# Improved Online Kick Generation Method for Humanoid Soccer Robots

Seung-Joon Yi , Stephen McGill, and Daniel D. Lee

***Abstract*— Approaching and kicking the ball toward the goal is the very essence of robotic soccer. However, for humanoid robots, it is hard to make a fast and stable kick as it includes large momentum changes while standing on a single foot. Simple keyframe based approaches tend to fail due to differences between robots or external perturbations, so they need to be executed slowly. Slow execution increases the risk of losing the ball preparing for the kick, however.**

**In this paper, we first review two online kick generation methods that we have been using for the RoboCup competition: a stationary kick controller that uses inverse kinematics and feedback stabilization and a walk-kick controller which is based on our analytic zero moment position (ZMP) based walk controller. Finally, we present a ZMP preview based kick generation method that combines the power of the stationary kick with the speed of the walk-kick controller.**

## I. Introduction

Kicking the ball is an essential movement in order to score in robotic soccer. Developing a kick includes two main factors: power and speed. It is obvious that a robot with powerful kick that moves the ball across the entire field has a much higher chance of scoring than a robot that requires several kicks to reach the goal. However, small, repeated kicks are just as important, as the robot that takes a long time to deploy the powerful kick will probably lose the ball to a quicker opponent.

In developing a variety of kicks for specific scenarios, the simplest approach is to design a number of sets of joint angles, or keyframes, and make a continuous motion out of them using interpolation techniques. This approach is also commonly used allow a robot to get up after falling down. However, for humanoid robots to kick, they should keep their balance on one foot while moving the other foot quickly. Extended periods of time on one foot poses a big challenge in terms of stability. It is generally hard to hand design stable keyframes for kick motions, which get even harder due to individual differences among robots. Also, as keyframe motions are designed in advance, they cannot cope with various external perturbations soccer robots experience during the match. As a result, keyframe-based kick approach must compromise power or speed in order to achieve usable stability. Thus, these robots are slowed down a lot for powerful kicks, or confined to a fairly weak kick.

Thus a better approach is to generate the kick motion in an online fashion, which enables real-time modulation of the kick motion to compensate for actual ball positions and external perturbations. In this paper, we first introduce the stationary kick controller with feedback stabilization that uses the inverse kinematics of the robot and a few parameters to generate a generalized kick motion online, as shown in Figure 1. Similar parameter based kick motions have been used in RoboCup [1], including online motion generation [2]. Next, we present a less powerful but faster kick controller that utilizes the analytic ZMP based locomotion controller to perform kick during locomotion, which we dub "walk-kick". Finally, we suggest a novel kick controller that combines the virtues of the previous algorithms: a ZMP preview controller based kick that can initiate a powerful kick without having to stop walking. Of note, our approach does not focus on differing kick directions, as like other previous work [3].

Fig. 1. DARwIn-OP robot kicking a ball with a stationary kick.

The remainder of the paper proceeds as follows. Section II describes the outline of the control architecture. Section III reviews the stationary kick controller that uses inverse kinematics and sensory feedback to generate a stabilized kick motion. Section IV reviews the walk-kick controller that extends our analytical ZMP based walk algorithm to perform a quick kick. Section V presents the preview control based kick controller that enables a smooth transition between walking and kicking without stopping. Sections VI and VII shows results from using a physics-based simulation and

GRASP Laboratory, University of Pennsylvania, Philadelphia, PA 19104
{yiseung,smcgill3, ddlee}@seas.upenn.edu

from experiments using the DARwIn-OP humanoid robot. Finally, we conclude with a discussion of potential future directions arising from this work.

## II. The Motion Controller Architecture

Our motion controller consists of a step planner, trajectory generation controllers and an inverse kinematics (IK) solver. The step planner plans the next foot and torso location for each step, and the foot trajectory controller generates foot trajectories for the swing foot, while the torso trajectory controller generates center of mass (COM) trajectories so that the resulting ZMP lies within the support polygon for dynamic stability. Finally, the IK solver solves for the actual joint angles so that torso and feet follow the desired trajectory.

