In Proceedings of 18th RoboCup International Symposium, Joao Pessoa, Brasil, July 2014.
Best Science Paper Award.

Balanced Walking with Capture Steps

Marcell Missura and Sven Behnke

Autonomous Intelligent Systems, Computer Science, Univ. of Bonn, Germany
{missura,behnke}@cs.uni-bonn.de
http://ais.uni-bonn.de

Abstract. Bipedal walking is one of the most essential skills required to play soccer with humanoid robots. Superior walking speed and stability often gives teams the winning edge when their robots are the first at the ball, maintain ball control, and drive the ball towards the opponent goal with sure feet. In this contribution, we present an implementation of our Capture Step Framework on a real soccer robot, and show robust omnidirectional walking. The robot not only manages to locomote on an even surface, but can also cope with various disturbances, such as pushes, collisions, and stepping on the feet of an opponent. The actuation is compliant and the robot walks with stretched knees.

1 Introduction

For the RoboCup initiative, which has the goal of defeating the human world champions in the game of soccer by the year of 2050, it is of particular interest to conceive a bipedal walk with human-like capabilities. However, the complexity of the walking motion, the formulation of sufficiently simple models that account for balance, and the difficulties that arise from controlling a humanoid body with a high number of degrees of freedom within a feedback loop, make this task particularly difficult. While a number of sophisticated approaches exist that promise some degree of robustness, the RoboCup experience shows that state of the art algorithms do not find their way into the dynamic world of low-cost robots competing on the soccer field. One of the reasons for this is simply that the required sensors, high-precision actuators, and computational power are not available on custom built prototypes and affordable standard platforms that are used in robotic soccer games. The amount of expertise required to successfully integrate a complex algorithm into already complex soccer software in a real robot environment is also not a negligible factor.

The Capture Step Framework [1] has been designed with the aforementioned limitations in mind. Using only postural information provided by motor encoders and an inertial measurement unit (IMU), it can produce a stable omnidirectional walk with push-recovery capabilities. Zero moment point control, foot placement, and step timing strategies are utilized simultaneously. The balance computations

This work is supported by Deutsche Forschungsgemeinschaft (German Research Foundation, DFG) under grants BE 2556/6 and BE 2556/10.

are based on a two-dimensional linear inverted pendulum model, and can interface with already working open-loop gait pattern generators. The framework can operate on systems with high sensor noise, high latency, and imprecise actuation. Furthermore, it does not restrict the center of mass to a plane, or the feet to remain flat on the ground. In this paper, we demonstrate the capabilities of the framework on a real soccer robot in combination with compliant actuation and walking with stretched knees. We outline the theoretical concept of the Capture Step Framework, provide details about the implementation on a real robot, and show a video and experimental data as evidence of the most prominent features.

2 Related Work

Zero moment point (ZMP) tracking with preview control [2] is the most popular approach to bipedal walking to date. A number of footsteps planned ahead are used to define a future ZMP reference, e.g. by placing the ZMP in the center of the footsteps and allowing for a smooth transition from one foot to the other during the double support phase. A continuous center of mass (CoM) trajectory that minimizes the ZMP tracking error is then generated in a Model Predictive Control [3] setting. As long as the actual ZMP stays well inside the support polygon, stable walking is guaranteed. Using ZMP preview control, high quality hardware [4,5,6] can walk reliably on flat ground. Next generation walking controllers from the ZMP preview family [7,8,9] also consider foot placement in addition to ZMP control either by including the footstep locations in the optimization process, or by using a simplified model to compute a footstep plan online. The approaches that have matured beyond a theoretical state require either precise physical modeling, the estimation of an impact force, or the feedback of a measured ZMP location. All of these requirements are difficult to meet for soccer robots.

Recently, Urata et al.[10] presented an impressive foot placement-based controller on a real robot that is capable of recovering from strong pushes. Instead of optimizing the CoM trajectory for a single ZMP reference, a fast iterative method is used to sample a whole set of ZMP/CoM trajectory pairs for three steps into the future. Triggered by a disturbance, the algorithm selects the best available footstep plan according to given optimization criteria. Resampling during execution of the footstep plan is not possible. The robot has to be able to track a fixed motion trajectory for the duration of the recovery. Specialized hardware was used for meeting the precision requirements.

