The 6th Workshop on Humanoid Soccer Robots @ Humanoids Conference, Bled (Slovenia), October 2011.

A Closed-Loop Dribbling Gait For The Standard Platform League

J.J. Alcaraz-Jiménez
University of Murcia, Spain
E-mail: juanjoalcaraz@um.es

D. Herrero-Pérez
Polytechnic University of Cartagena, Spain
E-mail: david.herrero@upct.es

H. Martínez-Barberá
University of Murcia, Spain
E-mail: humberto@um.es

***Abstract*—When it comes to human soccer, the players who have good dribbling skills are very appreciated, since they have a strong impact in the result of the game. In this work we will expose an approach to incorporate dribbling as part of a closed loop ZMP-based gait. Information about the underlying theoretical models of the gait as well as reference values for the system settings are provided.**

***Index Terms*—RoboCup, biped walking, dribbling.**

I. Introduction

In order to foster research in robotics related areas, the RoboCup federation organizes annually a robotic soccer world championship, as well as a number of regional competitions. The gait described here is part of the developments of the L3M team, which participates in the Standard Platform League (SPL) of the RoboCup competitions. In that league, all teams must compete with the same hardware. Therefore, their research activities focus on the development of software algorithms. Since 2008, the common robotic platform employed in the SPL is the humanoid robot Nao [1], developed by the French company Aldebaran Robotics. Nao is 56 cm tall and weights 4.8 Kg. In the RoboCup version of the robot, it has 21 DoF (5 in each leg, 1 shared by booth legs, 4 in each arm and 2 in the neck).

Several locomotion systems have been developed by the SPL teams. The teams that do not have their own locomotion module usually employ the gait provided by the robot manufacturers, developed by Gouaillier *et al.* [2]. However, some teams have designed faster gaits. The fastest speed till the moment has been reported by the Nao Devils team, which reaches 44 cm/s with a walk engine developed by Czarnetzki *et al.* [3]. On the other hand, at lower speeds Graf *et al.* [4] and Xue *et al.* [5] designed more robust and not so aggressive with the hardware gaits.

The gaits from Gouaillier *et al.*, Czarnetzki *et al.* and Xue *et al.* generate the positions of the Center of Mass (CoM) from a reference trajectory of the Zero Moment Point (ZMP) [6] which is always kept within the boundaries of the support polygon. In another way, Graf *et al.* employ a gait based on the 3D-LIMP model, initially proposed in [7]. All these gaits performed quite well in the RoboCup 2011 Championship, and the teams that developed their own systems ended up in the first places.

In addition, most of the teams have developed different types of kicks that are executed independently of the gait. In [8] Müller *et al.* showed an approach to design kicks and to ensure the stability of the robot during the kick stage.

In this work, we will show a system that allows the integration of the kicks in the walking engine. The step duration is dynamically adjusted to cover the needs of the kick. Moreover we will see how to define sequences of kicks and steps to create more complex actions such as dribblings.

Dribbling is important in robotic soccer because it allows the robot to avoid other players without loosing the control of the ball. Although it has already been studied in other leagues [9][10], no dribbling approach has been published, to the best of our knowledge, for humanoids robots.

The integration of the kicks in the gait improves the agility of the robot during the game, since the total duration of the kick or dribbling action is reduced. Furthermore, the kick designer is freed from the task of handling the balance of the robot, because it is controlled by the closed-loop gait.

First, we will describe the walking engine in section II and the approach employed to ensure the balance of the robot in section III. In section IV we will explain the configuration of the dribbling actions and the procedure to integrate them in the gait, while in section V we will analyze a real dribbling experiment. Finally, section VI is dedicated to reflect on the performance of the system and to outline future improvements.

II. Walk Engine

To achieve successful soccer playing, at least the problems of perception, behavior and locomotion must be addressed. First, the robot will employ its sensors to provide the behavior module with the information about the state of the environment. Then, the behavior module will reason about its current state and produce a target speed vector for the robot. Finally, the locomotion module will generate the sequence of values for the joint actuators that will make the robot move at that target speed. The middleware software layer developed by the manufacturers of the robot manages sensor readings and low-level motor signals.

