ZMP-BASED HUMANOID WALKING ENGINE WITH ARMS MOVEMENT AND STABILIZATION

Marcos R. O. A. Maximo*, Carlos H. C. Ribeiro*

*Autonomous Computational Systems Lab (LAB-SCA),
Computer Science Division, Aeronautics Institute of Technology
Praça Marechal Eduardo Gomes, 50, Vila das Acácias, 12228-900
São José dos Campos, SP, Brazil

Emails: mmaximo@ita.br, carlos@ita.br

Abstract— In this paper, we present an omnidirectional walking engine for a humanoid robot. Our approach is based on the Zero Moment Point (ZMP) concept, which provides an useful criterion for biped stability. To avoid dealing directly with the complex dynamics of a high degrees of freedom humanoid robot, we used the 3D Linear Inverted Pendulum Model (3D-LIPM) to approximate the robot dynamics. 3D-LIPM dictates that the center of mass (CoM) and ZMP dynamics are related by uncoupled second order ordinary differential equations. Thus, we may compute a suitable CoM trajectory to maintain the robot balance by solving a boundary value problem analytically. Furthermore, we employed strategies to improve the walking robustness, namely, we make the robot move its arms in order to compensate the yaw moment induced by the legs and we developed a feedback controller that uses the torso angular velocities to stabilize the walk. Simulated and real experiments on a custom enhanced Bioloid robot are shown to validate the proposed methods.

Keywords— Robotics, Biped Locomotion, Humanoid Robots.

Resumo— Neste artigo, apresenta-se uma caminhada omnidirecional para um robô humanoide. Nossa abordagem é baseada no conceito de *Zero Moment Point* (ZMP), que provê um critério útil para estabilidade bípede. Para evitar lidar diretamente com a dinâmica complexa de um robô humanoide com muitos graus de liberdade, usa-se o *3D Linear Inverted Pendulum Model* (3D-LIPM) para aproximar a dinâmica do robô. O 3D-LIPM dita que as dinâmicas do centro de massa (CM) e do ZMP estão relacionadas por equações diferenciais ordinárias de segunda ordem desacopladas. Então, pode-se calcular uma trajetória para o CM adequada para manter o equilíbrio do robô através da solução analítica de um problema de valor de contorno. Além disso, usa-se estratégias para melhorar a robustez da caminhada, a saber, faz-se o robô mover seus braços para compensar o momento induzido em guinada pelas pernas e foi desenvolvido um controlador que usa as velocidades angulares do torso para estabilizar a caminhada. Experimentos simulados e reais em um robô Bioloid incrementado são apresentados para validar os métodos propostos.

Palavras-chave— Robótica, Locomoção Bípede, Robôs Humanoides.

1 Introduction

Humanoid robots are claimed to be more adequate than wheeled ones for locomotion in unstructured human environments that include floor discontinuities, ladders, rocks, debris, etc. However, walk control for humanoid robots (biped locomotion) involves characteristics acknowledged for making a control problem hard, such as high non-linearity, underactuation, and many degrees of freedom. Current state-of-the-art humanoid robots are able to achieve high speeds on flat ground; nevertheless, they are still outmatched by humans in terms of robustness and energy efficiency (Collins et al., 2005). Therefore, humanoid walking is still considered an unsolved problem in Robotics.

Biped locomotion methods can be grouped into two main approaches: model-based and model-free. Model-free methods try to create a controller directly, without a mathematical model for the dynamics of the robot.On the other hand, model-based methods depend on mathematical models of the robot's dynamics. Due to the high dimensionality of a high degrees of freedom humanoid robot, reduced models that focus on the center of mass (CoM) dynamics are often used, such as the linear inverted pendulum (Kajita et al., 2001) and centroidal dynamics (Orin et al., 2013) models. Then, a controller is developed for the model assuming that the robot's dynamics follows the model. This approach has produced good practical results, such as the walking performance achieved by the Honda ASIMO robot (Takenaka et al., 2009). Nevertheless, the best model-based walks still operate in a conservative way and produces inefficient motion as far as energy comsumption is concerned (Collins et al., 2005).