Englsberger et al. [11] proposed using a capture point trajectory as reference input for gait generation instead of the ZMP. As the capture point can easily be computed from the CoM state, it is more suitable for state feedback than the ZMP. We incorporated the core ZMP control equation as one of the building blocks of our framework. The capture point approach is potentially suitable for soccer robots, but it does not consider adaptive foot placement and step timing.

Focusing on the methods that are applied by leading teams of the Humanoid League, it is notable that the preferred algorithms are simple and light-weight. Most of them are open-loop [12,13], sometimes with limited state feedback for posture control. Perhaps the most advanced closed-loop walk so far was presented for the Nao standard platform by Graf and Röfer [14], who proposed the online-adjustment of step parameters based on the solution of a system of linear pendulum equations. In the KidSize class, the DARwIn robot comes with a fast and reliable open-loop walk (Yi et. al. [15]). Parameterized ZMP and CoM trajectories are generated analytically using simple linear inverted pendulum model equations. Zhao et. al. [16] suggested an elegant open-loop gait generation technique inspired by passive walking down a shallow slope. This approach creates a virtual downwards slope by shortening the swing leg before support exchange and recharging energy by extending the leg again during the support phase.

The most closely related works published by the authors themselves are the Capture Step balance controller [1] and the open-loop central pattern generator [12] that were combined in this framework, and implemented on a real robot.

3 The Capture Step Framework

The Capture Step Framework is an omnidirectional bipedal gait generator with separable conceptual modules. The main software components are the state estimation, reference trajectory generation, balance control, and motion generation. The layout of the components is shown in Fig. 1. The framework simplifies the full-body dynamics of the robot to the trajectory of a single point mass, which is assumed to move like a linear inverted pendulum. In strong contrast to classic ZMP preview algorithms, motion trajectories are expressed directly for the center of mass and adaptive footstep locations arise as the output of the trajectory generation process. In each iteration of the main control loop, an ideal CoM reference trajectory is computed that depends only on the desired walking velocity. The state estimation component maps the current pose of the robot to the position and velocity of a point mass. Then, the balance control module computes a zero moment point offset and an estimated time for the next support exchange to steer the current state towards the reference trajectory while preserving balance with an adequate step size. The timing and the step location encode a step motion that is generated on a lower level without further concern for balance.

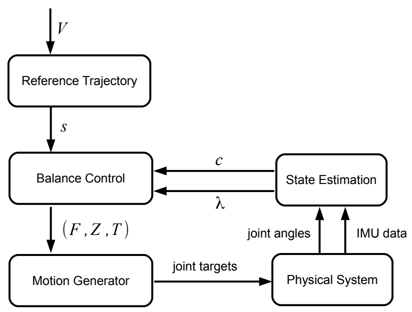

Fig. 1. Conceptual structure of our gait control architecture.

3.1 Reference Trajectory Generation

It is common practice to compose a pendulum motion by superposing two uncoupled one-dimensional linear inverted pendulum models. One model describes the lateral motion of a point mass and the other the sagittal motion. The two models are synchronized at a shared moment of support exchange, where each model is reset to a post-step state. Fig. 2 shows schematic trajectories for the lateral and the sagittal dimensions. There is an interesting conceptual difference between the two projections. In the sagittal dimension, the point mass crosses the pivot point in every step cycle. In the lateral dimension, however, the point mass oscillates between two supports and never crosses the pendulum pivot point.

We identify four parameters that characterize the walking motion. The lateral distance between the pivot point and the apex of the point mass trajectory is denoted as α. It is evident that the lateral component of the center of mass velocity equals zero in this point. As long as the apex distance is greater than zero, the point mass will return and is guaranteed to reach a support exchange location in a range bounded by δ and ω. When walking in place, we assume the support exchange to occur at the minimal distance δ. When walking with a nonzero lateral velocity, the walker first takes a long step with the leading leg and the support exchange occurs at a distance up to an upper bound ω, depending on the desired lateral walking velocity. The large leading step is followed by a small trailing step with a support exchange at δ. In the sagittal direction only one parameter is needed. σ defines an upper bound for the pass-through velocity of the point mass right above the pivot point. When walking in place, the pass-through velocity is zero, and it increases up to σ depending on the desired speed of locomotion. The chosen time for the support exchange is the moment when the CoM reaches its designated support exchange location between δ and ω. We assume that the support exchange happens instantaneously, and do not include a double support phase in our model. Finally, as the linear inverted pendulum is driven by the simple dynamic equation $\ddot{x} = C^2 x$, we regard C as a parameter that defines the gravitational effect on the point mass trajectory. We estimate the introduced parameters from our robot by having it walk in open-loop mode and averaging the trajectory apexes, the support exchange points, and the sagittal pass-through velocity.