We have designed the motion controller with modularity and generalizability in mind, and the whole motion framework is easily ported to a number of different platforms. We have successfully used the motion controller on several platforms, including different generations of the DARwIn kidsize robot, the Nao standard platform robot, and the DARwIn-XOS teensize robot and CHARLI adultsize robot.

In this paper, we present how the components of the basic motion controller can be utilized for kick generation in an online fashion. Instead of using keyframes, or sets of pre-specified joint angles, we utilize the kinematics and dynamics of the robot to generate an adaptive kick motion online. To kick the ball more quickly, we also extend our locomotion engine to perform walk-kicks without stopping.

## III. Stationary Kick Controller

When the robot keeps the static balance during whole kick sequence, which means the COM position of the robot always lies in the support region, we call this approach a stationary kick. Typical keyframe based kicks fall into this category. Instead of designing the kick based on keyframes in the space of joint angles, we define a small number of parameters that determine the movement of the torso and the kick foot in 3D space. With these parameters, we generate foot and torso trajectories in real time using IK.

There are many advantages to this approach. Designing and tuning a new kick is much easier than making a keyframe kick in joint space, since only a few human understandable parameters are used. Also, with abstracted parameters, the kick can easily generalized to robots with different joint configurations. Additionally, sensory feedback can be used for active stabilization during kicking, potentially allowing for faster and more powerful kicks.

We define a kick as a sequence of $KICK_i$ kick elements:

$$KICK_i = \{SF, t_{STEP}, L_i, T_i, R_i, L_{i+1}, T_{i+1}, R_{i+1}\} \qquad (1)$$

where $SF$ is the support foot, $t_{STEP}$ is the time of that element, $L, T, R$ are 6D coordinate$(x, y, z, \psi, \theta, \phi)$ for left foot, torso and right foot. Each kick element corresponds to an action during the kick such as lifting, kicking and landing.

Fig. 2. Stationary kicks implemented on various humanoid robot platforms

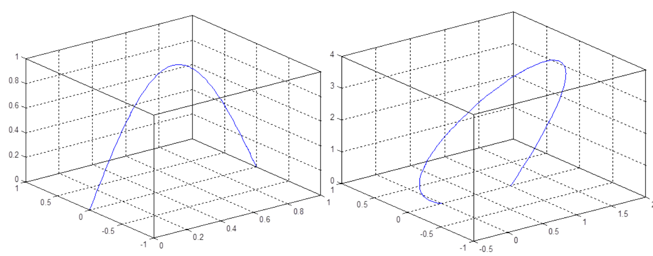


Fig. 3. Comparison of foot trajectories for walking and kicking

Stabilization is achieved by modulating the torso position and angle based on inertial sensors. We found that the stabilization is very effective in coping with individual differences among our robots, which makes it very hard to make a single non-stabilized keyframe that works well over many robots. With the help of this simple operation space definition for kicking, we can easily make and test a number of different kicks in a short time for multiple platforms, which was especially useful for RoboCup challenges such as passing or the high kick. Figure 2 shows a number of stationary kicks implemented on various humanoid robot platforms.

## IV. Walk-Kick Controller

The main disadvantage of the static kick is a robot must fully stop walking and take some time to stabilize itself. When two robots are confronting each other, which happens a lot during RoboCup matches, a quick kick is much more helpful than a strong but slow kick. For this reason, many teams use a quick, less powerful kick that can be triggered

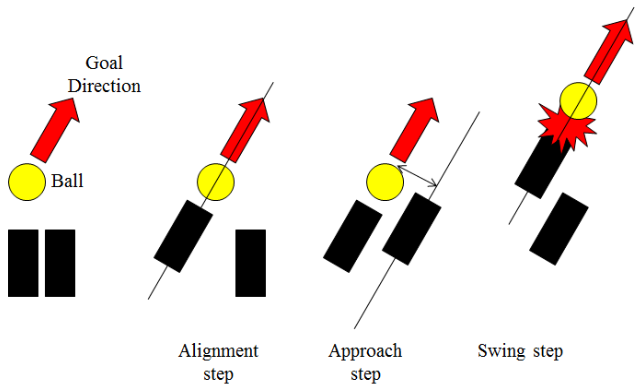



Fig. 4. Walkkick with online alignment

during a single step of locomotion, which are usually called in-walk kicks or simply walk-kicks.