A popular concept in the humanoid walking literature is the Zero Moment Point (ZMP) (Vukobratović and Borovac, 2004). The ZMP is often called the dynamic version of the CoM due to the following stability criterion: a biped is dynamically stable at a given time instant if the ZMP is inside the support polygon (the convex hull of all contact points on the ground) (Vukobratović and Borovac, 2004). Therefore, a common strategy is to generate a gait that maintains the ZMP inside the support polygon at all times (Takenaka et al., 2009).

The main contribution of this work is an

omnidirectional walking engine that shows stable walking both in simulation and in a real robot. Our approach is based on the ZMP concept and uses the 3D Linear Inverted Pendulum Model (3D-LIPM) to approximate the robot's dynamics. The ZMP equations resulting from the 3D-LIPM are simple enough to allow us to compute a suitable center of mass (CoM) trajectory to maintain the robot balance by solving a boundary value problem analytically. Moreover, the work contributes with simple strategies involving arms movement and a stabilization controller to improve the walking robustness against disturbances.

The remaining of this paper is organized as follows. Section 2 presents the linear inverted pendulum model. In Section 3, we explain our walking engine. Section 4 shows simulated and real experiments to validate the methods developed in this work. Finally, Section 5 concludes and shares our ideas for future work.

2 3D Linear Inverted Pendulum Model (3D-LIPM)

In the 3D-LIPM (Kajita et al., 2001), the complex multibody dynamics of a humanoid robot is reduced to an inverted pendulum where the pendulum mass represents the robot's CoM. Additionally, Kajita et al. demonstrated that if the pendulum mass is constrained to move along a horizontal plane, then the resulting ZMP dynamics is given by the following uncoupled linear differential equations:

$$x_{ZMP} = x_{CoM} - \frac{z_{CoM}}{g}\ddot{x}_{CoM} \tag{1}$$

$$y_{ZMP} = y_{CoM} - \frac{z_{CoM}}{g}\ddot{y}_{CoM} \tag{2}$$

or, in vector form:

$$\mathbf{x}_{ZMP} = \mathbf{x}_{CoM} - \frac{z_{CoM}}{g}\ddot{\mathbf{x}}_{CoM} \tag{3}$$

where $\mathbf{x}_{ZMP} = [x_{ZMP}, y_{ZMP}]^T$ is the ZMP position, $\mathbf{x}_{CoM} = [x_{CoM}, y_{CoM}]^T$ is the CoM position, $\ddot{\mathbf{x}}_{CoM} = [\ddot{x}_{CoM}, \ddot{y}_{CoM}]^T$ is the CoM acceleration, z_{CoM} is the CoM height and g is the gravity acceleration. We refer the interested reader to (Wieber et al., 2015) for a proof that carefully discusses the assumptions and approximations involved in this model. Note that implementing a CoM trajectory planned using the 3D-LIPM in a real humanoid robot requires translating the CoM trajectory into joints trajectories through Inverse Kinematics (IK) (Kajita et al., 2014).

3 Walking Engine

The input to the walking engine is the desired velocity $\mathbf{v} = [v_x, v_y, v_\psi]^T$ with respect to the local coordinate system of the robot, where v_x, v_y, and v_ψ are speeds in forward, sideways and rotational directions, respectively. The output of the algorithm is a vector $\mathbf{q}$ containing the angular positions of the joints. Note that we assume position-controlled joints. If the joints are torque-controlled, the joints trajectories may be implemented using a position feedback loop for each joint.

The walking engine consists of a series of modules. First, at the beginning of each step, a footstep planner decides the torso and the swing foot poses that the robot must achieve at the end of the step. In order to achieve these planned poses while keeping the robot balance, the algorithm determines trajectories for the CoM and the swing foot. Then, IK is used to translate the CoM and swing foot poses into joints positions. Finally, a feedback controller modifies the joints positions to stabilize the walk.

