Vision-based motion primitives for reactive walking

Mauricio Garcia, Olivier Stasse, Jean-Bernard Hayet, Claudia Esteves, Jean-Paul Laumond

Vision-based motion primitives for reactive walking

Mauricio Garcia[1,2] and Olivier Stasse[2] and Jean-Bernard Hayet[1] and Claudia Esteves[3] and Jean-Paul Laumond[2]

***Abstract*— This paper presents a method for reactive walking allowing visual servoing and adaptation of footsteps trajectories in real-time. This is done by building upon recent advances in the fields of optimal control for a walking pattern generator [1] and planning for a nonholomic robot with field-of-view constraints [2]. Herdt et al. [1] provided a controller where a humanoid robot is driven by its Center-of-Mass (CoM) velocity. A natural extension, proposed in [3], is to control a humanoid robot directly by using vision-based control techniques. However, the trajectories realized by the robot in that case are generated to minimize the distance in the image feature space and might create unnecessary motion in the space of the footprints. This paper aims at solving this problem by making the CoM follow a convenient space of trajectories for which the robot behaves overall better. The motion primitives obtained in [2] provide the space of trajectories used in this work.**

I. Introduction

A. Motivation and related work

In the last few decades, significant advances have been made on the real-time generation of stable locomotion sequences for humanoid robots (e.g. [1], [4]). Parallely, research has been led towards planning trajectories that are appropriate for humanoid robots walking in cluttered environments, with different types of restrictions (e.g. [5], [6]). From the seminal work of [5], a main trend of research in footstep planning has been to consider a limited set of known actions (quite often footsteps) and transitions, and to find an optimal path over them. The use of a fixed-set-of-actions approach can be limiting, as it may produce unnecessary motions near the obstacles [7], while not usually dealing with the problem of robust perturbation rejection.

A vast amount of work exist to lessen the amount of movements generated around the obstacles. Chestnutt et al. proposed in [8] an adaptation mechanism to search around the set of transitions. More recently, Hornung et al. [9], proposed a method to deal with highly dynamical environments by keeping both, accurate short-term goals and rough long-term goals. As doing this may lead to local optima, the author propose a method to automatically adapt the set of actions according to the environment traversability characteristics.

Regarding the problem of robust perturbation rejection, several advances have also been done. They can be divided into three strategies: (1) ankle-foot stabilization, (2) whole-body stabilization, and (3) footstep generation. Using the capture-point or the Center-of-Pressure (CoP) as an indicator of stability, one can switch between a Finite-State-Machine strategy [10] and a learned strategy [11]. A recent methodology proposed by Herdt et al. [1] is to automatically find the CoM and CoP trajectories and footstep positions simultaneously. This result is fed to a whole-body controller to deal coherently with the three cases. Here, the notion of footsteps disappears, allowing the user to provide a reference velocity as input to the pattern generator. Moreover, because the range of footsteps is explored by a guided search in the space of the whole-body controller transitions, a large set of possible footsteps is available in real-time.

From an application perspective, however, it is very difficult to decouple planning and control from each other. Planning is needed to avoid local optima, and control is needed to reject disturbances and adapt to modeling errors. Some planning methods try to account for the motion and control capabilities of the humanoid robot by using inverse kinematics [12]. Despite real-time implementation [13], this method suffers from local minima in planning footsteps. As

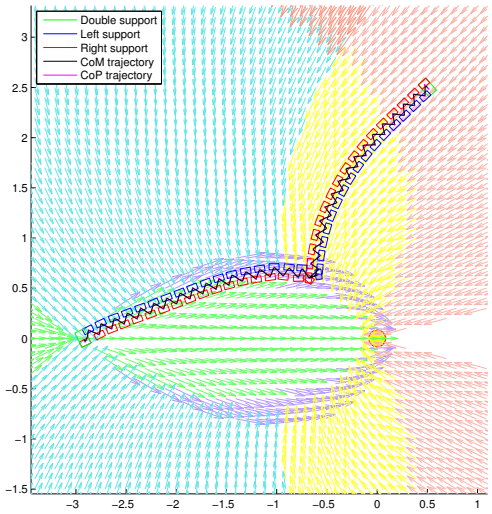


Fig. 1. Vector fields generated by optimal trajectories planned with visibility constraints for a non-holonomic robot [2]. To each color is associated a family of global paths that reach the goal (at $(-3, 0)$) according to the current configuration of the robot while keeping the landmark (yellow disk) visible at all times. The footsteps, CoM and CoP trajectories for the humanoid robot are all depicted as indicated in the upper box.