A simple way to implement a walk-kick is to use a custom foot trajectory for the kick step, as shown in Figure 3, while walking in place. With this technique, the underlying locomotion controller and walk parameters are usually unchanged. This approach has been widely used in the Nao league, but it still has a few disadvantages.

For smaller robots with short leg lengths, or when the stepping frequency is high, there is not enough time for the kick leg to accelerate, hit the ball, and return to the initial position, which results in fairly weak kicking power. Additionally, the robot must walk in place while kicking, so it takes time for the robot to re-accelerate to follow the ball.

Thus, our approach includes custom kick step sequences in addition to special foot trajectories, utilizing our queue based locomotion controller. The front kick is implemented by putting two steps in the step queue: a support step and a kick step. For the kick step, we use a longer step period and a special foot trajectory to maximize the foot velocity at hitting the ball. After the robot kicks the ball, the step queue is emptied and the robot resumes walking according to its commanded walk velocity without stopping. Similarly, a side walk-kick consists of three steps including two normal steps and one special step.

As the step positions are not pre-specified, we can implement a feature named "online alignment" which is shown in Figure 4. If the robot pose has some positional or angular error when it starts kicking, it is compensated by modulating the support step position and angle.

We have found that as the body is also moving forward, the dynamic front kick has more range than its static kick counterpart. It can be executed very fast, too – it takes 3 steps in worst case, which is still considerably faster than the static kick case. Kick strength is much higher with initial support step compared to one without it, but still much less than static kick. However, as the robot completes its kick much faster and it keeps moving forward during kicking, it can quickly catch up to the ball and kick again. Figure 5 shows a number of walk-kick examples implemented on various humanoid platforms.

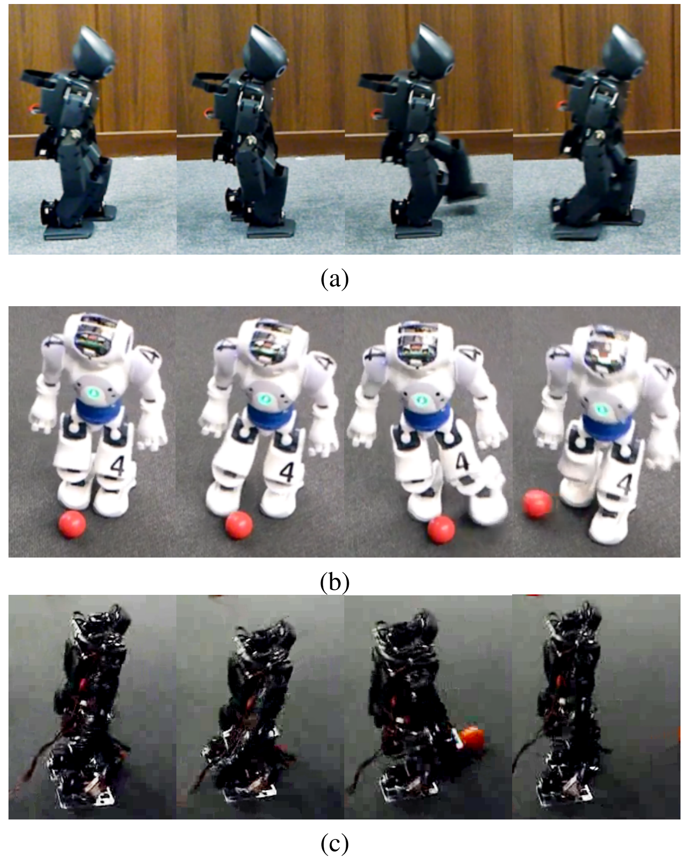
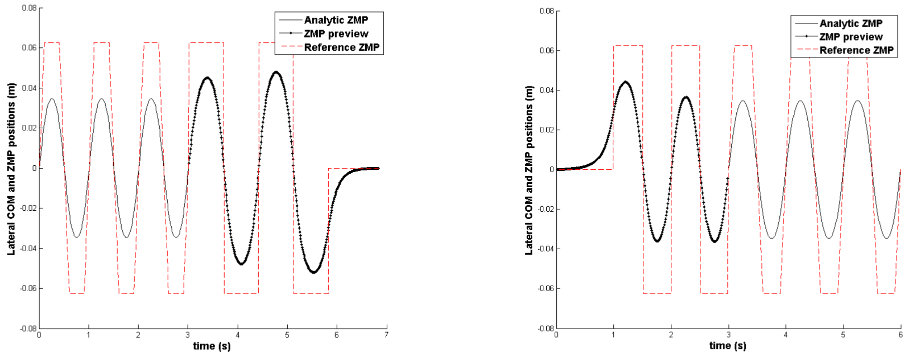