Before a more detailed explanation, we need to clarify some design decisions and introduce notation. To be precise, we consider a step as a walk cycle with fixed duration T where the robot moves a single foot. The foot which is moved is called swing foot throughout the step, independently if it is on the ground or in the air. Each step begins and ends with double support phases of duration $T_{ds}/2$, so the double support ratio is $r_{ds} = T_{ds}/T$. In the middle of the step, there is a single support phase that begins at $t = t_b$ and ends at $t = t_e$, where t indicates time measured since the beginning of the step. Moreover, let us define a normalized time and time events:

$$\varphi = \frac{t}{T},\ \varphi_b = \frac{t_b}{T},\ \varphi_e = \frac{t_e}{T} \tag{4}$$

Since the CoM of a multibody robot depend on the relative body parts positions, we use a fixed position in the torso as an approximation for the CoM. In the following explanation, we call a pose a three-dimensional vector containing $x-y$ position and yaw orientation. Moreover, we use $\mathbf{p}_b[k] = [x_b[k], y_b[k], \psi_b[k]]^T$ and $\mathbf{x}_b[k] = [x_b[k], y_b[k]]^T$ to denote the pose and the position of body part b at the beginning of the time step k, respectively. Finally, we must add that the computations explained in subsections 3.1, 3.2, and 3.3 are carried out in a coordinate system at the center of the support foot with axes rotated to match the support foot's orientation.

3.1 Planning footsteps

To provide an omnidirectional abstraction, we want the torso to obey the following dynamical system:

$$\begin{bmatrix}\dot{x}(t)\\ \dot{y}(t)\\ \dot{\psi}(t)\end{bmatrix} = \begin{bmatrix}v_x(t)\cos\psi(t) - v_y(t)\sin\psi(t)\\ v_x(t)\sin\psi(t) + v_y(t)\cos\psi(t)\\ v_\psi(t)\end{bmatrix} \tag{5}$$

However, a torso trajectory that perfectly follows the omnidirectional model is unlikely to be stable, e.g. lateral oscillations are usually needed for stable forward walk. Thus, we try to match the omnidirectional model only at the beginning of the steps by considering a time discretization of Equation (5) with T as sample time.

Furthermore, the robot's geometry must be taken into account, since the physical dimensions of the legs limit foot reachability and self-collision between the body parts must be avoided. Then, at the beginning of a step k, the footstep planner uses heuristics to select torso and swing foot poses at the beginning of the next step that satisfies these geometric constraints while following the omnidirectional model as closely as possible. For details, please see (Maximo, 2015).

3.2 CoM Trajectory

We use a polygonal ZMP reference that keeps the ZMP at the center of the support foot during single support:

$$\mathbf{m}_{d1} = \frac{1}{T_{d1}} \left(\mathbf{x}_s[k] - \mathbf{x}_t[k] \right) \tag{6}$$

$$\mathbf{m}_{d2} = \frac{1}{T_{d2}} \left(\mathbf{x}_t[k+1] - \mathbf{x}_s[k] \right) \tag{7}$$

$$\mathbf{x}_{ZMP}(t) = \begin{cases} \mathbf{x}_t[k] + \mathbf{m}_{d1} t, t \in [0, t_b) \\ \mathbf{x}_s[k], t \in [t_b, t_e) \\ \mathbf{x}_s[k] + \mathbf{m}_{d2}(t - t_e), t \in [t_e, T] \end{cases} \tag{8}$$

where the indices t and s refer to the torso and the support foot, respectively, and k is the current step. We also want the CoM trajectory to match the CoM positions at the beginning and at the end of the step:

$$\mathbf{x}_{CoM}(0) = \mathbf{x}_t[k] \tag{9}$$

$$\mathbf{x}_{CoM}(T) = \mathbf{x}_t[k+1] \tag{10}$$

Equations (3), (8), (9), and (10) define a boundary value problem whose solution is (Yi et al., 2011):

$$\mathbf{x}_{CoM}(t) = \mathbf{x}_1(t) + \mathbf{x}_2(t) \tag{11}$$

where:

$$\mathbf{x}_1(t) = \mathbf{x}_{ZMP}(t) + \mathbf{c}_1 e^{\lambda t} + \mathbf{c}_2 e^{-\lambda t} \tag{12}$$

$$\mathbf{x}_2(t) = \begin{cases} -\frac{1}{\lambda}\mathbf{m}_{d1} \sinh\left(\lambda\left(t - t_b\right)\right), t \in [0, t_b) \\ 0, t \in [t_b, t_e) \\ -\frac{1}{\lambda}\mathbf{m}_{d2} \sinh\left(\lambda\left(t - t_e\right)\right), t \in [t_e, T] \end{cases} \tag{13}$$

$$\lambda = \sqrt{\frac{g}{z_{CoM}}} \tag{14}$$

$$\mathbf{k}_{d1} = \frac{1}{\lambda}\mathbf{m}_{d1} \sinh\left(-\lambda t_b\right) \tag{15}$$

$$\mathbf{k}_{d2} = \frac{1}{\lambda}\mathbf{m}_{d2} \sinh\left(\lambda\left(T - t_e\right)\right) \tag{16}$$

$$\mathbf{c}_1 = \frac{1}{e^{\lambda T} - e^{-\lambda T}} \left(\mathbf{k}_{d2} - \mathbf{k}_{d1} e^{-\lambda T} \right) \tag{17}$$

$$\mathbf{c}_2 = \frac{1}{e^{\lambda T} - e^{-\lambda T}} \left(\mathbf{k}_{d1} e^{\lambda T} - \mathbf{k}_{d2} \right) \tag{18}$$

Therefore, the CoM trajectory in the x and y directions is given by Equation (11). The torso height must be kept constant at z_{CoM}, as required by the 3D-LIPM. A trajectory for the torso in orientation is still missing. To solve this, we interpolate between $\psi_t[k]$ and and $\psi_t[k+1]$ using a interpolation function $h\left(\varphi\right)$:

$$\psi_t(\varphi) = h(\varphi)\psi_t[k] + (1 - h(\varphi))\psi_t[k+1] \tag{19}$$

We want the trajectory function in Equation (19) to be of class $C^1([0,1))$ (i.e. continuous up to its first derivative) to make it more adequate for position-tracking joints. Furthermore, we do not want to rotate the torso during double support to avoid violating the constraints imposed by the closed kinematic chain. Therefore, inspired by (Graf et al., 2009), we choose:

$$h(\varphi) = \begin{cases} 0, \varphi \in [0, \varphi_b) \\ 0.5\left(1 - \cos\left(\pi \frac{\varphi - \varphi_b}{\varphi_e - \varphi_b}\right)\right), \varphi \in [\varphi_b, \varphi_e) \\ 1, \varphi \in [\varphi_e, 1] \end{cases} \tag{20}$$

3.3 Swing Foot Trajectory

Within a step, the swing foot pose $\mathbf{p}_a(\varphi)$ is obtained interpolating between $\mathbf{p}_a[k]$ and $\mathbf{p}_a[k+1]$:

$$\mathbf{p}_a(\varphi) = h(\varphi)\mathbf{p}_a[k+1] + (1 - h(\varphi))\mathbf{p}_a[k] \tag{21}$$

A more general solution may use different interpolation functions for different directions. However, applying $h(\varphi)$ to all directions generate an adequate solution. Using $h(\varphi)$ also brings other interesting effects: $\mathbf{p}_a$ does not change during double support; and $\dot{\mathbf{p}}_a\left(\varphi_b\right) = \dot{\mathbf{p}}_a\left(\varphi_e\right) = 0$, thus the foot leaves and touches the ground smoothly.