[1] Mauricio Garcia and Jean-Bernard Hayet are with CIMAT, A. C., Jalisco S/N Mineral de Valenciana, CP 36240 Guanajuato, Mexico {mjgarciav,jbhayet}@cimat.mx.

[2] Mauricio Garcia, Olivier Stasse and Jean-Paul Laumond are with LAAS-CNRS, 7 av. du Colonel Roche, F-31400, Toulouse, France, {mjgarcia,ostasse,jpl}@laas.fr.

[3] Claudia Esteves is with Universidad de Guanajuato, Guanajuato, Mexico, cesteves@cimat.mx.

in [14], we propose in this paper to use a planning approach integrating a constraint given by a task (e.g. visual-servoing). However, here, we modify the walking controller in such a way to use the planner as a generator of a full vector field that provides new local solutions from any given configuration and not only as a provider of single reference trajectories.

B. Contribution

Our contribution in this work is two-fold: (1) we propose to adapt a classical pattern generator algorithm to include visibility constraints and global planning data from an external motion planner; and (2), more broadly, we make a step towards filling in the gap between traditional motion planning approaches and on-line locomotion generation algorithms.

As a result of our approach, a footprint trajectory for a humanoid looking at the point $(0, 0)$, with initial configuration at the upper-right side corner, is depicted in Fig. 1[1].

This paper is organized as follows. In Section II, we give a summary of the global planner we use to produce paths that ensure the visibility of an object of interest in the plane. In Section III, we show how to use the optimal paths synthesis inside the pattern generation optimization, given an objective to reach and an object of interest to keep in sight. As the initial outputs from the planner are non-holonomic paths that may include in-site rotations, we propose in Section IV a strategy to utilize, when possible, the holonomic capabilities of the robot. In Section V, we present extensive simulation results of our approach and we conclude in Section VI.

II. Global planning with visibility constraints

For visual servoing, localization, surveillance, or any robotic task based on the observation of visual cues, it is important to ensure that the robot under control does not lose sight of the landmark(s) used as a visual reference. Hence, several works have been proposed to provide motion planners that guarantee landmark visibility, mainly for wheeled robots.

With this concern in mind, we borrow one such global planner, described in [6], as a base tool for generating paths that guarantee to enforce the required visual constraints while giving locally optimal trajectories in distance. This base planner for humanoid robots uses as an underlying model of motion, a disk-shaped non-holonomic Differential Drive Robot (DDR). In [6], if a solution path is found for the DDR, it is converted into a solution path for the humanoid robot by generating a footprints sequence coherent with the humanoid dimensions. A nice property inherited from using the DDR model of motion is that the global planner is complete, i.e. if a solution exists, it will give one, otherwise, it will say so.

However, even if, locally, the used motion primitives are optimal in distance, this approach does not necessarily give globally optimal trajectories. We will not detail the complete methodology here, but we recall the basic results hereafter.

The aforementioned algorithm takes as an input a 2D map of the environment, with its obstacles and visual landmarks, an initial configuration of the robot, (x_A, y_A, θ_A), and a final (goal) configuration, (x_B, y_B, θ_B). The visibility constraint on (x, y, θ) is written

$$\begin{aligned} \theta = \alpha - \phi + (2k+1)\pi,\ k \in \mathbb{Z}, \\ \phi_- \leq \phi \leq \phi_+, \end{aligned} \tag{1}$$

where ϕ_-, ϕ_+ are the angle sensor limits and α the configuration polar angle.

Its output is a set of footprints to be followed by the robot. The building of this path follows a recursive strategy: A roadmap is built over the free space, which includes both, points free of collision with the obstacles in the environment and points that are not within the shadows cast by the landmarks. Then, initial and final configurations are tested, to check whether the optimal primitives of [2] can connect them without colliding forbidden regions. If the computed path is safe, then the algorithm ends, otherwise the holonomic shortest path on the roadmap from the initial configuration to the final (if it exists) is divided in two at its middle point, and the process is repeated on the two sub-parts.

The resulting path is made of several parts, each one corresponding to a locally optimal path given from [2] (made of a combination of straight lines, logarithmic spirals and in-site rotations). This allows to define a set of S sub-segments to be performed by the robot along locally optimal paths, which will be referred to as (p^s, q^s), for $s = 1 \dots S$:

$$p^s = (x_p^s, y_p^s, \theta_p^s)^T \ ; \ q^s = (x_q^s, y_q^s, \theta_q^s)^T,$$

with p^s, q^s the initial and final configurations. The path is executed by reaching the successive sub-goals q^s.

III. Defining a local reference trajectory

Let us focus on each sub-path s. Suppose that we start with the robot at some configuration close to p^s (see Fig. 2), because localization may not be perfect, and that we want to reach the sub-goal q^s, as determined in [6].