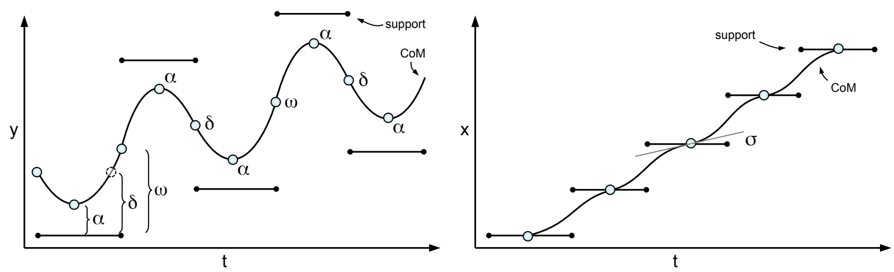

Fig. 2. The two-dimensional physical model is composed of a lateral motion (left) and a sagittal motion (right). The reference trajectory is described by four configuration parameters that define the lateral distance at the step apex (α), the minimal and maximal support exchange locations (δ and ω), and the maximum sagittal velocity at the step apex (σ).

We use this parameterized pendulum model to generate ideal reference trajectories for a permitted range of walking velocities. Since the reference trajectory follows the laws of the linear inverted pendulum model, a single end-of-step state s is sufficient to represent the entire trajectory. It defines a target state that the balance controller attempts to reach at the end of the step. The input into the gait control framework is a desired walking velocity vector $V = (V_x, V_y, V_\theta)$, $V \in [-\hat{V}_x, \hat{V}_x] \times [-\hat{V}_y, \hat{V}_y] \times [-\hat{V}_\theta, \hat{V}_\theta]$, with bounded components for the sagittal, lateral, and rotational directions. Let $\bar{V} = (\frac{V_x}{\hat{V}_x}, \frac{V_y}{\hat{V}_y}, \frac{V_\theta}{\hat{V}_\theta})$ be the componentwise normalized input velocity. Given the configuration parameters α, δ, ω, σ, and C, we compute the nominal support exchange state $s = (s_x, \dot{s}_x, s_y, \dot{s}_y)$:

$$s_x = \bar{V}_x \frac{\sigma}{C} \sinh(C\tau), \tag{1}$$

$$\dot{s}_x = \bar{V}_x \sigma \cosh(C\tau), \tag{2}$$

$$s_y = \begin{cases} \lambda\xi, & \text{if } \lambda = \operatorname{sgn}(V_y) \\ \lambda\delta, & \text{else} \end{cases}, \tag{3}$$

$$\dot{s}_y = \lambda C \sqrt{s_y^2 - \alpha^2}, \tag{4}$$

$$\xi = \delta + |\bar{V}_y|(\omega - \delta), \tag{5}$$

$$\tau = \frac{1}{C} \ln\left(\frac{\xi}{\alpha} + \sqrt{\frac{\xi^2}{\alpha^2} - 1}\right), \tag{6}$$

where $\lambda \in \{-1, 1\}$ denotes the sign of the support leg. The nominal state s is expressed in coordinates relative to the current support foot. Please note that ξ and τ express meaningful quantities. ξ is the lateral support exchange location for the leading step, interpolated between the minimal support exchange location δ and the maximal support exchange location ω. τ is the "half step time" that the CoM travels, starting at the lateral apex α with a velocity of zero to the support exchange location ξ.

