba. The tracking system used to enforce the visual constraints is part of the ViSP package [18].

In our first experiment, illustrated by Fig. 11, we have used a closet fixed on a wall as a unique visual landmark. The same code that we used for the simulations above was used here for generating safe 2D footprints path. Then, we used our pattern generator, described in section 5, to produce a path that ensures all the described constraints on the real robot.

At this stage, small disturbances on the execution of the path are not explicitly considered. Realization of the motion is done using the pattern generator from [11]. The online pattern generator includes an stabilization process to handle small disturbances when the motion is being executed.

Sample configurations during the execution of the path are shown in Fig. 11. The landmark is highlighted using red lines around it. The obstacles are the moving walls placed around the environment. In this one-landmark scenario, the robot executes the path by first approaching the obstacle to avoid the obstacles and then, making backward steps to arrive to the final configuration, placed on the other side of the wall. Note that a shortest path might have been found by walking on the other side of the wall but the landmark would have been occluded by the obstacle making a surveillance or a localization task much more difficult.

A second experiment was performed using a two-landmark scenario (Fig. 12). Here, the same closet as before is used as the first landmark and an LCD screen with a pink solid image showing on it is used as the second landmark. Both of these landmarks are shown inside red rectangles in Fig. 12(a). The first landmark can be seen for a large portion of the computed trajectory (see Fig. 12(b)) just to the part when the obstacle in the middle of the environment is passed (Fig. 12(c)-(f)). After this part of the environment, the first landmark is occluded and the robot is forced to switch obstacles and look at the LCD screen (Fig. 12(h)-(k)). Fig. 12 (l) to (n) are the landmarks from the robot viewpoint. The first image is a configuration where the robot is looking at the first landmark, the second one is a configuration when the robot is switching between the first and the second landmarks and therefore both landmarks are in sight and the third image is a configuration is when the second landmark is in sight.

The major issue for the implementation of both of these experiments lies on the sensitivity to the vision processes. For instance in the first part of the video, the robot is visual servoing on a model map on the environment. If the robot lose track

of the object, because the visibility has been violated, the robot has to stop find back the object which in some case, may imply human intervention.

Videos for both of these experiments can be found on `http://www.cimat.mx/~cesteves/VigilantHumanoid/`

7. Conclusions and Future work

We presented an algorithm that allows planning motion for humanoid robots with sensory constraints that are imposed by the presence of one or several landmarks among which at least one must be visible during the whole trajectory. As opposed to most existing approaches, we incorporate these constraints not a posteriori but at the planning level, with a reduced model of the robot (a differential drive robot), a model to which recent studies within the Neuroscience community has given some support. Once a plan has been found for this reduced model, we compute a series of footprints that feed a stack of tasks to generate the humanoid whole-body walking pattern. Chief among the tasks in the stack are the maintenance of the robot stability and a visual servoing task to satisfy all the visibility constraints.

Although our approach generates backward motion that could be considered as unsafe for the robot, we claim that in specific surveillance or localization applications, the algorithm is helpful, as the problem of robot safety can be managed with range sensors. Our approach not only has been tested on simulation examples for the HRP-2 humanoid robot, but we have also implemented it onboard on the real HRP-2 platform.

Among ongoing and future work, we aim at incorporating more motion primitives to the motion planner for the reduced model. In particular, lateral motions are currently not supported in our scheme but could be helpful in many situations. Approaches such as the one presented in [26] might be a reasonable solution in our case to include these lateral motions.

Another ongoing work is to consider the landmarks not as one point but as several interest points in an image to consider landmarks as segments of lines or planar patches. This would allow us to perform a visually-based control when performing the trajectory, which would be more robust than a position-based trajectory tracking.

References

1. G. Arechavaleta, J-P. Laumond, H. Hicheur, and A. Berthoz. On the nonholonomic nature of human locomotion. *Autonomous Robots*, 25, 2008.
2. S. Bhattacharya, R. Murrieta-Cid, and S. Hutchinson. Optimal paths for landmark-based navigation by differential-drive vehicles with field-of-view constraints. *IEEE Trans. on Robotics*, 23(1):47–59, 2007.
3. J. Chestnutt and J.J. Kuffner. A tiered planning strategy for biped navigation. In *Proc. of the IEEE-RAS Intl. Conf. on Humanoid Robots.*, 2004.
4. J. Chestnutt, J.J. Kuffner, K. Nishiwaki, and S. Kagami. Planning biped navigation