A. Adapting the reference trajectory for the MPC

Without loss of generality, we suppose that the object of interest is located at $\mathbf{L}^s = (0, 0)$. In all the following, x, y, θ refer to coordinates w.r.t a global frame centered at this origin. It is noteworthy that, given the sub-goal $q^s = (x_q^s, y_q^s, \theta_q^s)^T$ to reach, and the landmark it is associated to, the work of [2] gives us a *full synthesis of optimal paths in free space*. This synthesis is represented as a mapping σ:

$$\begin{array}{rccc} \sigma & : & \mathbb{R}^2 & \mapsto & \mathbb{S}^1 \times \mathbb{R}^2 \\ & & (x, y) & \to & (\theta^*(x, y), v^*(x, y), \omega^*(x, y)) \end{array}$$

where $\theta^*(x, y)$ is the orientation the robot (viewed as a nonholonomic cart) should have in order to start walking along the shortest path to q^s, from its current position (x, y) [2]. The pair $(v^*(x, y), \omega^*(x, y))$ are the instantaneous (reference) velocities needed to follow the shortest paths. From the aforementioned work, we know that these paths are *either in-site rotations, or straight lines, or spiral segments*. Without loss of generality, we can suppose that $v^*(x, y) = 0$ (for in-site rotations) or $v^*(x, y) = \pm 1$ (elsewhere). Given

[1]Most figures in this paper are best appreciated in color.

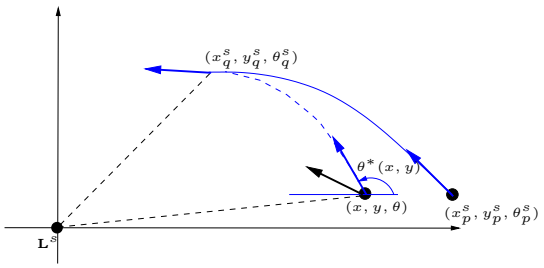


Fig. 2. From a configuration (x, y, θ) and its corresponding position (x, y), and a sub-goal $q^s = (x_q^s, y_q^s, \theta_q^s)$ to reach, the optimal path (dashed line) is the one we want the humanoid robot to follow. For that, we use the tangent orientation $\theta^*(x, y)$ to this path. Because of the errors in control or localization, this optimal path may be different from the shortest path (solid line) computed from the first configuration $p^s = (x_p^s, y_p^s, \theta_p^s)$.

the linear velocity $v^*(x, y)$, $\omega^*(x, y)$ is defined in function of the nature of the path segment,

$$\begin{cases} \omega^*(x,y) & = & 0 & \text{(straight line)} \\ & = & \pm\frac{\sin(\phi_+)}{\sqrt{x^2+y^2}} & \text{(spiral),} \end{cases}$$

where ϕ_+ is the maximal bearing angle possible for the landmark, given the sensor and robot characteristics.

Our claim is that the knowledge of optimal policies at each point allows some flexibility when generating a walking pattern, by optimizing the footprint positions and the CoM trajectory "around" the nominal path output from the planner, by using these policies within the pattern generation.

In pattern generation algorithms such as [1], the cost function within the Model Predictive Control (MPC) window uses the CoM configurations. Here, focusing on the (x, y, θ) CoM coordinates, we handle as an input of the algorithm a reference trajectory to be followed, and a synthesis of shortest paths leading to a given objective, as a direct output from [2]. As depicted in Fig. 2, at each configuration evaluated within the MPC, two forces should apply through the optimization scheme: one driving the robot to the "correct" orientation $\theta^*(x, y)$, and one making the velocities follow the shortest path, $(v(x, y), \omega(x, y))$. Also, a strong visibility constraint should apply, to ensure the object of interest to stay visible. In Fig. 3(a), we give an illustration of the shortest paths synthesis from [2], displayed through the local orientations of optimal paths at each point of the plane, given the objective to reach ("end point") and the landmark.

B. Quadratic programming MPC

Since the work of [4], walking pattern generation algorithms have focused on determining constant-height (at height h) trajectories of the CoM with piecewise-constant jerks time profiles. These profiles are computed by a MPC scheme for time intervals of length T, under a simplified cart-plane mechanical model simulated for a near time horizon, within N time intervals. In this paper, we follow a similar approach and will refer to the time interval indices as k.