The locomotion module is sketched in Fig. 1. The input received from the behaviors module has two data: the target speed for the robot and the type of gait action. Every type of gait action defines a different algorithm to generate the future footsteps and foot trajectories. The footsteps sequence is generated by the Footstep Planner, as we will explain in section IV.

When the robot is walking, its feet swing alternatively to reach the new positions of the footsteps route. The trajectory that a foot follows in the air is calculated by the Swinging-Foot Pattern Generator. The output of this module is a sequence of cartesian positions and a rotation matrix of the nonsupporting foot in the supporting-foot frame. These positions are delivered to the inverse kinematics module.

Additionally, to prevent the robot from falling, it is necessary to ensure certain stability conditions. In this work, we will employ the inverted pendulum model and the ZMP stability criterion. The ZMP specifies the point with respect to which dynamic reaction force at the contact of the foot with the ground do not produce any moment [6].

The dynamic balance condition resides in keeping the ZMP within the limits of the support polygon (single-support stage) or the convex hull (double-support stage). Since the feet move to follow the footsteps sequence defined, the shape and position of the convex hull will vary. We will therefore define trajectories for moving the ZMP and keeping it within the limits of the convex hull. These trajectories are generated in the ZMP Trajectory Planner module.

The position of the ZMP depends on the dynamics of the Center of Mass (CoM) of the robot. The CoM is the mean location of all the mass in the system. This position may not correspond to the position of any individual mass and allows the use of simplified models of motion.

The position of the CoM is estimated with the data collected from three sources of information: the Hall-effect position sensors of the joints of the robot, the gyrometer sensors of the inertial unit (which is at the torso of the robot) and the Force Sensor Resistors (FSR), located on the corners of the soles of the feet.

With the trajectories obtained by the ZMP Trajectory Planner and the estimation of the current position of the CoM, it is possible to employ an optimum control system called *Model Predictive Controller* to generate the future positions of the CoM that minimize the deviations of the ZMP from the reference trajectory, as will be detailed in section III. The resulting CoM trajectory plan is delivered to the inverse kinematics module that, together with the swinging-foot pattern, will generate the next position for the joint actuators.

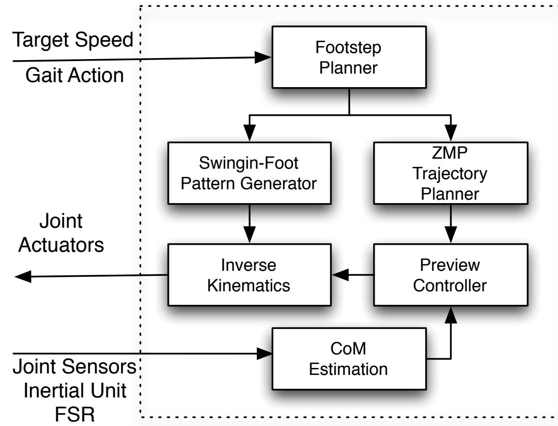

Fig. 1. Walk Engine.

III. Balance model

In order to allow the robot to walk omnidirectionally, it is fundamental to carefully analyze the balance issue. In this section we will briefly describe the approach employed to generate the movement of the CoM that will prevent the robot from falling.

A. Preview Controller

According to the 3D model of the linear inverted pendulum (3D-LIMP), the robot behaves as a unique point placed at the CoM where all the mass of the robot is concentrated. Assuming this simplifications, the ZMP criterion is used to determine the stability of he robot. In the next equations, only the sagittal component is analyzed, results in the coronal plane are equivalent.

The system of reference will change with every new step, placing its origin at the projection on the ground of the ankle of the supporting foot. The x coordinate axis will point from the back to the front of the foot, the y axis from the right to the left and the z axis upwards.

