Vision-driven walking pattern generation for humanoid reactive walking

Mauricio Garcia, Olivier Stasse, Jean-Bernard Hayet

▶ **To cite this version:**

Mauricio Garcia, Olivier Stasse, Jean-Bernard Hayet. Vision-driven walking pattern generation for humanoid reactive walking. International Conference on Robotics and Automation, May 2014, Hong-Kong, China. 7p. hal-00990088

HAL Id: hal-00990088
https://hal.science/hal-00990088

Submitted on 13 May 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Vision-driven walking pattern generation for humanoid reactive walking

Mauricio Garcia[1,2] and Olivier Stasse[2] and Jean-Bernard Hayet[1]

***Abstract*— We present a novel approach to introduce visual information in the walking pattern generator for humanoid robots in a more direct way than the current existing methods. We make use a model predictive control (MPC) visual servoing strategy, which is combined to the walking motion generator. We define two schemes based on that principle: a position-based and an image-based scheme, with a Quadratic Program (QP) formulation in both cases. Finally, we present some simulation results validating our approach.**

I. Introduction

The last years have seen the rise of a new wave of very efficient control systems for humanoid robots walking generation, based on dynamical balance criteria such as the position of the Center Of Pressure (CoP), that can compute the foot steps placement in a completely online way [6], opening the door to reactive navigation strategies in humanoid robotics. Linear Model Predictive Control (LMPC) is the core element of most of these methods. LMPC "previews" the behavior of the system by applying virtually a sequence of controls from the current robot configuration and estimates, with a simple but very efficient dynamical model, the result of these controls at a given horizon. The optimization over jerk sequences can be remarkably written as a Quadratic Program (QP), subject to a set of constraints explicitly handled (e.g., the CoP position being inside the support polygon). The result of this optimization is a whole sequence of controls, and one only applies the first optimal control in the next iteration. Hence, the key of these methods is to get very fast optimization to be able to generate new optimal controls at a high frequency, which is possible because of the existence of very efficient techniques to solve QPs.

Now, the next step would consist in finding a way to "close the loop" in the traditional perception-decision-action paradigm, and use perceptual data as a feedback to the locomotion, e.g. for positioning precisely the robot. Our work has been oriented towards this goal, and more precisely towards the integration of visual errors from a visual servoing task to guide the robot through locomotion, within the same LMPC framework mentioned above. Some approaches exist that directly define footsteps in the time horizon based on sensors information about the environment, i.e. integrated through mapping, in applications such as tracking, stair-climbing or obstacle avoidance [2], [5], [8]–[11]. On the other hand, for some tasks such as manipulation or interaction with human,

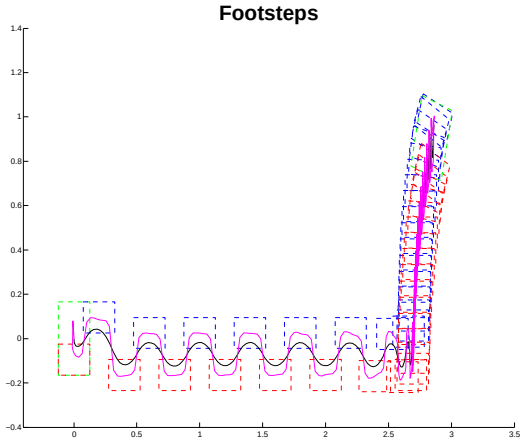

Fig. 1. Humanoid walking pattern generation (WPG): Our algorithm generates a Center of Mass (CoM) trajectory and a sequence of footsteps to be done to reach a visual reference (defined here by four points), while ensuring stability during the walk. In this figure, we depict the generated foot steps (red and blue rectangles), the CoM trajectory (black) and CoP trajectory (magenta). In the first part of the trajectory, the visual errors are reduced by an initial rotation and a motion in direction to the target; in the second part, they are reduced by a circular arc around the target.

it may be required to get precise positioning only relatively to one specific feature, e.g. the manipulation workspace or the human with which the robot will interact, without the need for a global mapping strategy. For this purpose, visual servoing has been very popular in the last two decades in wheeled robotics, and its use in humanoid robotics is only at its beginning. In one of the pioneering works, Dune et al. [3] have proposed to couple a "traditional" visual control scheme with one of the aforementioned walking pattern generation (WPG) methods, seen as a black box. The idea is to compute a desired velocity of the center of mass (CoM) by using visual servoing, and to use it as a reference velocity in the pattern generator. This method suffers from a number of drawbacks. First, because the visual servoing scheme and the pattern generator are completely decoupled, the visual information is not directly feeding the pattern generator, it simply gives a reference velocity to follow, which is not guaranteed to be followed. Second, the reference velocity computed from the visual servoing module takes into account visual features undergoing a rather complex motion in the image, because the camera itself undergoes a not-so-easy-to-model motion in the world, because of the robot stepping. In [3], this problem is tackled by performing the visual servoing not on the real, observed features, but instead on virtual features corresponding to "ideal" features from which

[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 and Olivier Stasse are with LAAS-CNRS, 7 av. du Colonel Roche, F-31400, Toulouse, France, {mjgarcia,ostasse}@laas.fr.

the sway motion have been removed. A similar approach has been recently proposed by Faragasso and al. [4]. In this later paper, the velocity is considered by the integration of the robot displacement over a step period and is relying on a "black-box" approach with the WPG presented in [6].