Fig. 5. Examples of walk-kick.(a) DARwIn-OP doing front walk-kick (b) Nao doing side walk-kick (c) DARwIn-HP doing front walk-kick

## V. Preview Control Based Kick Controller

In the previous sections, we have seen the stationary kick controller that keeps the robot statically balanced during the kick, and the walk-kick controller that keeps the robot dynamically balanced by utilizing a ZMP based locomotion controller. The stationary kick controller can be more powerful as it keeps the robot in single support phase for a longer time than walk-kick controller. However, it is slow as it includes a full stop of locomotion and does not utilize the dynamics of the robot. On the other hand, the walk-kick controller is quick but less powerful due to limited single-support time while kicking.

In this section, we present a totally new approach that combines the advantages of two methods by utilizing a hybrid locomotion controller that seamlessly switches between two different modes. Locomotion is handled by the analytic ZMP controller that we have been using; when a kick signal is triggered, the walk controller uses a ZMP preview subcontroller that uses a pre-designed kick step sequence and custom foot trajectory to initiate a dynamic kick. After the kick sequence, the walk controller switches back to using the analytic subcontroller to keep chasing the ball, since the analytic sub controller is better at rejecting disturbances. We will further explain the method in following subsections.




(a) (b)

Fig. 6. Transitions between the two kick trajectory generation subcontrollers are shown. The black line denotes the COM trajectory, while the red dashed line denotes the ZMP.

### A. Analytic ZMP Based COM Trajectory Generation

In this subsection we briefly review our analytic ZMP based locomotion algorithm [4]. The $i_{th}$ step is defined the same way

$$STEP_i^R = \{SF_i, L_i, R_i, L_{1+1}, R_{i+1}, t_{STEP}^i\}, \tag{2}$$

where $SF_i \in \{LEFT, RIGHT\}$ denotes the support foot, $L_i$, $R_i$ the pose of left and right feet in $(x, y, \theta)$, and $t_{STEP}$ the duration of the step. We confine the ZMP trajectory $p_i(\phi)$ to have a trapezoidal form, which is defined as

$$p_i(\phi) = \begin{cases} C_i(1-\frac{\phi}{\phi_1}) + L_i\frac{\phi}{\phi_1} & 0 \le \phi < \phi_1 \\ L_i & \phi_1 \le \phi < \phi_2 \\ C_{i+1}(1-\frac{1-\phi}{1-\phi_2}) + L_i\frac{1-\phi}{1-\phi_2} & \phi_2 \le \phi < 1 \end{cases}, \tag{3}$$

for the left support case. We find the following analytic solution of COM trajectory with zero ZMP error during the step period $0 \le \phi < 1$