To produce balanced and stable movements, [4] formulates the dynamics of the CoM, axis per axis, as

$$\begin{cases} \hat{x}_{k+1} & = & A\hat{x}_k + J\dddot{x}_k \\ \xi_k^x & = & Z\hat{x}_k \end{cases} \quad (2)$$

for the example of the x axis, where vectors $\hat{x}_k = (x_k, \dot{x}_k, \ddot{x}_k)^\top$ stack the position, velocity and acceleration of the CoM along the x-axis and ξ_k^x is the CoP $x-$coordinate, both at time k. The matrices A, J, Z are defined as

$$A = \begin{pmatrix} 1 & T & \frac{1}{2}T^2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{pmatrix}, J = \begin{pmatrix} \frac{1}{6}T^3 \\ \frac{1}{2}T^2 \\ T \end{pmatrix} \text{ and } Z = \begin{pmatrix} 1 & 0 & \frac{h}{g} \end{pmatrix},$$

where g is the gravity. To predict the CoM trajectories within the next N intervals, the dynamics is applied recursively, starting from the initial position $\hat{x}_k$, in terms of the jerk values to be applied, $\dddot{C}^x = (\dddot{x}_k, \dddot{x}_{k+1}, ..., \dddot{x}_{k+N-1})^\top$, where P_{ps} and P_{pu} are constant matrices,

$$C_{k+1}^x \stackrel{\text{def}}{=} \begin{pmatrix} x_{k+1} & \dots & x_{k+N} \end{pmatrix}^\top = P_{ps}\hat{x}_k + P_{pu}\dddot{C}_k^x. \quad (3)$$

Similar expressions can be obtained for the y component and for the velocity and acceleration of the CoM.

As an evolution of the original work of Kajita [4], where the footsteps positions (and, correspondingly, the CoP positions) are fed to the pattern generation, the work of [1] introduced automatic footstep planning, and reduced the necessary input to a simple stack of reference velocities $(\dot{X}_{k+1}^{ref}, \dot{Y}_{k+1}^{ref})$. This leads to the optimization problem:

$$\begin{aligned} \min_{U_k} \quad & \frac{\alpha}{2}\left\|\dddot{C}_k^x\right\|^2 + \frac{\beta}{2}\left\|\dot{C}_{k+1}^x - \dot{X}_{k+1}^{ref}\right\|^2 + \frac{\gamma}{2}\left\|Z_{k+1}^x - Z_{k+1}^{x_{ref}}\right\|^2 \\ & + \frac{\alpha}{2}\left\|\dddot{C}_k^y\right\|^2 + \frac{\beta}{2}\left\|\dot{C}_{k+1}^y - \dot{Y}_{k+1}^{ref}\right\|^2 + \frac{\gamma}{2}\left\|Z_{k+1}^y - Z_{k+1}^{y_{ref}}\right\|^2 \end{aligned} \quad (4)$$

with the variables to optimize kept in the vector $U_k \stackrel{\text{def}}{=} \left((\dddot{C}_k^x)^\top, (X_k^f)^\top, (\dddot{C}_k^y)^\top, (Y_k^f)^\top\right)^\top$. The ZMP references $Z_{k+1}^{x_{ref}}, Z_{k+1}^{y_{ref}}$ depend linearly on the variables X_k^f, Y_k^f which are the position of the next footsteps in the horizon.

As the sequence of CoP positions $Z_{k+1}^x = [\xi_{k+1}^x \ \cdots \ \xi_{k+N}^x]$ is linear in the variables to optimize (from Eq. 2), the problem is a Quadratic Program (QP)

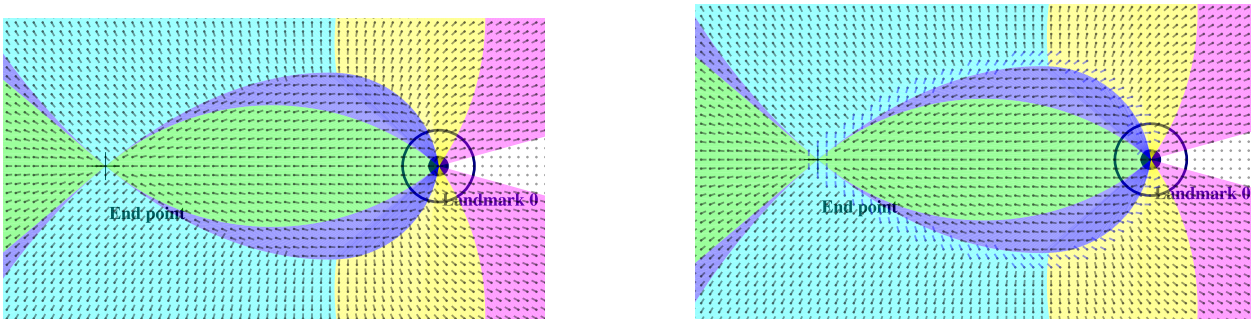
$$\min_{U_k} \frac{1}{2}U_k^\top Q_k U_k + p_k^\top U_k, \quad (5)$$

under linear constraints arising, among others, from the inclusion of the CoP inside the support polygon [1].