In this paper, we propose a simpler alternative, that can face much more naturally to the problems mentioned before, and, above all, is much more easily integrated with the locomotion algorithm, instead of operating as a separate "black box". Our idea is that since locomotion is based on previewing algorithms such as LMPC, we can also "preview" the visual features evolution given the application of a series of controls, and use it to guide the steps of the robot directly within the LMPC. The main difficulty is to keep the problem formulation as a QP. To this end, we build upon two families of works, the one from Herdt et al. [6] about the LMPC locomotion formulation and the one from Allibert et al. [1], that has used Model Predictive Control in visual servoing.

In Section II, we give reminders about the WPG and about model predictive visual servoing. Then we present two approaches to include the visual servoing: a PBVS scheme in Section III, and an IBVS scheme in Section IV. We present results in Section V and conclude in Section VI.

II. Walking pattern generation

We recall how the current walking motion generation schemes, inherited from [7], settle the generation of balanced and stable motion through the determination of a CoM trajectory. Details can be found, e.g. in [6]. Suppose that the trajectory has periodic piece-wise constant jerks on time intervals T, the CoM dynamics along the x-axis is

$$\begin{aligned} \hat{x}_{k+1} &= \mathbf{A}\hat{x}_k + \mathbf{B}\dddot{x}_k \\ \xi^x_k &= \mathbf{C}\hat{x}_k \end{aligned} \tag{1}$$

where $\hat{x}_k = (x_k, \dot{x}_k, \ddot{x}_k)^\top$ stacks the $x-$position, velocity and acceleration of the CoM at time k, and $\mathbf{A}, \mathbf{B}, \mathbf{C}$ constant matrices. The vector ξ^x_k is the CoP at time k. By applying recursively Eq. 1, we express the CoM trajectory on larger intervals, in terms of the initial position $\hat{x}_k$ and the whole sequence of applied jerks $\dddot{X}_k = (\dddot{x}_k, \dddot{x}_{k+1}, ..., \dddot{x}_{k+N-1})^\top$,

$$X_{k+1} = \begin{pmatrix} x_{k+1} \dots x_{k+N} \end{pmatrix}^T = \mathbf{P}_{ps}\hat{x}_k + \mathbf{P}_{pu}\dddot{X}_k. \tag{2}$$

Similar expressions can be derived for the CoM velocity and acceleration, and for the y or heading components. In [7], the positions of the center of pressure (CoP) are an input, so that a preliminary footstep planning was necessary. Then, a re-formulation was proposed to handle automatic footstep placement [6], by using a reference velocity $(\dot{X}^{ref}_{k+1}, \dot{Y}^{ref}_{k+1})$. The resulting constrained optimization problem is:

$$\begin{aligned} \min_{U_k} \quad & \frac{\alpha}{2}\left\|\ddot{X}_k\right\|^2 + \frac{\beta}{2}\left\|\dot{X}_{k+1} - \dot{X}^{ref}_{k+1}\right\|^2 + \frac{\gamma}{2}\left\|Z^x_{k+1} - Z^{x_{ref}}_{k+1}\right\|^2 \\ & + \frac{\alpha}{2}\left\|\ddot{Y}_k\right\|^2 + \frac{\beta}{2}\left\|\dot{Y}_{k+1} - \dot{Y}^{ref}_{k+1}\right\|^2 + \frac{\gamma}{2}\left\|Z^y_{k+1} - Z^{y_{ref}}_{k+1}\right\|^2 \end{aligned} \tag{3}$$

with $U_k \stackrel{\text{def}}{=} \left(\ddot{X}_k^\top, (X^f_k)^\top, \ddot{Y}_k^\top, (Y^f_k)^\top\right)^\top$, and X^f_k, Y^f_k the footsteps positions. The vector of CoP values on the horizon is $Z^x_{k+1} = [\xi^x_{k+1} \ \cdots \ \xi^x_{k+N}]$. The $Z^{x_{ref}}_{k+1}$ are the center of the support polygon at each iteration. This can be written as a canonical Quadratic Program (QP)

$$\min_{U_k} \frac{1}{2}U_k^\top \mathbf{Q}_k U_k + p_k^\top U_k. \tag{4}$$

The main contribution of this paper is that instead of controlling the robot through the references velocities (which may be derived from a traditional visual servoing approach [3]), we do it by using directly the visual features. We keep the problem as a QP so it can be solved efficiently in real time, in an optimization scheme similar to 4.

In another area, visual servoing comes in two main flavors: Image based visual servoing (IBVS) uses image features; Position based visual servoing (PBVS) uses the 3-D pose(s) of object(s) of interest. In both cases, a velocity controller such as $\mathrm{v}^c = -\lambda\mathbf{L}^+\mathbf{e}$, uses $\mathbf{e}$, the current vector of errors, and $\mathbf{L}^+$, the Moore-Penrose pseudo-inverse of the interaction matrix $\mathbf{L}$ relating the velocity of the features and the velocity of the camera. It outputs v^c, the velocity of the camera. Here, we orient the optimization of the jerks and foot placements (Eq. 3) by taking into account the expected evolution of the visual servoing (VS) errors so that, instead of minimizing the VS errors at current time k, one would like to foresee its evolution at the horizon $[k+1, k+N]$.