3.2 State Estimation

The state estimation module aggregates measurements from the physical robot to estimate the current pendulum state $c = (c_x, \dot{c}_x, c_y, \dot{c}_y)$ expressed in coordinates relative to the current support foot, and the sign $\lambda \in \{-1, 1\}$ of the support leg. More precisely, the joint angle information obtained from the motor encoders is used to update a kinematic model using a forward kinematics algorithm. Then, the entire model is rotated around the center of the current support foot such that the torso inclination matches the angle measured by the IMU. When the vertical coordinate of the swing foot has a lower value than the vertical coordinate of the support foot, the roles of the feet are switched, and a footstep frame is set to the ground projection of the new support foot preserving the global orientation that the new support had in this moment. The footstep frame remains fixed until the next support exchange occurs. With respect to the footstep frame, we measure the coordinates of the ground projection of the point in the center between the hip joints and present them to a Kalman filter to obtain a smoothed pendulum state c. The relocation of the footstep frame at the support exchange introduces an unavoidable discontinuity in the pendulum state trajectory. To compensate for undesired effects on the Kalman filter, we reinitialize the filter with the first coordinates measured after the support exchange and a velocity vector that is rotated into the new support frame, such that the continuity of the velocity in the global reference frame is preserved.

By tracking a fixed point on the robot frame instead of the true center of mass, we not only avoid having to provide masses and inertias to construct a physical model, but we also exclude noise due to moving body parts. We abstain from using quantities that are difficult to measure, such as torques, forces, and accelerations. Environmental disturbances and the dynamic influences of body parts that are strong enough to change the trajectory of this fixed point will still result in an immediate reaction of the balance controller. There is no need to estimate an impact force, or any other magnitude of a disturbance.

Without attempting to be overly precise, we used the same general humanoid kinematic model that we used previously in simulation [1], and adjusted the lengths of the body segments to the measured lengths from the real robot. In this general humanoid model, all degrees of freedom in each joint intersect at a point as a simplification. This is clearly not the case with the real hardware.

3.3 Balance Control

Given the current pendulum state c and the desired end-of-step state s, the balance control module computes a time T for the support exchange, a zero moment point offset Z, and the footstep location F where the swing foot is expected to touch down. The concept of the balance module is illustrated in Fig. 3. The zero moment point offset Z is expressed relative to the ankle joint. It steers the center of mass towards the target state s during the current step. However, as the zero moment point is physically bounded to remain inside the support polygon, the effect of the zero moment point is limited and the target state is not guaranteed to be reached. Based on the estimated support exchange time T and the zero moment point offset Z, we predict the achievable end-of-step state c' and use it to compute the step coordinates F expressed with respect to the predicted state c'. For the computation of all of these parameters, analytic formulae are derived from the linear inverted

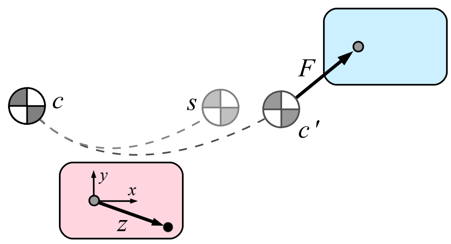

Fig. 3. Balance control computes a zero moment point offset Z that steers the center of mass c towards the nominal support exchange state s. The location of the next step F is computed with respect to the predicted achievable end-of-step state c'.

pendulum model in closed form. We compute the **lateral ZMP offset**

$$Z_y = \frac{s_y 2Ce^{C\check{T}} - c_y C(1 + e^{2C\check{T}}) + \dot{c}_y(1 - e^{2C\check{T}})}{C(e^{2C\check{T}} - 2e^{C\check{T}} + 1)} \tag{7}$$

in a way that it attempts to reach the lateral support exchange location s_y (3) at the nominal step time $\check{T}$ of an ideal step and helps to maintain a desired step frequency. Z_y has to be bounded to a reasonable range, for example the width of the foot, and thus a fixed frequency cannot be guaranteed. We set $\check{T} = 2\tau$ (6) whenever a support exchange occurs and decrement it by the iteration time of the main control loop (in our case 12 ms) with every iteration. Please note that $\check{T}$ can have a negative value if the nominal step time is exceeded.