The movements of the CoM will be constrained to an horizontal plane. In this way, the x coordinate of the ZMP, p, for a robot with the CoM at position $c(x, y, z)$ is given by (1) . Gravitational acceleration is represented by g.

$$p = x - \frac{z}{g}\ddot{x} \tag{1}$$

A more difficult problem is to obtain the trajectory of the CoM for a provided reference ZMP trajectory. For this goal, we will employ a ZMP Preview Control scheme, and the analytical solution proposed by Wieber *et. al.* [11], where a detailed explanation of this approach can be found. In this scheme, the CoM and ZMP trajectories are first discretized in constant time fragments of duration T where a constant jerk ($\dddot{x}_k$) is applied to the CoM.

$$\hat{x}_k = \begin{bmatrix} x(KT) \\ \dot{x}(KT) \\ \ddot{x}(KT) \end{bmatrix}, \dddot{x}_k = \dddot{x}(KT), p_k = p(KT) \tag{2}$$

By integrating $\dddot{x}_k$ we get:

$$\hat{x}_{k+1} = \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \hat{x}_k + \begin{bmatrix} T^3/6 \\ T^2/2 \\ T \end{bmatrix} \dddot{x}_k. \tag{3}$$

And equation (1) can be written like:

$$p_k = \begin{bmatrix} 1 & 0 & z/g \end{bmatrix} \hat{x}. \tag{4}$$

Our goal will be to obtain an optimal CoM trajectory that minimizes both the tracking error of ZMP reference trajectory and the jerk of the CoM. It is important to add the jerk to the optimization problem, since otherwise the solution would include abrupt movements of the CoM that would not be possible to perform in the real robot. Consequently, a trade-off must be done between the error of the trajectory tracking and the jerk of the CoM. In (5) the trade-off is handled by the weight parameters Q and R.

$$\min_{\dddot{x}_k, \dddot{x}_{k+1}, \ldots} \sum_{i=k}^{\infty} \frac{1}{2} Q \left(p_{i+1} - p_{i+1}^{ref} \right)^2 + \frac{1}{2} R \dddot{x}_i^2 \tag{5}$$

Equation (5) can be solved with a *Riccati* equation. However, by limiting the duration of the reference trajectory of the ZMP to N samples, a more efficient solution can be found:

$$\dddot{x}_k = -e^T \left(M_u^T M_u + \frac{R}{Q} I_{NxN} \right)^{-1} * \\ M_u^T \left(M_x \hat{x} - P_k^{ref} \right). \tag{6}$$

The matrixes employed in the result equation (6), are defined in the expressions (7),(8),(9) and (10).

$$e = [1, 0...0]^T \tag{7}$$

$$M_u = \begin{bmatrix} \frac{T^3}{6} & 0 & 0 \\ \vdots & \ddots & 0 \\ (1 + 3N + 3N^2)\frac{T^3}{6} & \ldots & \frac{T^3}{6} - T\frac{z}{g} \end{bmatrix} \tag{8}$$

$$M_x = \begin{bmatrix} 1 & T & \frac{T^2}{2} - \frac{z}{g} \\ \vdots & \vdots & \vdots \\ 1 & NT & \frac{N^2 T^2}{2} - \frac{z}{g} \end{bmatrix} \tag{9}$$

$$P_k = \begin{bmatrix} p_k & \ldots & p_{x+N-1} \end{bmatrix} \tag{10}$$

The equation (6) shows the way this mathematical tool can be used to optimize the generation of a CoM trajectory to track a reference ZMP trajectory. However, it is still necessary to define the reference trajectories.

These trajectories are constrained in time and in space, lying at all time within the limits of the convex-hull. Smoothness of the trajectories is also a desirable property, because it decreases the jerk that has to be applied to the CoM. In this way, we will employ spline curves to move the ZMP reference during the double-support stages from one foot to the other. During the single-support stages, the ZMP will stay still in the middle point of the soles.

Although the execution of the locomotion approach described above is satisfactory at certain speeds, there are important deficiencies in its performance that prevent the robot from attaining a faster and more robust gait. To improve the robustness of the system, it is imperative to correct the estimated state of the CoM with the information received from the sensors.

B. CoM Estimation

In order to estimate the position of the CoM, we will conjugate the information from three different sources: the joints sensors, the gyrometers and the FSRs.

Also, it is required to define previously the reference frames that will be used to mix this information. Considering we will chose as reference foot the one that should support the weight of the robot at the current step, we will define a frame at the sole of this foot in the way detailed in the preview controller description. It is named the supporting sole frame (SSF). In the same way, we can define at the sole of the nonsupporting foot the nonsupporting sole frame (NSF). However, the soles of the robot are not always horizontal on the ground. We must therefore define a frame for the projection of the supporting sole on the ground (SGF) and for the projection of the nonsupporting sole on the ground (NGF). Finally, since the inertial unit is located at the torso, the gyrometers provide their measures in the torso frame (TF).