strategies in complex environments. In *Proc. of the IEEE-RAS Intl. Conf. on Humanoid Robots.*, 2003.
5. Adrien Escande and Abderrahmane Kheddar. Contact planning for acyclic motion with task constraints and experiment on hrp-2 humanoid. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 435 – 440, 2009.
6. K. Hauser, T. Bretl, and J-C. Latombe. Non-gaited humanoid locomotion planning. In *Proc. of the IEEE-RAS Intl. Conf. on Humanoid Robots*, pages 7–12, 2005.
7. J.-B. Hayet, C. Esteves, and R. Murrieta-Cid. Motion planning for maintaining landmarks visibility for a differential drive robot, 2010. Submitted to Robotics and Autonomous Systems.
8. J.B. Hayet, C. Esteves, G. Arechavaleta, and E. Yoshida. Motion planning for a vigilant humanoid robot. In *Proc. of the IEEE-RAS Intl. Conf. on Humanoid Robots.*, pages 196–201, 2009.
9. J.B. Hayet, C. Esteves, and R. Murrieta-Cid. A motion planner for maintaining landmark visibility with a differential drive robot. In Springer, editor, *Proc. of the Intl. Workshop on the Algorithmic Foundations of Robotics*, pages 333–347, 2008.
10. H. Hicheur, Q-C. Pham, G. Arechavaleta, J-P. Laumond, and A. Berthoz. The formation of trajectories during goal-oriented locomotion in humans i: A stereotyped behaviour. *European Journal of Neuroscience*, 26(8):2376–2390, 2007.
11. 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 *Proc. of the IEEE Intl. Conf. on Robotics and Automation*, pages 1620–1626, 2003.
12. F. Kanehiro, H. Hirukawa, and S. Kajita. OpenHRP: Open architecture humanoid robotics platform. *Intl. Journal of Robotics Research*, 23(2):155–165, 2004.
13. O. Kanoun, F. Lamiraux, P-B. Wieber, F. Kanehiro, E. Yoshida, and J-P. Laumond. Prioritizing linear equality and inequality systems: Application to local motion planning for redundant robots. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation*, pages 2939–2944, 2009.
14. J.-P. Laumond, P.E. Jacobs, M. Taïx, and R.M. Murray. A motion planner for nonholonomic mobile robots. *IEEE Trans. on Robotics and Automation*, 10(5):577–593, 1994.
15. N. Mansard and F. Chaumette. Task sequencing for high level sensor-based control. *IEEE Trans. on Robotics*, 23(1):60–72, 2007.
16. N. Mansard, O. Khatib, and A. Kheddar. A unified approach to integrate unilateral constraints in the stack of tasks. *IEEE Trans. on Robotics*, 25(3):670–685, 2009.
17. N. Mansard, O. Stasse, F. Chaumette, and K. Yokoi. Visually-guided grasping while walking on a humanoid robot. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation*, pages 3041–3047, 2007.
18. E. Marchand, F. Spindler, and F. Chaumette. Visp for visual servoing: A generic software platform with a wide class of robot control skills. *IEEE Robotics and Automation Magazine, Special Issue on Software Packages for Vision-Based Control of Motion*, 12(4):40–52, December 2005.
19. B. Mirtich and J. Canny. Using skeletons for nonholonomic path planning among obstacles. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation*, pages 2533–2540, 1992.
20. Y. Nakamura. *Advanced Robotics: Redundancy and Optimization*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, lst edition, 1990.
21. K. Nishiwaki, J.J. Kuffner, S. Kagami, M. Inaba, and H. Inoue. The experimental humanoid robot H7: A research platform for autonomous behaviour. *Philosophical*

Trans. of the Royal Society, 365:79–107, 2007.
22. P. Salaris, D. Fontanelli, L. Pallottino, and A. Bicchi. Shortest paths for a robot with nonholonomic and field-of-view constraints. *IEEE Trans. on Robotics*, 26(2):269–281, 2010.
23. B. Siciliano and J-J. Slotine. A general framework for managing multiple tasks in highly redundant robotic systems. In *Proc. of the IEEE Intl. Conf. on Advanced Robotics*, pages 1211–1216, 1991.
24. O. Stasse, F. Saïdi, K. Yokoi, B. Verrelst, B. Vanderborght, A. Davison, N. Mansard, and C. Esteves. Integrating walking and vision to increase humanoid autonomy. *Intl. Journal of Humanoid Robotics*, 5(2):287–310, 2008.
25. T. Sugihara, Y. Nakamura, and H. Inoue. Real time humanoid motion generation through zmp manipulation based on inverted pendulum control. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation*, pages 1404–1409, 2002.
26. T-V-A. Truong, D. Flavigne, J. Pettré, K. Mombaur, and J-P. Laumond. Reactive synthesizing of human locomotion combining nonholonomic and holonomic behaviors. In *Proc. of the IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics*, pages 632 – 637, 2010.
27. Y. Yang and O. Brock. Elastic roadmaps: Globally task-consistent motion for autonomous mobile manipulation in dynamic environments. In *Proc. of Robotics: Science and Systems*, 2006.
28. E. Yoshida, C. Esteves, I. Belousov, J-P. Laumond, T. Sakaguchi, and K. Yokoi. Planning 3d collision-free dynamic robotic motion through iterative reshaping. *IEEE Trans. on Robotics*, 24(5):1186–1198, 2008.
29. E. Yoshida, J-P. Laumond, C. Esteves, O. Kanoun, A. Mallet, T. Sakaguchi, and K. Yokoi. Motion autonomy for humanoids: Experiments on HRP-2 no. 14. *Computer Animation and Virtual Worlds*, 20(5-6):511–522, 2009.
30. E. Yoshida, M. Poirier, J-P. Laumond, O. Kanoun, F. Lamiraux, R. Alami, and K. Yokoi. Pivoting based manipulation by a humanoid robot. *Autonomous Robots*, 28(1):77–88, January 2010.