$$x_i(\phi) = \begin{cases} p_i(\phi) + a_i^p e^{\phi/\phi_{ZMP}} + a_i^n e^{-\phi/\phi_{ZMP}} \\ +m_i t_{ZMP}(\dfrac{\phi-\phi_1}{\phi_{ZMP}} - \sinh\dfrac{\phi-\phi_1}{\phi_{ZMP}}) & 0 \le \phi < \phi_1 \\ \\ p_i(\phi) + a_i^p e^{\phi/\phi_{ZMP}} + a_i^n e^{-\phi/\phi_{ZMP}} & \phi_1 \le \phi < \phi_2 \\ \\ p_i(\phi) + a_i^p e^{\phi/\phi_{ZMP}} + a_i^n e^{-\phi/\phi_{ZMP}} \\ +n_i t_{ZMP}(\dfrac{\phi-\phi_2}{\phi_{ZMP}} - \sinh\dfrac{\phi-\phi_2}{\phi_{ZMP}}) & \phi_2 \le \phi < 1 \end{cases} \tag{4}$$

where $\phi_{ZMP} = t_{ZMP}/t_{STEP}$ and $m_i$, $n_i$ are ZMP slopes which are defined as

$$m_i = \quad (L_i - C_i)/\phi_1 \tag{5}$$
$$n_i = \quad -(L_i - C_{i+1})/(1-\phi_2), \tag{6}$$

for the left support case. Parameters $a_i^p$ and $a_i^n$ can then be uniquely determined from the boundary conditions.

### B. ZMP Preview Based COM Trajectory Generation

Kajita et. al. [5] proposed a general approximation method to compute the COM trajectory given reference ZMP trajectory based on the following LIPM equation.

$$\ddot{x} = \frac{1}{{t_{ZMP}}^2}(x-p), \tag{7}$$

where $t_{ZMP} = \sqrt{z_0/g}$. If we define a new control variable $u_x$ as the time derivative of the acceleration of COM (8), then we can translate (7) into a strictly proper dynamical system:

$$\frac{\mathrm{d}}{\mathrm{d}t}\ddot{x} = u_x \tag{8}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u_x$$
$$p_x = \begin{bmatrix} 1 & 0 & -t_{ZMP}^2 \end{bmatrix}\begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix} \tag{9}$$

If we discretize the system of (9) with sampling time of $T$ then

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \\ p(k) &= Cx(k), \end{aligned} \tag{10}$$

where

$$\begin{aligned} x(k) &\equiv \begin{bmatrix} x(kT) & \dot{x}(kT) & \ddot{x}(kT) \end{bmatrix}^T, \\ u(k) &\equiv u_x(kT), \\ p(k) &\equiv p_x(kT), \\ A &\equiv \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix}, \\ B &\equiv \begin{bmatrix} T^3/6 \\ T^2/2 \\ T \end{bmatrix}^T, \\ C &\equiv \begin{bmatrix} 1 & 0 & -t_{ZMP}^2 \end{bmatrix}. \end{aligned}$$

Then, given the reference ZMP $p^{ref}(k)$, the performance index can be specified as

$$J = \sum_{i=k}^{\infty} \{Q_e e(i)^2 + R\Delta u^2(i) + \Delta x^T(i) Q_x \Delta x(i)\} \tag{11}$$

where $e(i) \equiv p(i) - p^{ref}(i)$ is ZMP error, $\Delta x(i)$ and $\Delta u(i)$ are the incremental state vector and control input $x(k) - x(k-1)$ , $u(k) - u(k-1)$, and $Q_e$, $Q_x$ and $R$ are weights. If we assume that the ZMP reference $p^{ref}$ can be previewed for $N$

future steps at every sampling time, the optimal controller that minimizes the performance index (11) is given as

$$u(k) = -G_i \sum_{i=0}^{k} e(k) - G_x x(k) - \sum_{j=1}^{N} G_p(j) p^{ref}(k+j), \quad (12)$$

where $G_i$, $G_x$ and $G_p(j)$ are gains that can be calculated in advance from weights and system parameter of (10).