In [1], such a time horizon-aware scheme has been proposed. It introduces visual predictive control (VPC) as:

$$\min_{U_k} \quad \sum_{j=k+1}^{k+N} [\mathbf{s}^d_j - \mathbf{s}^m_j]^\top \mathbf{W}_j [\mathbf{s}^d_j - \mathbf{s}^m_j], \tag{5}$$

$$\text{subject to} \quad \mathbf{s}^d_j = \mathbf{s}^* - \mathbf{e}_j, \tag{6}$$

$$\mathbf{q}_j = f(\mathbf{q}_{j-1}, u_{j-1}), \tag{7}$$

$$\mathbf{s}^m_j = h(\mathbf{q}_j). \tag{8}$$

In Eq. 5, $U_k = u_{k:k+N-1}$ are the series of controls to be applied, $\mathbf{q}_j$ is the state, and $\mathbf{s}^*$, $\mathbf{s}^d_j$ and $\mathbf{s}^m_j$ are respectively the (static) reference, desired and predicted positions of the visual features. The terms $\mathbf{e}_j$ are the errors $\mathbf{s}_j - \mathbf{s}^m_j$ between real and predicted feature positions, assumed constant over the prediction horizon, equal to $\mathbf{e}_k = \mathbf{s}_k - \mathbf{s}^m_k$, i.e. the error at current time k, because by definition the s_j are not known for $j > k$. Since the errors are constant on the horizon window, $\mathbf{s}^d_j = \mathbf{s}^d_k = \mathbf{s}^* - \mathbf{e}_k$ are constant in the prediction horizon.

The Eq. 7 estimates the new state given the last state/control pair. In general, it is non-linear. Eq. 8 estimates the model output $\mathbf{s}^m_j$, given the current state $\mathbf{q}_j$, e.g., the camera projection model. Matrix $\mathbf{W}_j$ is used to weight errors in the prediction horizon and as suggested in [1], we consider $\mathbf{W}_j = diag(w_j)$. The $\mathbf{s}^m_j$ are the M predicted features at time j, $\mathbf{s}^m_j = (s^m_{1,j}, s^m_{2,j}, \dots, s^m_{M,j})^\top$. Eq. 5 can be written

$$\min_{U_k} \sum_{l=0}^{M} [\mathbf{S}^d_l - \mathbf{S}^m_{l,k}]^\top \mathbf{W} [\mathbf{S}^d_l - \mathbf{S}^m_{l,k}], \tag{9}$$

with $\mathbf{S}^m_{l,k}$ stacking the features positions in the horizon: $\mathbf{S}^m_{l,k} = \left(s^m_{l,k+1}, s^m_{l,k+2}, \ldots, s^m_{l,k+N}\right)^\top$, and S^d_l stacking the corresponding desired positions. This form motivates us in combining Eq. 9 and Eq. 3 to generate vision-driven motion primitives for reactive walking.

III. Position-Based Visual Servoing for Walking Pattern Generation

Using directly Eq. 9 in Eq. 3, we would lose the QP formulation due to the non-linearities in Eqs. 7 and 8. We can solve the first problem (Eq. 7) by using a similar approach as in [6], e.g. with the dynamic model of Eq. 2.

A. Linearization of the observation model

As we said, Eq. 8 implements the pinhole camera model. Let $p^w_{l'} = [x^w_{l'}, y^w_{l'}, z^w_{l'}]^\top$ be the 3D position of the l'-th landmark in the world frame. One can project it to the image plane by first transforming the landmark to the camera frame with the homogeneous transform $\mathbf{T}^{mc}\mathbf{T}^{wm}_j$ and then applying the projection $u(x,y,z) = \frac{x}{z}, v(x,y,z) = \frac{y}{z}$,

$$\begin{pmatrix} u_{l',j} \\ v_{l',j} \end{pmatrix} = \begin{pmatrix} u(x^c_{l',j}, y^c_{l',j}, z^c_{l',j}) \\ v(x^c_{l',j}, y^c_{l',j}, z^c_{l',j}) \end{pmatrix} = \begin{pmatrix} x^c_{l',j}/z^c_{l',j} \\ y^c_{l',j}/z^c_{l',j} \end{pmatrix}, \quad (10)$$

where $\mathbf{T}^{wm}_j$ is the transformation from the world to the CoM frame (varying with j) and $\mathbf{T}^{mc}$ is the transformation from the CoM to the camera frame, constant in our approach,

$$\mathbf{T}^{wm}_j = (\mathbf{T}^{mw}_j)^{-1} = \begin{pmatrix} (\mathbf{R}^{mw}_j)^{-1} & -(\mathbf{R}^{mw}_j)^{-1} t^{mw}_j \\ 0_{1\times 3} & 1 \end{pmatrix}$$

where t^{mw}_j is the position of the CoM in the world frame at time j, which depends in our control variables through Eq. 2. $\mathbf{R}^{mw}_j$ is the direction of the robot waist according to the world reference frame at time j. In our current formulation there is no free variables modifying this quantity, because it would break the problem linearity. More details about this problem are given in Section III-B.