C. Using the optimal path synthesis within the MPC

Instead of utilizing in the Equation 4, a single, constant reference trajectory, defined by the plan computed from p^s to q^s, and from which $(\dot{X}_{k+1}^{ref}, \dot{Y}_{k+1}^{ref})$ would be evaluated, our idea is to use explicitly the mapping σ described above. This way, we can adapt the path execution and find the shortest path to the next sub-goal from any point, not just from p^s. As explained above, one can associate to any x, y the tangent $\theta^*(x, y)$ to the shortest path. One way to enforce the tracking of these shortest paths is to set as a reference velocity (supposing we are following a piece of curve with $v = 1$) defined in the time horizon ($l > k$, where k is the current time index):

$$\dot{x}_l^{ref} = \cos(\theta^*(x_l, y_l)) \qquad \dot{y}_l^{ref} = \sin(\theta^*(x_l, y_l))$$



(a) Motion direction field resulting from the synthesis of [2].

(b) Motion direction field with lateral motions.

Fig. 3. Visualization of optimal motion directions $\theta^*(x, y)$. Both figures indicate the nature of optimal paths with the filling colors, and the tangent direction to the optimal curve all over a grid defined around the final point. In Fig. (a), the field is computed directly from the primitives of [2]. The locus of in-site rotations is identifiable at the dark blue zone border. In Fig. (b), the non-holonomy behavior close to this locus is replaced by lateral motions.

that depends (non-linearly) on x_l, y_l. The idea is to use these – variable – reference velocities inside the terms of the pattern generator QP. However, as this mapping is non-linear, a direct use would make us lose the QP form, that allows an efficient resolution of the problem. Because the mapping $\theta^*(x, y)$ has not always an analytic form we evaluate it numerically on a fine scale grid inside the zone of operation of the robot, as pre-computed values, and we also estimate numerically the partial derivatives $\frac{\partial\theta^*(x,y)}{\partial x}, \frac{\partial\theta^*(x,y)}{\partial y}$.

This way, we can approximate each of these reference velocities, at time position l within the optimization time window, by performing a linearization of θ^* around a reference position (x^0, y^0), i.e.,

$$\begin{aligned}\theta^*(x_l, y_l) \quad \approx \quad & \theta^*(x_l^0, y_l^0) \\ & + \tfrac{\partial\theta^*(x^0,y^0)}{\partial x}(x_l - x^0) + \tfrac{\partial\theta^*(x^0,y^0)}{\partial y}(y_l - y^0).\end{aligned}$$

Note that the linearization point (x^0, y^0) is chosen in the aforementioned grid of pre-computed values $(\theta^*(x, y), \frac{\partial\theta^*(x,y)}{\partial x}, \frac{\partial\theta^*(x,y)}{\partial y})$, at the closest point to the first (current) CoM position. This grid is depicted in the background of Figs. 1, 3, 4 among others. To simplify the notation, let us write

$$\theta^0 \stackrel{\text{def}}{=} \theta^*(x^0, y^0), \quad \frac{\partial\theta^0}{\partial x} \stackrel{\text{def}}{=} \frac{\partial\theta^*(x^0, y^0)}{\partial x}, \quad \frac{\partial\theta^0}{\partial y} \stackrel{\text{def}}{=} \frac{\partial\theta^*(x^0, y^0)}{\partial y}.$$

Then, we re-write the errors to the reference velocities at each time step $l > k$, as a linear function of $\dot{x}_l, \dot{y}_l, x_l, y_l$

$$\left\{\begin{array}{rcl} \dot{x}_l - \dot{x}_l^{ref} & = & \dot{x}_l - v\cos(\theta^0) \\ & & + v\sin(\theta^0)\frac{\partial\theta^0}{\partial x}(x_l - x^0) \\ & & + v\sin(\theta^0)\frac{\partial\theta^0}{\partial y}(y_l - y^0), \\ \dot{y}_l - \dot{y}_l^{ref} & = & \dot{y}_l - v\sin(\theta_l^0) \\ & & - v\cos(\theta^0)\frac{\partial\theta^0}{\partial x}(x_l - x^0) \\ & & - v\cos(\theta^0)\frac{\partial\theta^0}{\partial y}(y_l - y^0), \end{array}\right.$$

and by stacking the errors within the horizon window as in Eq. 3, we get the following linear relations