### C. Transition Between Subcontrollers

Transitioning from the reactive (analytic ZMP-based walk controller) to preview subcontroller is quite straightforward. Differentiating (4), we get following boundary values for $x_i(t)$:

$$\begin{aligned} x_i(t_{STEP}) =& C_{i+1} \\ \dot{x}_i(t_{STEP}) =& \dot{p}_i(t_{STEP}) + \frac{a_i^p e^{1/\phi_{ZMP}} - a_i^n e^{-1/\phi_{ZMP}}}{t_{STEP}\phi_{ZMP}} \\ &+ \frac{n_i t_{ZMP}}{t_{STEP}\phi_{ZMP}}(1 - \cosh\frac{1-\phi_2}{\phi_{ZMP}}) \\ \ddot{x}_i(t_{STEP}) =& \ddot{p}_i(t_{STEP}) + \frac{a_i^p e^{1/\phi_{ZMP}} + a_i^n e^{-1/\phi_{ZMP}}}{t_{STEP}^2\phi_{ZMP}^2} \\ &- \frac{n_i t_{ZMP}}{t_{STEP}^2\phi_{ZMP}^2} \sinh\frac{1-\phi_2}{\phi_{ZMP}} \end{aligned} \quad (13)$$

If we use those values as the initial value of $x(k)$, we can get a continuous COM trajectory up to the second derivative. Figure 6 (a) shows the COM and ZMP trajectories when we switch from reactive subcontroller to preview one. We can see that a longer step duration for preview controller results in a larger amplitude of COM trajectory.

On the other hand, transition from preview subcontroller to reactive subcontroller is not straightforward as the 3D COM trajectory from preview controller may not satisfy the boundary conditions of reactive subcontrollers, which can be derived as

$$\begin{aligned} x_i(0) &= C_i \\ \dot{x}_i(0) &= \dot{p}_i(0) + \frac{a_i^p - a_i^n + n_i t_{ZMP}(1 - \cosh\frac{-\phi_2}{\phi_{ZMP}})}{t_{STEP}\phi_{ZMP}} \\ \ddot{x}_i(0) &= \ddot{p}_i(0) + \frac{a_i^p + a_i^n - n_i t_{ZMP}\sinh\frac{-\phi_2}{\phi_{ZMP}}}{t_{STEP}^2\phi_{ZMP}^2} = 0 \end{aligned} \quad (14)$$

To ensure the preview controller to satisfy the boundary conditions, we add error terms to (11)

$$\begin{aligned} J = \quad & \sum_{i=k}^{\infty} \{Q_e e(i)^2 + R\Delta u^2(i) + \Delta x^T(i) Q_x \Delta x(i)\} \\ & + Q_{t_0}(x(T_{tr}) - x_i(0))^2 + Q_{t_1}(\dot{x}(T_{tr}) - \dot{x}_i(0))^2 \\ & + Q_{t_2}(\ddot{x}(T_{tr}) - \ddot{x}_i(0))^2 \end{aligned} \quad (15)$$

where $x(k)$, $\dot{x}(k)$, $\ddot{x}(k)$ represent the state of the ZMP preview controller at the discrete time $k$, $x_i(t)$, $x_i(t)$, $x_i(t)$ the state of analytic controller at continuous time $t$, $Q_{t_0}$, $Q_{t_1}$, $Q_{t_2}$ weights and $T_{tr}$ the discrete time of the transition. In addition to extending the performance index of preview controller, we put $N$ copies of reactive steps with the current walk velocity in the ZMP queue so that the COM trajectory from ZMP preview controller is close to the steady state COM trajectory from reactive controller. We have found that this algorithm generates continuous and smooth COM trajectories during transitions, as shown in Figure 6 (b).