In MPC, prediction is done over a finite horizon. Let us use a first order approximation of the projection (Eq. 10) for small (dx, dy, dz) to maintain the QP form. By using a Taylor series for $u(x,y,z)$ around some point (x_0, y_0, z_0) and substituting the derivatives,

$$\begin{cases} u(x_0+dx, y_0+dy, z_0+dz) \approx \frac{x_0}{z_0} + \frac{dx}{z_0} - \frac{x_0 dz}{z_0^2}, \\ v(x_0+dx, y_0+dy, z_0+dz) \approx \frac{y_0}{z_0} + \frac{dy}{z_0} - \frac{y_0 dz}{z_0^2}, \end{cases}$$

with $dx = x - x_0$, $dy = y - y_0$ and $dz = z - z_0$.

We linearize Eq. 10 for the *whole horizon*, around the first position $(j = k)$ of landmark l', i.e. at the linearization point $(x^c_{l',k}, y^c_{l',k}, z^c_{l',k})$. This way, we can express linearly the predicted position of the landmark l' at time $j > k$:

$$\begin{pmatrix} u_{l',j} \\ v_{l',j} \end{pmatrix} = \begin{pmatrix} \pi^{11}_{l',k} x^c_{l',j} + \pi^{13}_{l',k} z^c_{l',j} + u_{l',k} \\ \pi^{22}_{l',k} y^c_{l',j} + \pi^{23}_{l',k} z^c_{l',j} + v_{l',k} \end{pmatrix},$$

where $u_{l',k} = x^c_{l',k}/z^c_{l',k}$ and $v_{l',k} = y^c_{l',k}/z^c_{l',k}$ are the initial positions of the landmarks in the horizon and the coefficients π_{ij} the elements of the matrix

$$\Pi_{l',k} = \begin{pmatrix} 1/z^c_{l',k} & 0 & -u_{l',k}/z^c_{l',k} \\ 0 & 1/z^c_{l',k} & -v_{l',k}/z^c_{l',k} \end{pmatrix}.$$

Finally, we project the $l'-th$ landmark (constraint 8)

$$u_{l',j} = a^u_{l',k} x_j + b^u_{l',k} y_j + c^u_{l',k}, \quad (11)$$

with

$$\begin{cases} a^u_{l',k} &= \Pi^u_{l',k} \mathbf{R}^{wc}_1 \\ b^u_{l',k} &= \Pi^u_{l',k} \mathbf{R}^{wc}_2 \\ c^u_{l',k} &= \Pi^u_{l',k} (\mathbf{R}^{wc} p^w_{l'} + t^{mc} + \mathbf{R}^{wc}_3 z_j) + u_{l',k}. \end{cases}$$

where $\Pi^u_{l',k}$ is the first row of $\Pi_{l',k}$, $\mathbf{R}^{wc} t^{mw}_j = \mathbf{R}^{wc}_1 x_j + \mathbf{R}^{wc}_2 y_j + \mathbf{R}^{wc}_3 z_j$, and $\mathbf{R}^{wc}_i$ is the i-th column of $\mathbf{R}^{wc}$ [1] and x_j, y_j, z_j the position of the CoM at time j (see Section II). Equivalently, we have $v_{l',j} = a^v_{l',k} x_j + b^v_{l',k} y_j + c^v_{l',k}$.

Stacking the features $u_{l',j}$ and the CoM positions for the whole horizon and using Eq. 2, we get a vector $\mathbf{S}^m_{l,k}$ similar to the one introduced in Eq. 9

$$\mathbf{S}^m_{l',k} = \mathbf{A}^u_{l',k} X_{k+1} + \mathbf{B}^u_{l',k} Y_{k+1} + \mathbf{C}^u_{l',k},$$

with $\mathbf{A}^u_{l',k} = a^u_{l',k} \mathbf{I}_{N\times N}$, $\mathbf{B}^u_{l',k} = b^u_{l',k} \mathbf{I}_{N\times N}$, and $\mathbf{C}^u_{l',k} = c^u_{l',k} \begin{pmatrix} 1, 1, \ldots 1 \end{pmatrix}^\top$, which corresponds to the predicted coordinates u of the landmark l'-th in the horizon.

Every projected landmark provides two coordinates (u, v) and we treat each one as an individual feature. This means that the l-th feature is the u-coordinate of landmark $l' = \lfloor l/2 \rfloor$ for l even, and the v-coordinate for l odd. Generalizing:

$$\mathbf{S}^m_{l,k} = \mathbf{A}_{l,k} X_{k+1} + \mathbf{B}_{l,k} Y_{k+1} + \mathbf{C}_{l,k}, \quad (12)$$

with $\mathbf{A}_{l,k} = \mathbf{A}^u_{l',k}$ for l even and $\mathbf{A}_{l,k} = \mathbf{A}^v_{l',k}$ for l odd. The same holds for $\mathbf{B}_{l,k}$ and $\mathbf{C}_{l,k}$. Finally, we can introduce visual servoing (Eq. 12) in the walking generation as a canonical QP, by minimizing the following objective function of $U_k \stackrel{\text{def}}{=} \left(\dddot{X}_k^\top, (X^f_k)^\top, \dddot{Y}_k^\top, (Y^f_k)^\top\right)^\top$

$$\begin{aligned} \min_{U_k} \quad & \frac{\alpha}{2} \left\| \dddot{X}_k \right\|^2 + \frac{\alpha}{2} \left\| \dddot{Y}_k \right\|^2 \\ & + \frac{\gamma}{2} \left\| Z^x_{k+1} - Z^{x_{ref}}_{k+1} \right\|^2 + \frac{\gamma}{2} \left\| Z^y_{k+1} - Z^{y_{ref}}_{k+1} \right\|^2 \\ & + \frac{\beta}{2} \sum_{l=0}^{M} [\mathbf{S}^d_l - \mathbf{S}^m_{l,k}]^\top \mathbf{W} [\mathbf{S}^d_l - \mathbf{S}^m_{l,k}]. \end{aligned}$$