$$\begin{aligned} \dot{C}_{k+1}^x - \dot{X}_{k+1}^{ref} &= \dot{C}_{k+1}^x - \dot{C}_{k+1}^{0,x} + A_0^x C_{k+1}^x + B_0^x C_{k+1}^y, \\ \dot{C}_{k+1}^y - \dot{Y}_{k+1}^{ref} &= \dot{C}_{k+1}^y - \dot{C}_{k+1}^{0,y} + A_0^y C_{k+1}^x + B_0^y C_{k+1}^y, \end{aligned} \quad (6)$$

where $A_0^x, B_0^x, A_0^y, B_0^y$ are diagonal matrices collecting the terms $v\sin(\theta_l^0)\frac{\partial\theta_l^0}{\partial x}$ and alike. Then, the walking pattern generation is formulated exactly as in Eq. 4, with the reference velocities given by Eq. 6, and with the optimization variable being U_k, leading to a canonical Quadratic Program (QP) similar to Eq. 5.

The constraints arising from the CoP position to be included in the support polygon being non-linear in the feet orientation, we simply set the robot and feet orientations as follows and include the computed values into the QP.

D. Control of the rotation angle

In order for the robot to be oriented with the tangent to the optimal path, θ^0, and to keep the QP form, we use a decoupled approach for the control of the rotation angles of the CoM and the feet [15]. Hence, in a first stage, we optimize the orientations in the MPC time window by

$$\begin{aligned} \min_{\dddot{C}_k^\theta, \dddot{F}_k^\theta} \quad & \frac{\beta}{2}\left\|C_{k+1}^\theta - \theta^0\right\|^2 + \frac{\gamma}{2}\left\|F_{k+1}^\theta - \theta^0\right\|^2 \\ & + \frac{\alpha}{2}\left\|\dddot{C}_k^\theta\right\|^2 + \frac{\alpha}{2}\left\|\dddot{F}_k^\theta\right\|^2, \end{aligned}$$

and then in a second stage, we introduce these angles as constant in the main QP (Eq. 4). This approach gives us the advantage of introducing constraints like maximum rotation between both feet, between feet and trunk and also a rotation limit to keep the visibility of the landmarks.

IV. Including holonomic behavior

Handling holonomic and non-holonomic behaviors together during locomotion has been previously been discussed in [16] in a context of motion planning.

One of the most visible disadvantages of the optimal non-holonomic paths given from [6] is the presence of in-site rotations, that are not efficient in terms of footsteps number. One solution is to use a different formulation of the cost function using weights according to the robot direction and control the head. But including this vision based control in Eq. 4 is incompatible with a QP formulation. A strategy we

propose here is to take advantage of the fact that the set of points where in-site rotations occur is very well defined geometrically in the plane, as a direct consequence of the synthesis from [2]. As illustrated in Fig. 3(a), it is the outer boundary of the partition zone where the shortest paths have to be done as line segments followed by spirals, i.e. the dark blue region of the figure. This curve is made of two parts [2], one arc of circle and one piece of logarithmic spiral. Hence, we propose to perform the following: for all points inside the partition regions in contact with this locus, we evaluate its distance in terms of the primitive to be done to reach this locus and modify the reference velocities as follows.

If the robot is far from the locus of in-site rotations, then the path to follow is continuously derivable and goes either forwards or backwards; in that case, we use the "non-holonomic behavior" as described in Section III, with the robot orientation controlled to stay close to $\theta^*(x_l, y_l)$,

$$\dot{x}_l^{ref} = \cos(\theta^*(x_l, y_l)) \ , \ \dot{y}_l^{ref} = \sin(\theta^*(x_l, y_l)),$$

and if the robot configuration is close to this locus, then we use instead lateral motions, with orientation $\phi(x_l, y_l)+\pi$, where $\phi(x_l, y_l) = \arctan \frac{y_l}{x_l}$ is the polar angle of (x_l, y_l),

$$\dot{x}_l^{ref} = -\varepsilon \sin(\phi(x_l, y_l)) \ , \ \dot{y}_l^{ref} = \varepsilon \cos(\phi(x_l, y_l)),$$

where $\varepsilon = \pm 1$ in function of the relative position of the goal to reach with respect to the evaluated point.

In Fig. 3(b), we illustrate this modification by drawing the lateral motion field with blue arrows, together with the "non holonomic" field "far" from the locus of in-site rotations.

V. Results

In this section, we present three experiments. In the first one, we test the performance of our approach in the absence of localization uncertainty, and with only non-holonomic motion. In the second one, we introduce localization uncertainty. In the last one, we present an improvement to the border behavior with the possibility of holonomic motion. The three experiments have the same set-up, with initial position in $(0.5, 2.5)$, landmark position in $(0, 0)$ and final position in $(-3, 0)$. We chose this trajectory because it makes the robot pass through several regions of the partition and illustrates the border behavior which is one of the problems we found.