Fig. 11. The execution of a trajectory with obstacles on the HRP-2 robot at four time-stamps, from two different points of view.

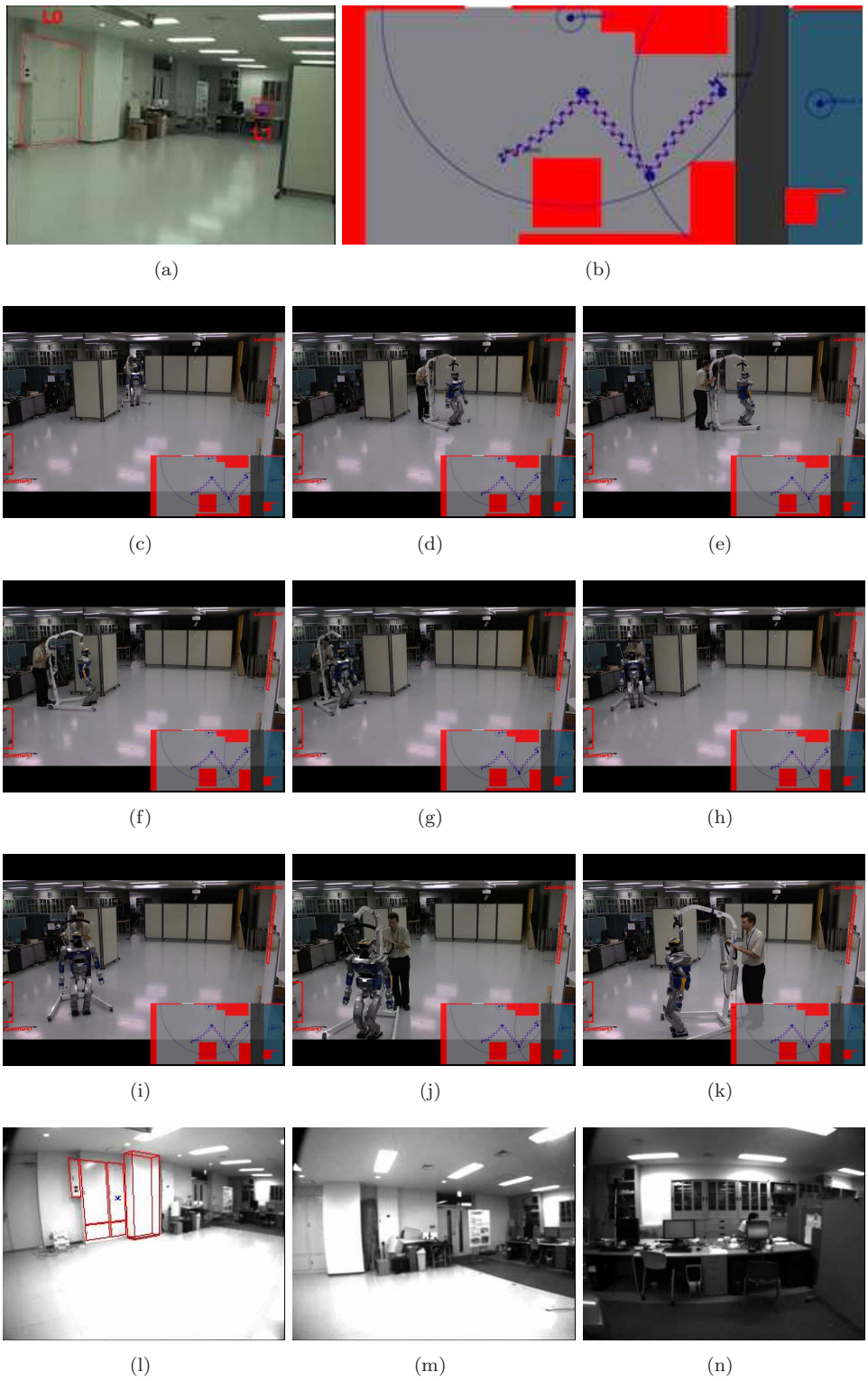


Fig. 12. Two-landmark trajectory execution. (a) The landmarks are shown in the image. One is a white closet on the wall and the other is an LCD screen with a pink image. (b) The computed trajectory for two landmarks and a big obstacle in the middle of the room. The rest of the obstacles are on the edges of the room. (c) to (k) The execution of the trajectory by HRP-2 robot. (g) One of the switching configurations. (l) View from to robot to the first landmark. (m) View from the robot switching landmarks, at this point both landmarks are in sight. (n) View from the robot to the second landmark, which is the LCD screen from the left.