Due to an observed sensitivity of the lateral oscillation to disturbances [17], we attribute the computation of the predicted **step time** T entirely to the lateral direction. We want the support exchange to occur when the CoM reaches the nominal lateral support exchange location s_y. Taking the bounded lateral ZMP offset into account, T is given by

$$T = \frac{1}{C} \ln \left(\frac{s_y - Z_y}{c_y - Z_y + \frac{\dot{c}_y}{C}} + \sqrt{\frac{(s_y - Z_y)^2}{(c_y - Z_y + \frac{\dot{c}_y}{C})^2} - \frac{c_y - Z_y - \frac{\dot{c}_y}{C}}{c_y - Z_y + \frac{\dot{c}_y}{C}}} \right). \tag{8}$$

However, there are two cases where the step time cannot be clearly determined. When after a strong disturbance the CoM is moving towards the pivot point in the lateral direction and is in danger of crossing it, the lateral orbital energy $E_y = \frac{1}{2}(\dot{c}_y^2 - C^2 c_y^2)$ is positive. In this case, we use a large constant time, e.g. 2 seconds, to "freeze" the robot and hope that it will return after all. The other case is when the support exchange location has already been crossed in the past, or will never be crossed due to a large disturbance. In this case it is advisable to step as soon as possible and the step time should be derived from the maximum allowed step frequency that the robot can handle. For the computation of the **sagittal ZMP offset**

$$Z_x = \frac{s_x + \frac{\dot{s}_x}{C} - e^{CT}(c_x + \frac{\dot{c}_x}{C})}{1 - e^{CT}}, \tag{9}$$

we use the capture point based formula proposed by Englsberger et al. in [11]. It computes the sagittal ZMP such that if the CoM continues to move along an optimal linear inverted pendulum trajectory, the ZMP stays constant for the remainder of the step and the capture point of the CoM will match the capture point of the target state s by the time T of the support exchange. Since it is not possible to fulfill three constraints: the location, the velocity, and the time, with one constant ZMP offset per step, we opt for the simplicity of this good approximation. Finally, the sagittal ZMP offset also has to be bounded to a reasonable range.

Please note that both dimensions of the ZMP offset have been calculated without direct feedback of a measured zero moment point location. Only the position and the velocity of the center of mass have been used, which are easy to obtain. Given the bounded ZMP offset Z and the step time T, we can now compute the estimated achievable end-of-step state c'

$$c'_x = (c_x - Z_x)\cosh(CT) + \frac{\dot{c}_x}{C}\sinh(CT), \tag{10}$$
$$\dot{c}'_x = (c_x - Z_x)C\sinh(CT) + \dot{c}_x\cosh(CT), \tag{11}$$
$$c'_y = (c_y - Z_y)\cosh(CT) + \frac{\dot{c}_y}{C}\sinh(CT), \tag{12}$$
$$\dot{c}'_y = (c_y - Z_y)C\sinh(CT) + \dot{c}_y\cosh(CT). \tag{13}$$

The **footstep location** is then given by

$$F = (\frac{\dot{c}'_x}{C}\tanh(C\tau), \lambda\sqrt{\frac{\dot{c}'^2_y}{C^2} + \alpha^2}). \tag{14}$$

In the sagittal direction we compute the nominal step size that would result in the same end-of-step CoM velocity as the predicted one. In the lateral direction, the footstep location is computed with an extended capture point formula so that the CoM will pass the apex of the next step at distance α with a velocity of zero. Note that the footstep location F is expressed with respect to the future CoM state c'. It can be trivially converted to a foot-to-foot step size $S = F + (c'_x, c'_y)$.

3.4 Motion Generator

The hierarchical layout of the Capture Step Framework allows us to interface with virtually any walking motion generator that can exhibit control of stepping motions using step size and timing parameters. In this work, we use a central pattern generated gait (CPG) [12] that has been used in competition games with repeated success. The CPG can be combined with compliant actuation and provides a certain amount of open-loop stability out of the box. Instead of relying on inverse kinematics and end-effector trajectories in Cartesian space, the CPG operates in an abstract actuation space that makes it easy to produce stretched-knee walking.