In Fig. 1, we depict the performance in perfect and noiseless conditions. We see that using the planner vector field for driving the robot to the goal produces smooth and stable trajectories of the CoM. However, in real conditions, the robot will not perform the control exactly e.g., because of sliding with the floor. Moreover the robot needs a localization system which will be inherently noisy. To model this situation we perturbed the current position of the CoM with white gaussian noise $\sim \mathcal{N}(0, \sigma^2)$ in each coordinate with $\sigma = 0.2$ and with $\sigma = 10$ degrees for the orientation.

In Fig. 4, left, we show the behavior in that situation. We can still appreciate a smooth trajectory in the regions far from the border of the dark blue region of the partition, where the vector field is smooth. We also notice the good behavior of the orientation angle mainly because of the QP formulation. The main problem of this approach also appears: The discontinuities in the partition regions in which an in-site rotation occurs. Hence, the robot may perform unnecessary maneuvers, involving in-site rotations, which is not optimal in the case of human walking (Fig. 4, right). Moreover, in-site rotations cause high rotation speeds of the feet (Fig. 5), not desirable from a balance point of view.

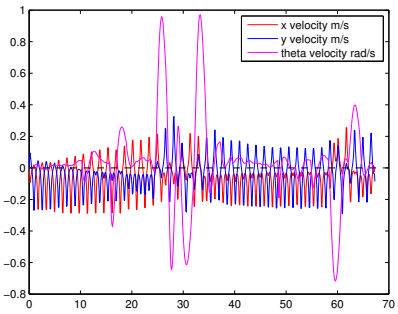


Fig. 5. Velocities profile for the trajectory depicted in Fig. 4. The overshoots in the angular velocity are caused by the in-site rotations at the border of the dark blue region of the partition.

To address the aforementioned problem, we present the results of the improvement introduced in Section IV. In Fig. 6 left, we perform the same trajectory but by introducing the possibility of holonomic motion in the region where in-site rotations would be necessary. The robot is not performing in-site rotations anymore, but a mixture of holonomic and non-holonomic motions, depending on its position with respect to the border. We can appreciate better the transition between non-holonomic and holonomic motions in Fig. 6, right.

VI. Conclusion

In most of the current literature, the link between planning and locomotion control has been given by footsteps. In this paper, we propose a novel approach that uses directly the optimal motion synthesis derived from the planner to the control system without going through footsteps but instead by computing them within the Walking Pattern Generator. With this approach, we can drive the robot to a desired goal by using an external planner, not necessarily the one used in this paper. We tested our approach simulating real situations such a localization noise. We have also tried to make the walking pattern more efficient by introducing the possibility of using holonomic motion. Our next objective is to test the approach introduced in this paper on the HRP-2 platform.

Acknowledgments

M. Garcia is supported by grant 263150 from the Mexican National Council of Science and Technology (CONACYT). O. Stasse is supported by the French project Romeo II.

References

[1] A. Herdt, H. Diedam, P.-B. Wieber, D. Dimitrov, K. Mombaur, and M. Diehl, "Online walking motion generation with automatic footstep placement," *Advanced Robotics*, vol. 24, no. 5-6, pp. 719–737, 2010.

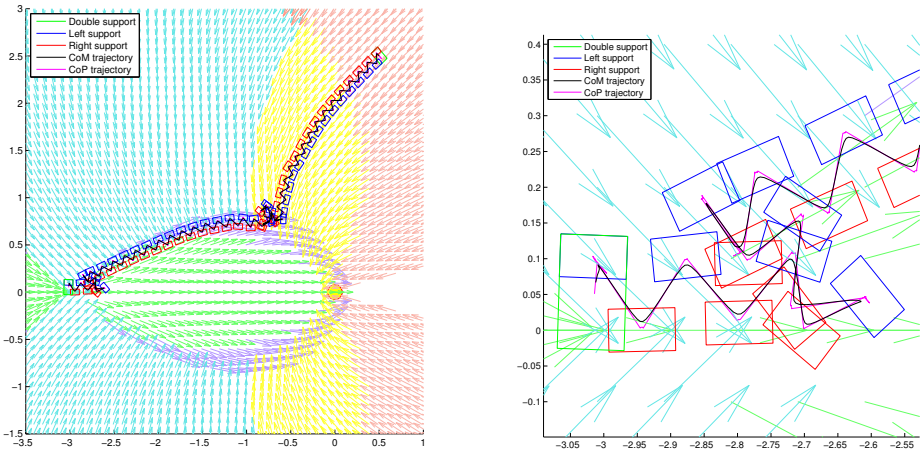


Fig. 4. Left: If the control is not followed well and if we use an imperfect localization system, problems may arise in the borders of the partition, in case of relying on purely non-holonomic behaviors. Zoom on the left picture: Close to $(-3, 0)$. As the robot compensates sliding and localization errors non-holonomically, more in-site rotations occur.