For vertical motion, we consider a maximum step height z_{step} and compute the z position of the swing foot $z_a(\varphi)$ using:

$$z_a(\varphi) = z_{step} v(\varphi) \tag{22}$$

where:

$$v(\varphi) = \begin{cases} 0, \varphi \in [0, \varphi_b) \cup [\varphi_e, 1] \\ 0.5\left(1 - \cos\left(2\pi \frac{\varphi - \varphi_b}{\varphi_e - \varphi_b}\right)\right), \varphi \in [\varphi_b, \varphi_e) \end{cases} \tag{23}$$

The function $z_a(\varphi)$ satisfies desired properties for this vertical trajectory: it is of class $C^1\left([0,1]\right)$; and we also have $\dot{z}_a(\varphi_b) = \dot{z}_a(\varphi_e) = 0$, thus the foot leaves and touches the ground with zero speed, which helps to reduce the impact.

3.4 Computing Joints Trajectories

In the previous sections, 3D positions and yaw angles for the torso and the swing foot were computed. To determine the remaining degrees of freedom (roll and pitch), we constrain the torso to be upright without inclination, and the swing foot to be parallel to the ground. Then, joints angles are computed through an analytic IK solution, which was derived using geometry and trigonometry. The robot model considered has six joints for each leg: hip-roll, hip-pitch, knee, foot-pitch, and foot-roll. For details, see (Maximo, 2015).

3.5 Adding Arms Movement

To compensate the yaw moment induced by the legs and avoid slipping, the robot moves its arms, as a human does. This is implemented by adding a x displacement to each arm based on the x displacement of the opposing leg:

$$x_{larm} = sx_r \tag{24}$$

$$x_{rarm} = sx_l \tag{25}$$

where x_{larm} and x_{rarm} are the left and right arms displacements, respectively, and the parameter s adjusts how much of the leg displacement is added to the opposing arm.

3.6 Stabilization Controller

Constraining the robot torso to be upright with no inclination yields that the angular velocities in roll and pitch should be zero. However, angular velocities appear while the robot is walking due to model imperfection and disturbances. Therefore, a stabilization controller computes displacements for the legs' joints using angular speeds as feedback:

$$\Delta q_{hr} = K_{hr}\omega_r \tag{26}$$

$$\Delta q_{hp} = K_{hp}\omega_p \tag{27}$$

$$\Delta q_{k} = K_{k}\omega_p \tag{28}$$

$$\Delta q_{fp} = K_{fp}\omega_p \tag{29}$$

$$\Delta q_{fr} = K_{fr}\omega_r \tag{30}$$

where ω_r and ω_p are the angular speeds in roll and pitch, respectively, and the indices hr, hp, k, fp, and fr stand for hip-roll, hip-pitch, knee, foot-pitch, and foot-roll, respectively. Moreover, Δq_* and K_* are joints displacements and controller gains, respectively. These displacements are added to the angles computed by IK.

4 Experiments

The robot used for the experiments is a humanoid robot based on the Bioloid Comprehensive Kit (Robotis, 2006). This robot has 18 degrees of freedom: 3 per arm (shoulder-yaw, shoulder-pitch and elbow-yaw) and 6 per leg (hip-roll, hip-pitch, knee, foot-pitch, and foot-roll).

4.1 Simulation

We built a simulation model for the Bioloid humanoid robot in the Virtual Robot Experimentation Platform (V-REP) (Coppelia Robotics, 2014) using the data provided by (Teodoro, 2007). The values used to configure the walking engine's parameters are shown in Table 1.

Param.	Meaning	Value
T	Step duration	0.5 s
z_{CoM}	Height of the CoM	0.23 m
z_{step}	Max. swing foot height	0.04 m
r_{ds}	Double support ratio	0.2
y_{sep}	Half of min. y feet separ.	0.04 m
s	Arms movement factor	0.8

Table 1: Values used to configure the walking engine's parameters in simulation.