The positions of the joints provided by the Hall-effect sensors of the Nao Robot are quite accurate. By using the Denavit-Hartenberg approach, we obtain the rotation of the torso as well as the position of the CoM in the SSF. On the other hand, the gyrometers provide a good estimate of the angular speed of the torso in the TF. Assuming that the rotation of the torso in the SGF is known, the angular speed of the torso can be expressed in the SSF.

Since we maintain an estimation of the NGF in the SGF, if accidentally the supporting foot looses contact with the floor and the theoretically nonsupporting foot becomes the one in contact, the information provided by the direct kinematics of the NSF can be translated to the SSF.

When the soles are not horizontal, the rotation of the torso will be employed to determine the rotation of the soles. In order to obtain an estimation of the rotation of the torso, we will use a Kalman filter, whose state matrix contains the angular position and speed of the torso.

By checking the values of the FSR sensors of a sole it is possible to know if that sole is horizontal on the ground or not. A force above a minimum threshold in at least three sensors (out of four) is required to determine an horizontal contact. If this is the case, the Kalman filter state matrix is updated with the measures provided by the joint sensors and the gyrometers. In another way, when an horizontal contact is not detected, only the information from the gyrometers is used to update the Kalman filter.

When the supporting sole of a foot is not horizontal on the ground, the distance from its lowest corner to the ankle projection on the sole is used to update the distance information of the sole pose in the SGF. The estimation of the pose of the supporting sole in the SGF can be then employed to project the CoM position from the SSF to the SGF, which is the frame needed by the preview controller approach.

C. Closed Loop

The use of some feedback in the preview controller is fundamental to increase its robustness, specially in non-slippery

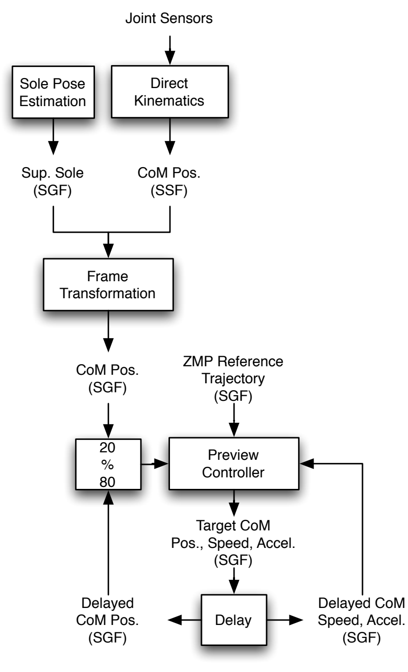

Fig. 2. Preview Controller with CoM position feedback.

surfaces such as the RoboCup carpets.

However, the estimations of the CoM described above can be noisy and discontinuous. If they are used as inputs of the preview controller, its output will vary largely in every cycle and the robot will shake.

To obtain inputs that use the information provided by the sensors while partially conserving the dynamical state of the CoM, we will employ the schema depicted in Fig. 2. With this approach, the position of the CoM estimated by the sensors at a certain instant is mixed at some proportion with the target position sent to the actuators for that instant. Nevertheless, for the speed and acceleration of the CoM only the target information is used, because the estimations provided by the sensors are too noisy.

Since there is a delay of 4 control cycles between the cycle when we send a target position to the actuators and the cycle when the sensors detect a reaction to that target position, it is obligatory to obtain the target position at least this number of cycles in advance. Also, in order to mix information from target positions and sensors, the target positions must be conveniently shifted so that sensor and actuator times match.

D. Divergence Avoidance

For an arbitrary reference ZMP trajectory, the exact solution for the CoM trajectory could easily diverge [12]. To avoid this situation, the input position and speed of the CoM to the Preview Controller are checked to remove risk circumstances. For instance, the CoM sensed position is constrained to the support polygon formed by the previous and future step. In the same way, sensed speeds are limited to those speeds that will not make the CoM exceed the support point in the lateral component plus a certain margin. As for the frontal component, speeds are limited to those speeds that will make the CoM leave the support polygon by the end of the single-support stage if the laws of the 3D-LIMP were followed.