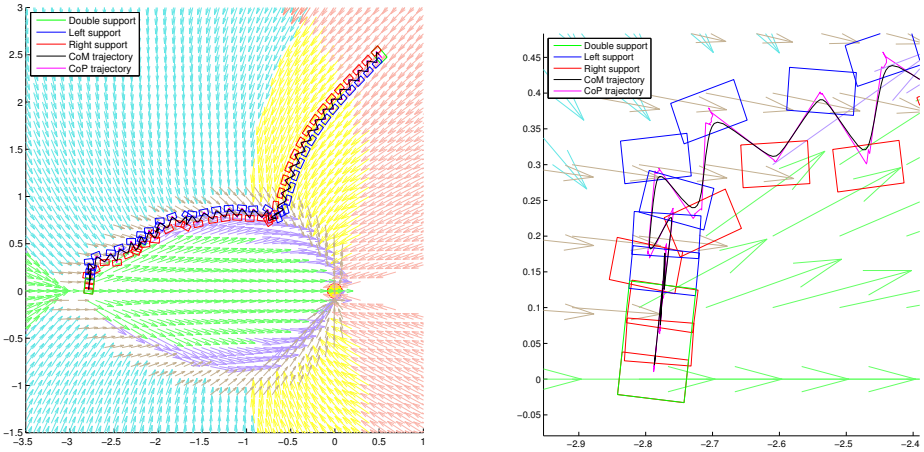


Fig. 6. Left: Trajectory with holonomic motion made possible. Lateral motion is allowed in the khaki vector field in the orientation discontinuity region. Right: Transition between non-holonomic and holonomic controls. Moving sideways is much more efficient for such a trajectory to the goal.

[2] 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*, vol. 26, pp. 269–281, 2010.

[3] M. Garcia, O. Stasse, and J.-B. Hayet, "A watching-while-walking pattern generator," in *IROS*, 2013, p. submitted.

[4] 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 *ICRA*, 2003, pp. 1620–1626.

[5] J. Chestnutt, M. Lau, G. Cheung, J. Kuffner, J. Hodgins, and T. Kanade, "Footstep planning for the honda asimo humanoid," in *ICRA*, 2005, pp. 629–634.

[6] J.-B. Hayet, C. Esteves, G. Arechavaleta, O. Stasse, and E. Yoshida, "Humanoid locomotion planning for visually-guided tasks," *Int. Journal of Humanoid Robotics*, vol. 09, pp. 1–26, 2012.

[7] J.-M. Bourgeot, N. Cislo, and B. Espiau, "Path-planning and tracking in a 3d complex environment for an anthropomorphic biped robot." in *IROS*, vol. 3, 2002, pp. 2509–2514.

[8] J. Chestnutt, K. Nishiwaki, J. Kuffner, and S. Kagami, "An adaptive action model for legged navigation planning," in *ICRA*, 2007, pp. 196–202.

[9] A. Hornung and M. Bennewitz, "Adaptive level-of-detail planning for efficient humanoid navigation," in *ICRA*, 2012, pp. 997–1002.

[10] K. Nishiwaki and S. Kagami, "Online walking control system for humanoids with short cycle pattern generation," *IJRR*, vol. 28, no. 6, pp. 729–742, 2009.

[11] S.-J. Yi, B.-T. Zhang, D. Hong, and D. Lee, "Online learning of a full body push recovery controller for omnidirectional walking," in *ICHR*, 2011, pp. 1–6.

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

[13] D. Dang, F. Lamiraux, and J.-P. Laumond, "A framework for manipulation and locomotion with realtime footstep replanning," in *ICHR*, 2011, pp. 676–681.

[14] N. Vahrenkamp, D. Berenson, T. Asfour, J. Kuffner, and R. Dillmann, "Humanoid motion planning for dual-arm manipulation and re-grasping tasks," in *IROS*, 2009, pp. 2464–2470.

[15] A. Herdt, N. Perrin, and P.-B. Wieber, "Walking without thinking about it," in *IROS*, 2010, pp. 190–195.

[16] K. Mombaur, J.-P. Laumond, and E. Yoshida, "An optimal control model unifying holonomic and nonholonomic walking," in *ICHR*, 2008, pp. 648–653.