### D. Kick Generation Using Two Subcontrollers

We use the same kick sequence description as (1) to define foot trajectories for the ZMP preview subcontroller. The ZMP position of each action is set to the center of the current support polygon, determined by the support foot information *SF* and current foot positions. The main benefit of the preview controller over the analytic ZMP one is that we can use arbitrary ZMP trajectory; we can insert a long kicking step in the middle of short walking steps without any ZMP fluctuation. We have found that the ZMP preview based kick controller can exactly replicate the strong stationary kick motion without noticeable stability issue with Nao platform, yet taking almost the same time as the weaker walk-kick controller. Figure 7 shows the example of ZMP preview based kick implemented on DARwIn-OP robot.

## VI. Experimental Results

In the beginning of the paper, we suggested two main factors in measuring the effectiveness of a kick: power and speed. We evaluated our three kick controllers in a controlled environment to measure the distance of each kick. We placed a ball several centimeters in front of the robot. Henceforth, we allowed the robot to approach the ball autonomously. After the robot aligned to the ball properly, it executed the kick. The alignment for each kick remained the same.

To measure the speed of the kick, we analyzed the timestamps for each step, and recorded them below. The underlying walk controller seeks to keep the robot tracking the timestamp commands. The important timestamps during the kicking process include the alignment for striking the ball (start time) and from alignment to chasing the ball after landing from the kick (total time). Distance and timing numbers are summarized in Figure 8 and Table I.

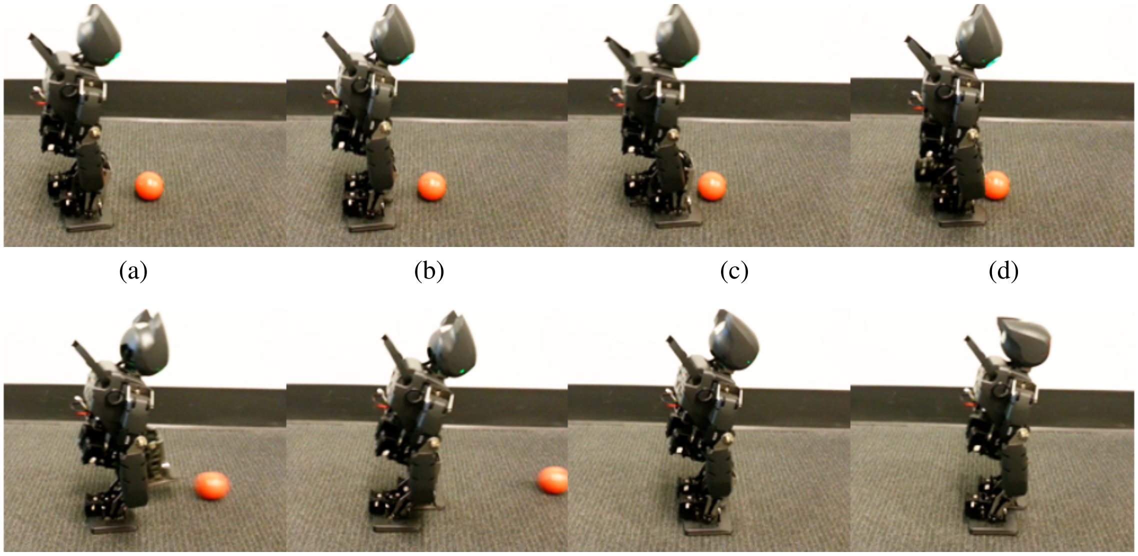
TABLE I

Average kick distances (centimeters) and time (seconds)

| | Stationary | Walk-Kick | Preview |
|---|---|---|---|
| Distance | 411 | 188 | 292 |
| Start Time | 2.60 | 0.70 | 0.70 |
| Total Time | 4.30 | 0.95 | 0.97 |

We can see that walk-kick is roughly 4 times faster than the stationary kick, but can push the ball half as far. On paper the walk-kick does not look quite useful compared to its much stronger stationary kick counterpart, but it worked well in practice. During the RoboCup 2011 and 2012, we experimented with a number of attack strategies utilizing these two type of kicks and found that kicking more quickly while constantly moving the ball forward works well against many teams.