IV. Dribbling

In this section we will explain the approach to design the combination of steps and foot trajectories that compose the kicks and the dribblings. The gait actions are the algorithms employed to create the next footsteps positions, durations and foot trajectories. Every type of gait action has a unique identifier: its *gaitActionID*.

A. Footstep Planner

The Footstep Planner module schedules the future parameters of the next steps that the robot will perform. It receives two command data. One is the target speed of the robot, and the other one is the *gaitActionID* code.

If the *gaitActionID* code indicates a standard gait, the footsteps positions and durations are calculated to satisfy the target speed requirements. On the other hand, if the *gaitActionID* changes, the position and duration of the footsteps are calculated according to the appropriate algorithm.

Regardless of the gaitActionID code received by the *FootStep Planner*, the steps are configured with the parameters shown in the next table, where $SupportingFootID$ indicates if the supporting foot is the right one or the left one.

SupportingFootID
Destination position of nonsupporting foot
Double Support duration
Single Support duration
FootTrajectoryID

The $footTrajectoryID$ code indicates the type of trajectory that will follow the nonsupporting foot during that step. If the robot is walking regularly, the duration of the single and double-support stages are set to their default values, and the nonsupporting foot will execute a spline trajectory to get to its destination.

However, the $footTrajectoryID$ can define any kind of trajectory, specially different types of kicks. Every *footTrajectoryID* has associated a single-support stage duration and a set of control points and duration for the Bezier curves that define the trajectory of the nonsupporting foot.

The simplest gaitAction performs a forward kick without stopping the gait. To achieve this it would be enough to modify the footTrajectoryID code of the next step and set the $footTrajectoryID$ code of a kick. Then, the robot will kick with the corresponding leg without stopping.

Furthermore, the gaitActions can be composed of several steps. In this way, more complex actions can be defined. For example, if the robot has the ball in front of itself and wants to change the sense of its walk without loosing the control of the ball, a gaitAction could be defined to kick the ball with the heel of a foot and turn an angle as wide as possible during the following steps until a 180 turn is reached. Indeed, the same foot that kicks the ball can start turning after the kick.

In the section V we will see an example of a basic dribbling action. First, the ball is kicked diagonally, and in the following step the robot is faced towards the expected ball position.

B. Foot Trajectory Design

In order to design the trajectories of the nonsupporting foot for the kicking gait actions, we have employed piecewise cubic Bezier curves, similarly to [8].

Cubic Bezier curves are defined by four control points P_0, P_1, P_2 and P_3 in the interval $0 <= t <= 1$ as in (11). The points P_0 and P_3 define the starting and the ending points of the trajectory, while the points P_1 and P_2 provide information to calculate the slope of the curve at the points P_0 and P_3 respectively.

$$b(t) = \sum_{i=0}^{3} \binom{3}{i} t^i (1-i)^{3-1} P_i \tag{11}$$

The design of complex trajectories can be achieved by the combination of several Bezier curves. In order to preserve the continuity and the differentiability, we have to apply the constraints (12) and (13) to the points of adjacent Bezier curves. The super index j of the control points indicate the sequential index of the Bezier curve, and T^j denote the duration of the Bezier curve j.

$${P_3}^j = {P_0}^{j+1} \tag{12}$$

$$\frac{{P_3}^j - {P_2}^j}{T^j} = \frac{{P_1}^{j+1} - {P_3}^j}{T^{j+1}} \tag{13}$$

Since the initial and the final position of the trajectory are defined by the stepPlanner, they will provide the points ${P_0}^0$ and ${P_3}^n$. In the same way, we know that the nonsupporting foot that performs the trajectory was the supporting foot in the previous and in the following step, so it must be still at the beginning and at the end positions. This forces the 0 value for the points ${P_1}^0$ and ${P_2}^n$. The rest of the points of the trajectory and the duration each one of the Bezier curves can be set freely, as long as constraints (12) and (13) are true.

In Fig. 3 we can observe the trajectory followed by the nonsupporting foot to perform a frontal kick.