Comprehensive testing results of the walking engine under many different velocity commands to provide an adequate validation of its omnidirectional capabilities cannot be shown here due to space restrictions. Therefore, we present an experiment where the robot was commanded to move in forward and turning directions simultaneously with velocity $[0.1\ m/s, 0\ m/s, \pi/9\ rad/s]^T$. We estimate the ZMP by computing the center of pressure (CoP) as a weighted average of the contact points positions (Kajita et al., 2014):

$$\mathbf{x}_{CoP} = \frac{\sum_{i=1}^{N_c} \mathbf{x}_{c,i} f_{z,i}}{\sum_{i=1}^{N_c} f_{z,i}} \tag{31}$$

where $\mathbf{x}_{CoP}$ is the CoP position, $\mathbf{x}_{c,i}$ is the position of the contact point i, $f_{z,i}$ is the z component of the ground reaction force acting on the contact point i, and N_c is the number of contact points. Regarding implementation, information about the contact points are obtained through the V-REP API. Moreover, the stabilization was deactivated for this experiment and the robot needs 1 second to get to the starting walking posture. Figure 1 shows the results. In Figure 1(a), modeling error and actuator dynamics make the CoM trajectory erratic and the CoP very noisy, while footsteps positions appear blurry due to slippage. Moreover, undesired roll and pitch oscillations may be seen in Figure 1(b). In Figure 1(b), the robot rotates about 70° in yaw after 3.5 s of effective motion, thus it achieves a rotational speed close to the requested one of $20°/s = \pi/9\ rad/s$.

To validate the stabilization controller, experiments were done where disturbing forces are exerted to the robot's torso during small time intervals: $F_1 = [0, 5, 0]^T\ N$ acts from 0 to 0.1 seconds and force $F_2 = [10, 5, 0]^T\ N$ acts from 3 to 3.05 seconds. F_1 tests the controller while the robot is at rest, and F_2 disturbs the robot during walking.

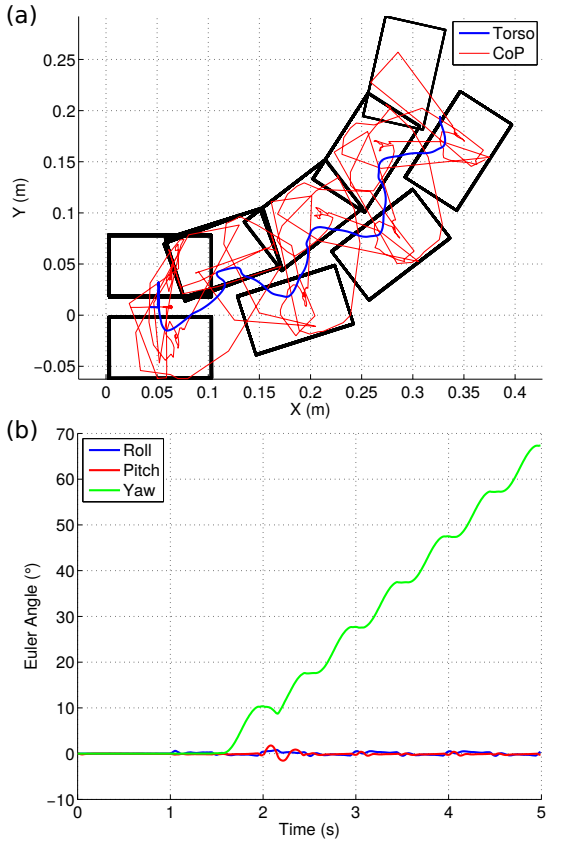

Figure 1: Turning Walk experiment: (a) $x - y$ view showing the torso (CoM approximation) trajectory (blue), the CoP trajectory (red), and footsteps positions (black) and (b) Euler angles.