B. Control of the rotation angle

So far, we have proposed a scheme to control the trajectory of the center of mass in x and y, and we supposed the heading angle constant. Introducing this heading angle in the minimization problem is not straightforward without losing linearity. Furthermore, the rotation angle plays a very important role here since sometimes most of the error

[1] To simplify the notations $R^{wc} = R^{mc} R^{wm}_j$

between $\mathbf{s}^d$ and $\mathbf{s}^m$ may be due to the angle. We propose to use the extension of the LMPC scheme proposed in [6], that deals with a reference angular velocity. The solution is decoupled, and first estimates the optimal rotation angles and introduces these values as known in the main QP ($\mathbf{R}_j^{mw}$). Hence, we first optimize the sequence of heading jerks by

$$\min_{\dddot{\Theta}_k, \dddot{\Theta}_k^f} \quad \frac{\beta}{2}\left\|\Theta_{k+1} - \Theta^0\right\|^2 + \frac{\gamma}{2}\left\|\Theta_{k+1}^f - \Theta^0\right\|^2 + \frac{\alpha}{2}\left\|\dddot{\Theta}_k\right\|^2 + \frac{\alpha}{2}\left\|\dddot{\Theta}_k^f\right\|^2, \tag{13}$$

with the same notations as for $\dddot{X}_k$ and $\dddot{Y}_k$: $\dddot{\Theta}_k$ is the sequence of N jerk values to apply, and Θ_{k+1} is the sequence of predicted θ values, i.e. the orientations of the trunk, $\Theta_{k+1} \stackrel{\text{def}}{=} (\theta_{k+1}, ..., \theta_{k+N})^\top$, and similarly for Θ_{k+1}^f, the feet orientations. A reference Θ^0 is defined at the starting configuration as a target feet orientation. Moreover, the trunk orientation Θ_k tries to follow the flying foot orientation Θ_k^f. The flying foot is the only one that moves in single support phase. Finally, zero speed, and zero acceleration are specified at both ends of the trajectories. In a second stage, we introduce these angles as constant in Eq. 3, in particular for defining the ZMP constraint. 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.

C. Visual constraints

Any linear constraint in the image plane (u, v), can be expressed as a linear constraint in the variables U_k. It means that we can have time- and landmark-varying constraints:

$$\mathbf{A}' \begin{pmatrix} \mathbf{A}_{l',k}^u X_{k+1} + \mathbf{B}_{l',k}^u Y_{k+1} + \mathbf{C}_{l',k}^u \\ \mathbf{A}_{l',k}^v X_{k+1} + \mathbf{B}_{l',k}^v Y_{k+1} + \mathbf{C}_{l',k}^v \end{pmatrix} \leq b'. \tag{14}$$

Bound constraints in (u, v) like visibility constraints are easily expressed with Eq. 14 and introduced in the QP.

IV. Image-Based Visual Servoing for Walking Pattern Generation

If we come back to $\mathbf{s}_k$, the vector of visual features, and relate it to $\mathbf{v}_k^c$, the instantaneous velocity vector of the camera at time k (in $\mathbb{R}^6$), we can derive the following equation, involving the interaction matrix $\mathbf{L}_k$, $\dot{\mathbf{s}}_k = \mathbf{L}_k \mathbf{v}_k^c$.

Suppose that the variations are piecewise constant on each time intervals of duration T of the Model Predictive Control, and then integrate in time the previous relation, we have

$$\mathbf{s}_{k+1} = \mathbf{s}_k + T\mathbf{L}_k \mathbf{v}_k^c. \tag{15}$$

Assume that the twist matrix ${}^c\mathbf{V}_m$ relating the CoM and the camera velocities remains constant: $\mathbf{v}_k^c = {}^c\mathbf{V}_m \mathbf{v}_k^m$.