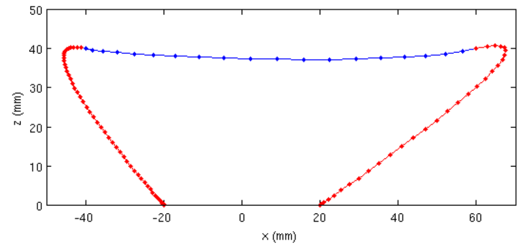

Fig. 3. Foot trajectory in the sagittal plane while performing a forward kick. The color changes indicate different Bezier trajectories.

V. Experimental validation

A. Platform Setup

In this section we will review some of the special tunings that make the system work in this platform. Although they cannot be considered as optimized parameters, they provide a reference or a starting point for similar developments.

The first parameter to be taken into account is the duration of the single and double-support stages. The addition of the two durations will be inversely proportional to the speed of the robot; hence the interest to reduce them. As the simple-support stage allows to advance the swinging leg, the shorter the duration of this stage, the larger the momentum that will suffer the torso of the robot. And this momentum is neglected by the 3D-LIMP model. On the other hand, the double-support stage is not strictly necessary, but its presence increases stability. A good compromise between stability and speed has been found for a duration of 250 ms for the single-support stage and of 100 ms for the double-support stage. The walking speed achieved with this configuration is similar to that of the gait provided by Aldebaran Robotics.

The value of the R/Q parameter, has been set to 10^{-6}. As the magnitude of the variation in the ZMP reference trajectory is smaller in the sagittal plane, the minimization of the jerk predominates over the reference ZMP trajectory tracking, in equation (5).

Concerning the height of the CoM, we have found that 260 mm provides acceptable performance. Lower values offer better stability against external perturbations and can lead to faster gaits, but the pitch joints in the ankles and the knees have to support a heavier load. This extra load involves a faster heating of the joints, which dramatically reduces the operational time of the system and could cause joint failures during the matches.

Since the duration of the single-support stage is much longer than the duration of the double-support stage, some of its duration could be dedicated to keep the swinging leg on the floor before and after the swinging movement to increase stability. In this way, we will only use 90% of the single-support duration to move the swinging leg, while the remaining 10% is dedicated to keep booth feet on the floor. Please note that it is not equivalent to employing the 10% of the duration to increase the duration of the double support-stage.

Regarding the stride configuration, the main parameter to define is the step height. Our experience suggests that a height of 15 mm in the central part of the stride provides good performance. Lower strides can be useful in sliding floors to generate sky-type gaits, but when these parameters are used on regular carpet floors, friction compromises stability.

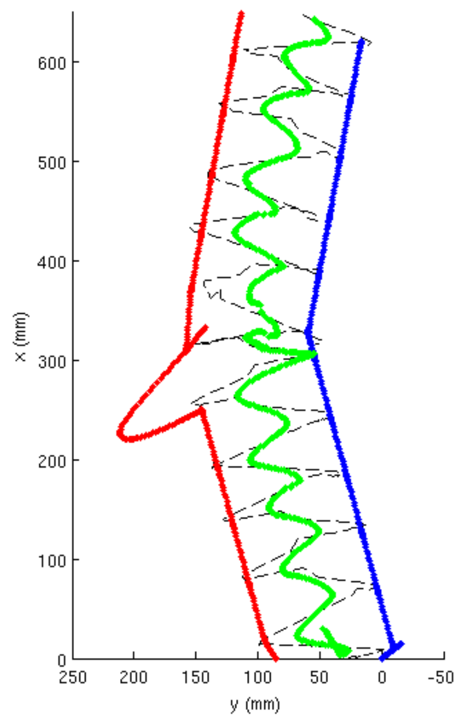

Fig. 4. Trajectories of the CoM (green), left foot (red) and right foot (blue) target signals while the robot performs the dribbling experiment. The ZPM of the CoM target signal is also displayed in black.

B. Dribbling Experiment

The experiment carried out to proof the efficacy of the approach is a basic dribbling towards the forward-right direction. To perform the dribbling it is necessary to link a kick in diagonal direction with a turning step in the same direction. During the whole action, the closed loop Preview Controller ensures the stability of the robot.