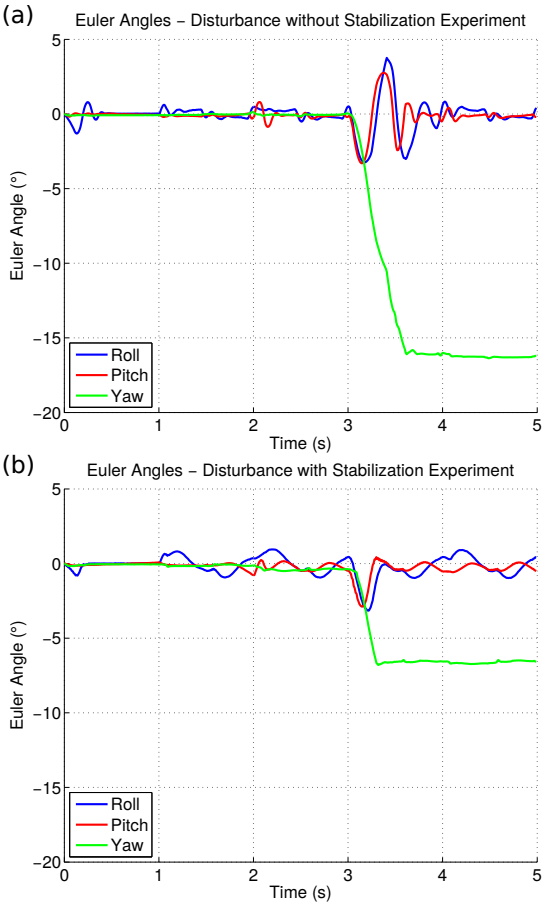

Figure 2: Experiments to validate the stabilization controller: (a) results without stabilization and (b) results with stabilization.

In these experiments, the robot commanded velocity is $[0.1\ m/s, 0\ m/s, 0\ rad/s]^T$. Figure 2 shows the results for the disturbance experiments considering two cases: (a) controller deactivated, and (b) controller activated. In Figure 2(a), we see oscillations in roll in the beginning due to F_1 and strong oscillations in the three Euler angles after 3 seconds due to F_2. If the controller is activated, the oscillations are damped much faster (qualitatively, a single period of oscillation is clearly visible), as shown in Figure 2(b). Furthermore, the undesired deviation produced in yaw by F_2 is reduced to approximately 40% of the original value by the stabilization. The following gains were used: $K_{hr} = 0.01\ s^{-1}$, $K_{hp} = 0.01\ s^{-1}$, $K_k = 0.01\ s^{-1}$, $K_{fp} = 0.02\ s^{-1}$, and $K_{fr} = 0.05\ s^{-1}$.

4.2 Real Robot

We custom enhanced the original Bioloid humanoid robot by adding a 1 GHz dual-core single board computer, a camera, an IMU, a subcontroller and voltage regulators. We increased the parameter y_{sep} to 0.05 m to avoid feet collision and we added offsets to the legs' pitch joints to compensate the high backlash and flexibility of the AX-12 servomotors used by the Bioloid kit.

Figure 3 presents an image sequence of our humanoid robot executing a forward walk with 0.1 m/s. The frame rate is 10 fps. Frontal and sideways walking patterns executed by the real robot can be watched at `https://www.itandroids.com.br/publications`. Arms movement and stabilization were not active during these experiments.

5 Conclusions and Future Work

This paper presented an omnidirectional walking engine where the input to the algorithm is a velocity vector with desired speeds in forward, sideways and rotational directions. First, a footstep planner computes torso and the swing foot poses at the beginning of the next step. Then, a boundary value problem is solved analytically to compute a stable CoM trajectory, while the remaining CoM and swing foot degrees of freedom are determined by interpolation. Finally, IK is used to compute the joints positions.

Executing the gait pattern in an open loop fashion yields stable walking. However, we developed strategies to improve the algorithm performance: arms movement to compensate induced yaw moment and a stabilization controller to