By applying recursively the Eq. 15, we can express the position of the features at time j inside the horizon window (i.e., in absolute time $k + j$) in terms of the velocity of the CoM $\mathbf{v}_m = (\dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\psi}, \dot{\theta})$

$$\mathbf{s}_{k+j} = \mathbf{s}_k + T \sum_{i=k}^{k+j-1} \mathbf{L}_i^c \mathbf{V}_m \mathbf{v}_i^m.$$

Now if we suppose that, within the horizon window, at time i, the interaction matrix $\mathbf{L}_i$ stays close to what is is in k, i.e. $\mathbf{L}_i \approx \mathbf{L}_k$, and if we make appear all the velocity components, and the corresponding columns of the twist matrix, $({}^c\mathbf{V}_m)_j$ for $j \in \{1 \dots 6\}$

$$\mathbf{s}_{k+j} = \mathbf{s}_k + T\mathbf{L}_k \left[({}^c\mathbf{V}_m)_1 \sum_{i=k}^{k+j-1} \dot{x}_i + ({}^c\mathbf{V}_m)_2 \sum_{i=k}^{k+j-1} \dot{y}_i + ... + ({}^c\mathbf{V}_m)_6 \sum_{i=k}^{k+j-1} \dot{\theta}_i \right]. \tag{16}$$

It is noticeable that, locally, close to time k, the evolution of the image features is *linear* with respect to the CoM velocity. We suppose that the CoM only translates in x and y axis and the trunk can rotate around the z axis: $\dot{z}_i = \dot{\phi}_i = \dot{\psi}_i = 0$. Now, consider the distinct features, i.e. the row l of $\mathbf{s}$, and stack all l's within the window horizon as $\mathbf{S}_{l,k+1} \stackrel{\text{def}}{=} (\mathbf{s}_{l,k+1} \dots \mathbf{s}_{l,k+N})^T$.

Since each velocity component can be expressed as a linear combination of their corresponding jerks values applied from k to i, $\mathbf{S}_{l,k+1}$ can finally be expressed as a *linear* function of $\dot{\hat{x}}_k, \dot{\hat{y}}_k, \dot{\hat{\theta}}_k$ (*constant* in our problem) and $\dddot{X}_k, \dddot{Y}_k, \dddot{\Theta}_k$ (*variables* of our problem).

Then, we can express the problem of determining the trajectory of the CoM such that the visual objective of the Image-Based Visual Servoing can be reached, with the same objective function as in Eq. 13. Again, direct optimization of the jerk values for x, y, and θ would lead to introduce nonlinearities in the QP formulation (from the constraints involving θ). Hence, we apply a two-step strategy as in the PBVS case. The first stage uses optimization over the jerks in x, y, θ but without considering the constraints on x, and y, the idea being to provide a reference trajectory for the angles. This first optimization problem to solve is then reduced to

$$\min_{\dddot{\Theta}_k, \dddot{\Theta}_k^f} \quad \frac{\gamma}{2}\sum_{l=0}^{M} [\mathbf{S}_l^d - \mathbf{S}_{l,k+1}]^\top \mathbf{W} [\mathbf{S}_l^d - \mathbf{S}_{l,k+1}] + \frac{\alpha}{2}\left\|\dddot{\Theta}_k\right\|^2 + \frac{\alpha}{2}\left\|\dddot{\Theta}_k^f\right\|^2 + \frac{\beta}{2}\left\|\dddot{X}_k\right\|^2 + \frac{\beta}{2}\left\|\dddot{Y}_k\right\|^2, \tag{17}$$

that is transformed into a QP thanks to Eq. 16.

In the second stage, we take the θ trajectories for the CoM and the feet as reconstructed in the first stage, and then optimize over the x, y translation jerks and over the feet positions. The angular velocity is taken as constant,

$$\min_{U_k} \quad \frac{\alpha}{2}\left\|\dddot{X}_k\right\|^2 + \frac{\alpha}{2}\left\|\dddot{Y}_k\right\|^2 + \frac{\beta}{2}\left\|Z_{k+1}^x - Z_{k+1}^{x_{ref}}\right\|^2 + \frac{\beta}{2}\left\|Z_{k+1}^y - Z_{k+1}^{y_{ref}}\right\|^2 + \frac{\gamma}{2}\sum_{l=0}^{M} [\mathbf{S}_l^d - \mathbf{S}_{l,k+1}]^\top \mathbf{W} [\mathbf{S}_l^d - \mathbf{S}_{l,k+1}],$$

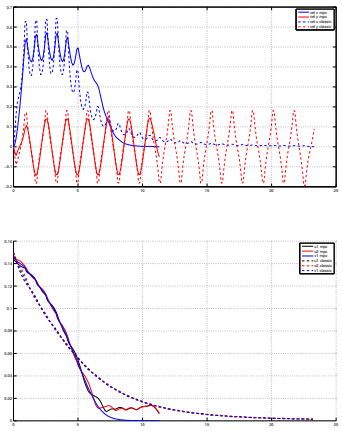

Fig. 4. Top, comparison of velocity profiles in the x (blue) and y (red) axes, for the approach of [3] (dashed line) and ours (solid line). Bottom, comparison of the features errors evolution.

with the predicted sequence of features from Eq. 16.

V. Results and Discussion

Hereafter, we present results of our approach, in a simulated environment, with the HRP2 robot model. We assume that no noise or modeling errors have been introduced. For all the tests (PBVS and IBVS), the initial position is $(0, 0)$ and the visual target, made of four points, is at $(5, 0)$. The figure 2 gives three trajectories realized by following our control scheme, with different goals and execution schemes. The left column corresponds to a PBVS one, the center and right columns to IBVS schemes. The first line gives the footsteps sequence, the second the evolution of the visual features (red is the initial configuration, blue the final one). The third and four lines depict the velocities and errors profiles. One can observe that in all cases, the dynamical balance is kept, that the visual reference is reached (more precisely in the case of PBVS than IBVS) and that most of the correction related to the rotation is done at the beginning. This is visible in the rightmost experiment, where one can distinguish three phases: orientation compensation (in site rotation), scaling (forward motion), and sideways motion along an arc of circle to reduce the residual visual errors.