Please note that in this experiment the vision system is not employed, since the position of the ball is known in advance. The accurate estimation of the position of the ball is a work that exceeds the scope of this paper.

In Fig. 4 we can observe the trajectory followed by the CoM of the robot in the transversal plane. The CoM moves an extra space to the right foot to let the left foot more time to perform the kick. The total time employed for the kick is 2.1 seconds. Once the kick is done, the next step of the right foot is a forward and turning step in the direction where the ball should have gone. Please note that the instability produced just after the ball kicking is correctly controlled by the MPC.

By advancing in the ball direction without waiting for the behavior signal, which would be queued at the end of the future footsteps sequence, the robot dribbling becomes more agile. On the other hand, if the robot looses the control of the ball during the dribbling, it will be in a worse situation than at the beginning, because it will walk a step in the direction where the ball is not. It is a risk that the behavior must consider, just as happens in real soccer. However, it is always possible to abort the dribbling online to mitigate the disadvantageous situation.

VI. Conclusions and Future Works

The gait described in this paper is a robust and parametric locomotion system that admits the integration of kicks in the walk engine. Moreover, it allows the design of complex actions composed by kicks and predefined steps.

In future developments we will incorporate the position of the ball to the gait action algorithms to improve their efficiency. The ambitious goal is to create an interface for the locomotion module similar to the one used in soccer video games. In these games, human players only indicate the running direction and sporadically the kick or dribbling action, while the machine takes care of the ball handling.

Acknowledgement

This work has been supported by Spanish Ministry of Science and Innovation under the DPI-2007-66556-C03-02 CICYT project and by the Spanish Ministry of Education through its FPU program.

References

[1] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, B. Maisonnier, *The NAO humanoid: a combination of performance and afford- ability* CoRR abs/0807.3223, 2008.

[2] D. Gouaillier, C. Collette, C. Kilner, *Omni-directional closed-loop walk for NAO*, in: Humanoid Robots (Humanoids), 10th IEEE-RAS International Conference on. pp. 448454, 2010.

[3] S. Czarnetzki, S. Kerner, O. Urbann, *Observer-based dynamic walking control for biped robots*, in Robotics and Autonomous Systems 57(8):839–845, 2009.

[4] C. Graf, T. Röfer, *A center of mass observing 3D-LIPM gait for the RoboCup Standard Platform League humanoid*, in RoboCup 2011: Robot Soccer World Cup XV, 2011.

[5] F. Xue, X. Chen, J. Liu, and D. Nardi, *Real Time Biped Walking Gait Pattern Generator for a Real Robot*, in RoboCup 2011: Robot Soccer World Cup XV, 2011.

[6] M. Vukobratovic, A.A. Frank, and D. Juricic, *On the Stability of Biped Locomotion*, in IEEE Transactions on Biomedical Engineering, 17(1):25-36, 1970.

[7] S. Kajitak, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi and H. Hirukawa, *The 3D Linear Inverted Pendulum Mode: A simple modeling for a biped walking pattern generation*, in Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Hawai, USA, 2001.

[8] J. Müller, T. Laue und T. Röfer, *Kicking a Ball Modeling Complex Dynamic Motions for Humanoid Robots*, RoboCup 2010: Robot Soccer World Cup XIV, Nr. 6556, pages 109120, Lecture Notes in Artificial Intelligence, 2010.

[9] B. D. Damas, P. U. Lima and L. M. Custódio, *A Modified Potential Fields Method for Robot Navigation Applied to Dribbling in Robotic Soccer*, in RoboCup 2002: Robot Soccer World Cup VI, Volume 2752/2003, pp. 65–77, Lecture Notes in Computer Science, 2003,

[10] J. Best, R. van de Molengraft, *An active ball handling mechanism for RoboCup* , in 10th Int. Conf. on Control, Automation, Robotics and Vision, 2008.

[11] P.-B. Wieber, *Trajectory Free Linear Model Predictive Control for Stable Walking in the Presence of Strong Perturbations*, IEEE-RAS Int. Conf. on Humanoid Robots, 2006.

[12] S. Hong, Y. Oh, Y. H. Chang, B. J. You, *Omni-directional Walking pattern Generation Method for Humanoid Robots with Quartic Polynomials*, in IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, San Diego, CA, USA, 2007.