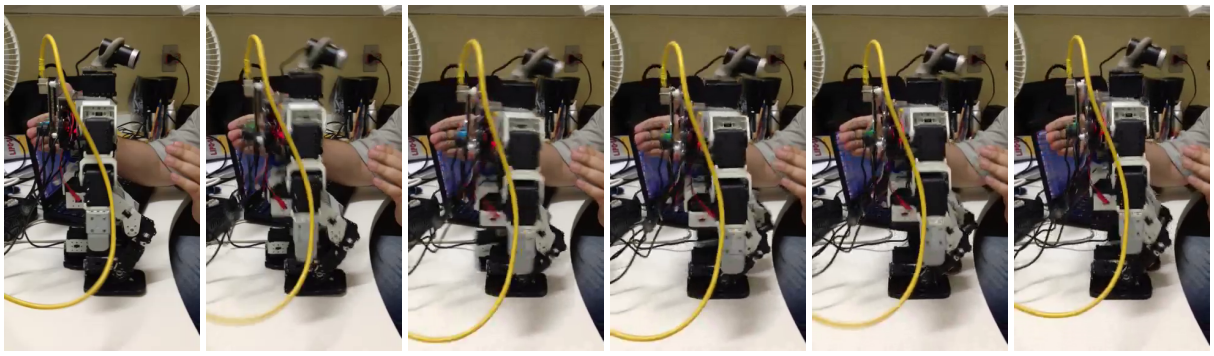

Figure 3: Real robot executing a forward walking of 0.1 m/s. The frame rate is 10 fps.

counter disturbances. Finally, experiments were executed in simulated and real environments to validate the walking engine.

This work may be extended in many directions. The stabilization controller may be improved by adding feedback from the roll and pitch angles of the torso. Another path involves using Model Predictive Control (MPC) to directly reason about the dynamical system constraints (Herdt et al., 2010).

Acknowledgments

Marcos R. O. A. Maximo thanks the students Davi Barroso, Luis Aguiar, Samuel Pinto, and Téssio Arruda for helping with the robot hardware used in the experiments. Carlos H. C. Ribeiro thanks CNPq (Grant no. 303738/2013-8).

References

Collins, S., Ruina, A., Tedrake, R. and Wisse, M. (2005). Efficient Bipedal Robots Based on Passive-Dynamic Walkers, *Science* **307**(5712): 1082–1085.

Coppelia Robotics (2014). Virtual Robot Experimentation Platform USER MANUAL, `http://www.coppeliarobotics.com/helpFiles/`. [Online; accessed at November 10, 2014].

Graf, C., Härtl, A., Röfer, T. and Laue, T. (2009). A Robust Closed-Loop Gait for the Standard Platform League Humanoid, *Proceedings of the Fourth Workshop on Humanoid Soccer Robots*.

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

Kajita, S., Hirukawa, H., Harada, K. and Yokoi, K. (2014). *Introduction to Humanoid Robotics*, Springer.

Kajita, S., Kanehiro, F., Kaneko, K., Yokoi, K. and Hirukawa, H. (2001). The 3D Linear Inverted Pendulum Mode: a Simple Modeling for a Biped Walking Pattern Generator, *Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 239–246.

Maximo, M. R. O. A. (2015). *Omnidirectional ZMP-Based Walking for a Humanoid Robot*, Master's thesis, Aeronautics Institute of Technology.

Orin, D. E., Goswami, A. and Lee, S.-H. (2013). Centroidal Dynamics of a Humanoid Robot, *Autonomous Robots* **35**(2): 161 – 167.

Robotis (2006). Bioloid User's Guide.

Takenaka, T., Matsumoto, T. and Yoshiike, T. (2009). Real time motion generation and control for biped robot -1st report: Walking gait pattern generation, *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Teodoro, P. D. D. (2007). *Humanoid Robot: Development of a Simulation Environment of an Entertainment Humanoid Robot*, Master's thesis, Instituto Superior Técnico.

Vukobratović, M. and Borovac, B. (2004). Zero-moment point – thirty five years of its life, *International Journal of Humanoid Robotics* **1**(01): 157–173.

Wieber, P.-B., Tedrake, R. and Kuindersma, S. (2015). Modeling and control of legged robots, *in* B. Siciliano and O. Khatib (eds), *Springer Handbook of Robotics, 2nd Ed*, Springer, chapter 48.

Yi, S.-J., Zhang, B.-T., Hong, D. and Lee, D. D. (2011). Online learning of a full body push recovery controller for omnidirectional walking, *Proceedings of the 11th IEEE-RAS International Conference on Humanoid Robots*.