To focus on the PBVS, the performance of the linearization done to maintain the QP form (see Section III) depends of the distance traveled inside the horizon, and on the velocity of the robot and the size of the horizon. In Fig. 5, we can see the linearized and real features trajectories for a given CoM trajectory and higher-than-usual velocities. Close to the beginning (linearization point), trajectories are close, and then differ. This example is extreme, since usual displacements are much smaller. When horizon displacements get bigger (far from the target), the robot just needs a tendency. When smaller, the linearization errors becomes negligible.

Also, in Fig. 3, we see that it is robust to perturbations: A trajectory is followed with PBVS to reach $(2, 0.1)$, but a strong perturbation in the CoM position is introduced (close to $(0.5, 0.2)$). This perturbation is recovered quasi-instantly.

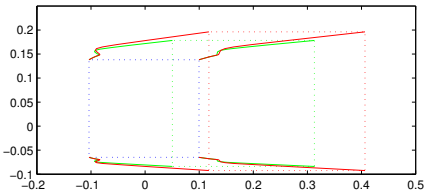

Fig. 5. Trajectory of the visual features in one iteration of the QP, and evolution of the features obtained by the linearization model (green lines) and by the non-linear model (red lines).

Due to stepping, features positions oscillate. One of the main advantages of using MPC is that it naturally filters these oscillations. It is remarkable that, in comparison with [3], we do not need to model explicitly the sway motion of the robot and the resulting motion of the features. The system does oscillate inside the horizon, but at the end, the optimal control is taken without oscillations.

We conducted a simple experiment to illustrate better the differences between our approach with [3]. In this experiment, the robot simply goes three meters forward. In Fig. 4 left, we present the profiles of velocities and errors, in both cases. We can appreciate (right figure) that the coupled approach (solid line) converges faster. Moreover, one can observe that, close to the goal, the features positions follow a smoother trajectory in the case of the visual predictive control than with the decoupled approach, where oscillations (as a result of the immediate stepping) are visible. Also, when the error gets small, [3] slows down to have a slow convergence rate. This is a normal feature, i.e. the error evolves with an exponential decay. With our approach, this is less the case, because we take into account future information. In the errors evolution we should note that the u components in the image plane are theoretically the oscillatory ones (from the stepping motion). In Fig. 4, we can see that the v component converges faster in the coupled approach. With the u components, there remains a small residual of the oscillation in all approaches.

VI. Conclusions

In this paper, we have proposed a novel approach to close the control loop by introducing visual information into the humanoid walking pattern generation. Our online pattern generator integrates the regulation of the relative pose of 3D (PBVS) or 2D (IBVS) image features while simultaneously ensuring safety and stability. In the first case, in order to keep the optimization formulation as a QP, the perspective projection equations are linearized around the features positions at the beginning of each cycle. As an ongoing work, we plan to perform real experiments on the HRP-2 humanoid platform.