While speed is important, distance also matters. The preview control based kick has the best of both worlds; it can



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 7. DARwIn-OP robot performing a dynamic kick in middle of reactive walking.

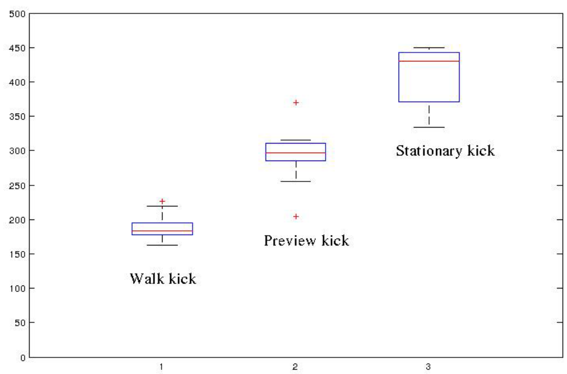



Fig. 8. A box plot comparison of kick distances shows that the preview kick can pouch the ball nearly as far as a stationary kick.

execute nearly as quickly as the walk-kick and push the ball nearly as far as the stationary kick. In tuning the preview kick, we chose parameters to keep the speed of the walk-kick while gaining meaningful additional kicking reach. On a soccer field, using the walk-kick, the robot needs three successful kicks to score from the center area of the field; the preview control kick requires only one or two in the same situation. This efficiency helped us greatly during RoboCup 2013 against a number of fast-moving teams.

## VII. CONCLUSIONS

We reviewed two of our online kick controllers for humanoid robotic soccer, which included the stationary kick controller that excels at power and the walk-kick controller based on analytic ZMP locomotion controller that is very fast. To unify each advantage, we presented a novel ZMP preview control based kick controller that combines the power of the stationary kick controller with the speed of walk-kick controller. Going forward, it will be important to use knowledge of the game scenario to choose among these three kick strategies, rather than choosing solely method. Additionally, the ZMP-Preview controller may allow for unorthodox kicking maneuvers that would otherwise be unstable

## ACKNOWLEDGMENTS

We acknowledge the support of the NSF PIRE program under contract OISE-0730206 and ONR SAFFIR program under contract N00014-11-1-0074. This work was also partially supported by the NRF grant of MEST (0421-20110032), the IT R&D program of MKE/KEIT (KI002138, MARS), and the ISTD program of MKE (10035348).

## REFERENCES

[1] J. Müller, T. Laue, and T. Röfer, "Kicking a ball - modeling complex dynamic motions for humanoid robots," in *RoboCup 2010: Robot Soccer World Cup XIV*, ser. Lecture Notes in Artificial Intelligence, J. R. del Solar, E. Chown, and P. G. Ploeger, Eds., vol. 6556. Springer; Heidelberg; http://www.springer.de/, 2011, pp. 109–120.

[2] F. Wenk and T. Röfer, "Online Generated Kick Motions for the NAO Balanced Using Inverse Dynamics. ," in *RoboCup 2013: Robot Soccer World Cup XVI Preproceedings*. RoboCup Federation.

[3] Y. Xu and H. Mellmann, "Adaptive motion control: Dynamic kick for a humanoid robot," in *KI 2010: Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science, R. Dillmann, J. Beyerer, U. Hanebeck, and T. Schultz, Eds. Springer Berlin Heidelberg, 2010, vol. 6359, pp. 392–399. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-16111-7_45

[4] S.-J. Yi, B.-T. Zhang, D. Hong, and D. D. Lee, "Online learning of a full body push recovery controller for omnidirectional walking," in *IEEE-RAS International Conference on Humanoid Robots*, 2011, pp. 1–6.

[5] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, and K. H. K. Yokoi, "Biped walking pattern generation by using preview control of zero-moment point," in *in Proceedings of the IEEE International Conference on Robotics and Automation*, 2003, pp. 1620–1626.