Acknowledgments

M. Garcia is supported by grant 263150 from the Mexican National Council of Science and Technology (CONACYT).

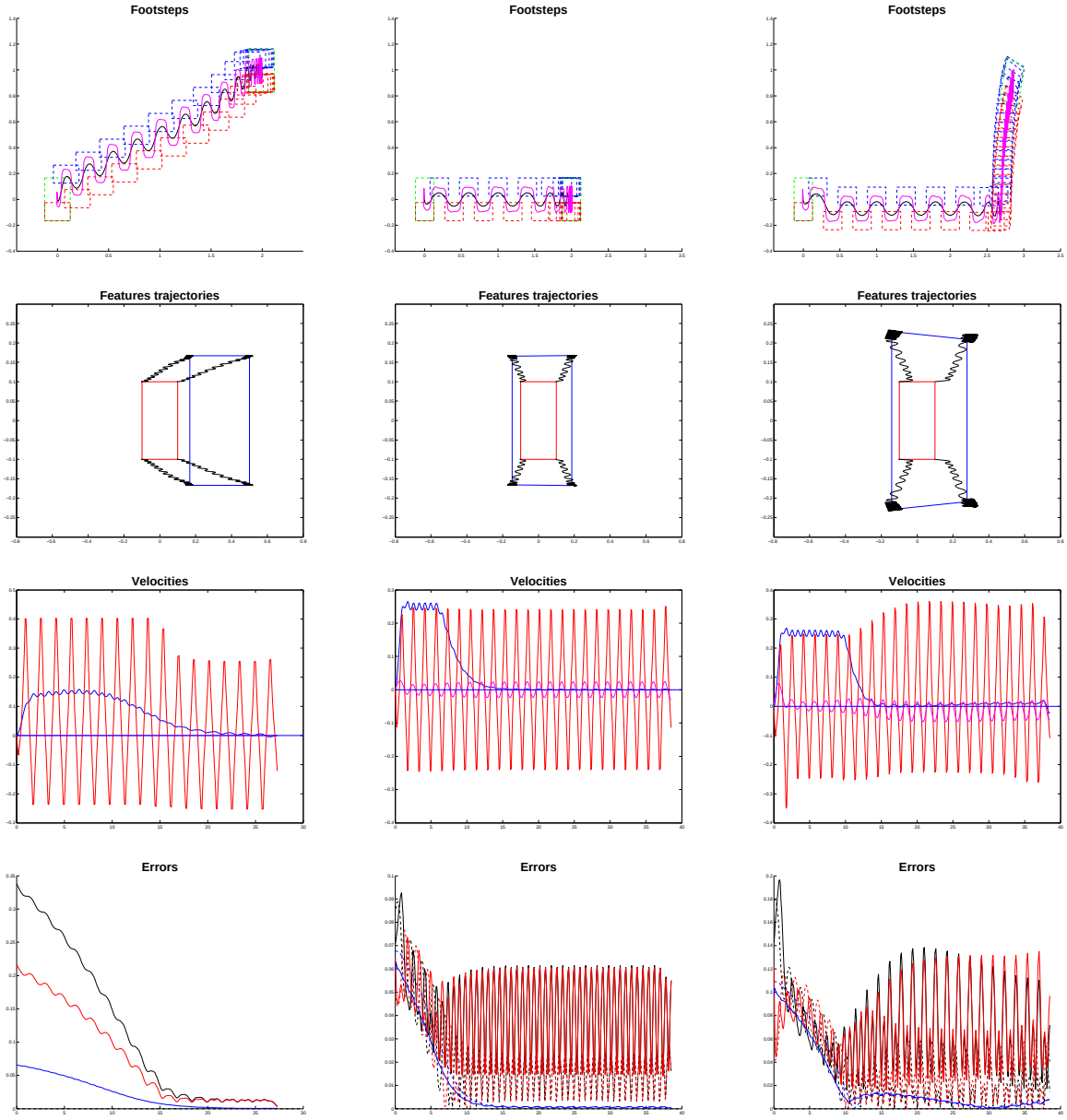

Fig. 2. PBVS (leftmost) and IBVS (center and rightmost) experiments for the humanoid robot reactive walking.

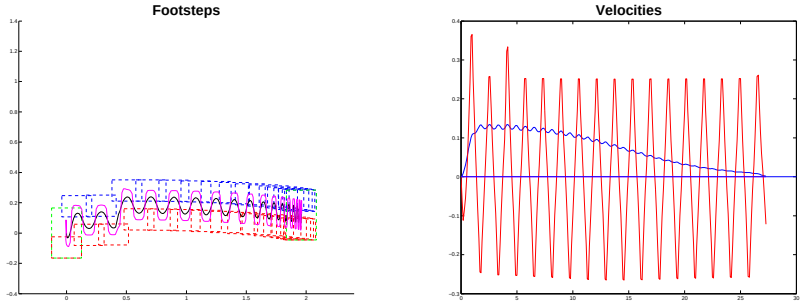

Fig. 3. Perturbation on a PBVS experiment for the humanoid robot reactive walking. A brutal pushing of the robot occurs close to $(0.5, 0.2)$.

REFERENCES

[1] G. Allibert, E. Courtial, and F. Chaumette. Visual servoing via nonlinear predictive control. In G. Chesi and K. Hashimoto, editors, *Visual Servoing via Advanced Numerical Methods*, volume 401 of *Lecture Notes in Control and Information Sciences*, pages 375–393. Springer London, 2010.

[2] J.E Chestnutt, P. Michel, J.J. Kuffner, and T. Kanade. Locomotion among dynamic obstacles for the honda asimo. In *IROS*, pages 2572–2573, 2007.

[3] C. Dune, A. Herdt, O. Stasse, P. B Wieber, K. Yokoi, and E. Yoshida. Cancelling the sway motion of dynamic walking in visual servoing. In *IROS*, pages 3175–3180, 2010.

[4] A. Faragasso, G. Oriolo, A. Paolillo, and M. Vendittelli. Vision-based corridor navigation for humanoid robots. In *ICRA*, 2013.

[5] J.-S. Gutmann, M. Fukuchi, and M. Fujita. 3d perception and environment map generation for humanoid robot navigation. *International Journal of Robotics Research*, 27(10):1117–1134, 2008.

[6] A. Herdt, H. Diedam, P.-B. Wieber, D. Dimitrov, K. Mombaur, and M. Diehl. Online walking motion generation with automatic footstep placement. *Advanced Robotics*, 24(5-6):719–737, 2010.

[7] 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*, volume 2, pages 1620 – 1626 vol.2, sept. 2003.

[8] J.-Y. Kim, I.-W. Park, J. Lee, and J.-H. Oh. Experiments of vision guided walking of humanoid robot, khr-2. In *Humanoids*, 2005.

[9] O. Lorch, A. Albert, J. Denk, M. Gerecke, R. Cupec, J. Seara, W. Gerth, and G. Schmidt. Experiments in vision-guided biped walking. In *IROS*, pages 2484–2490, 2002.

[10] D. Maier, A. Hornung, and M. Bennewitz. Real-time navigation in 3d environments based on depth camera data. In *Humanoids*, 2012.

[11] P. Michel, J. Chestnutt, S. Kagami, K. Nishiwaki, J. Kuffner, and T. Kanade. Gpu-accelerated real-time 3d tracking for humanoid locomotion and stair climbing. In *IROS*, pages 463–469, October 2007.