For the integration of the motion generator, the output of the balance control module has to be transformed to gait control parameters in order to produce the desired step sizes and timings on the physical robot. The CPG expects a walking velocity control vector $V \in [-1, 1]^3$ with parameters for the sagittal, lateral, and rotational directions. Essentially, the velocity input parameters result in step amplitudes on the real robot. To map the output step size of the balance controller to the velocity input space of the CPG, we use the CPG in open-loop mode to generate data that describe the velocity control to step size mapping as measured by the sensors of the real robot and approximate it with a linear function. Then, we convert the balance control output step size S to a CPG velocity input V using the inverse of the linear approximation. The rotational component of the vector is ignored by the balance control layer. We simply pass the normalized rotational velocity input $\bar{V}_\theta$ through to the motion generator. To map the step time T to the motion phase of the CPG, we compute a phase increment such that the gait phase induces a support exchange at time T in the future. As T approaches zero when the support exchange is imminent, the computation of the phase increment becomes increasingly unstable. It is advisable to inhibit the timing adaptation near the support exchange and gait frequency bounds must be used to filter numerically unstable cases. The most significant innovation of the CPG integration is the fact that we did not link the inverted pendulum modeled CoM motion directly with the pelvis of the physical robot. This is typically done using inverse kinematics in conventional plane-restricted ZMP-preview walkers. This simplification was not only found to increase stability, but also allows for a non-level CoM motion simply by not forbidding it. Consequently, the ZMP offset is not explicitely transformed to a motion component, but since it is responsible for the step size variation, the physical system still reflects the commanded ZMP by increasing or decreasing the step size accordingly.

4 Real Robot Implementation

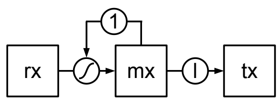

Fig. 4. A predictive noise filter is used to smooth the CoM state estimation and to predict a future state at the latency horizon.

When dealing with real hardware, sensor noise and control loop latency are quite significant compared to simulation. An integral component of the real robot implementation is a predictive noise filter illustrated in Fig. 4. The filter smoothes the CoM trajectory using model-based assumptions, and predicts a short-term future state to overcome the latency. The first building block, denoted “rx”, is the output of the state estimation, as it was described in Section 3.2. The second building block, denoted “mx”, is the model state. In every iteration, the model state is forwarded by one time frame using the laws of the linear inverted pendulum model. The forwarded model is then linearly interpolated with the rx state using a blending factor $b \in [0, 1]$. The result is written into the new mx state, and forwarded in time by the latency l to compute a “tx” state at the latency horizon. The tx state is then presented to the balance controller for further processing. We have determined a latency $l = 65\,\text{ms}$ on our real hardware. This is quite significant considering that one step is approximately $420\,\text{ms}$ long. The unfortunate implication is that the support exchange has to be induced, and the first portion of the motion signal for “the other leg” has to be sent out, well before the change of the leg sign λ is detected by the kinematic model. This is achieved by forcing the tx model to step if the latency l exceeds the estimated step time T. Stepping is performed by setting the position of the tx state to $-F$ and resetting the step time to $T = 2\tau$. When the step time T reaches a negative value, the mx model itself is stepped, whether the real support exchange has been detected or not. This means that at times near the support exchange, the rx state and the mx state may assume different support leg signs and cannot be blended. In this case we set $b = 0$ and essentially switch to open-loop mode, where the mx model state does not receive any sensor feedback for a while and computes a linear inverted pendulum

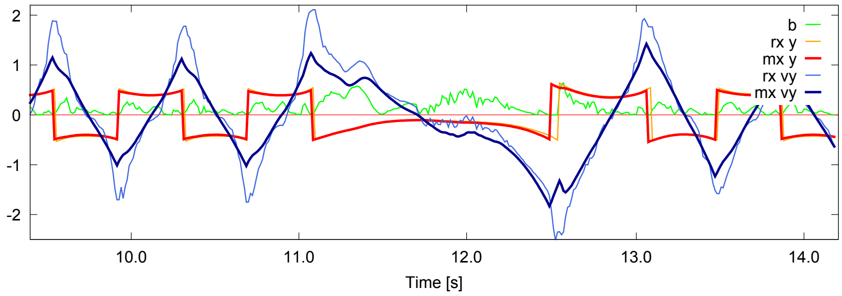

Fig. 5. Demonstration of the effects of the predictive filter.

simulation from the last known state until the blending gate can be opened again. Independently of this, we utilize two additional mechanisms to adjust the value of the blending factor, and to control the behavior of the predictive filter. Firstly, we inhibit adaptation shortly before and after the support exchange by smoothly decreasing b when the step time T approaches zero and allowing it to increase again after the support exchange. And secondly, we decrease the value of b when the Euclidean distance between the rx and the mx model is small. This way we avoid jitter when the model state matches the measured input well, but do not sacrifice system response when the model state deviates significantly from the measured state. Fig. 5 shows the lateral pendulum position and velocity recorded during an experiment. While the robot was walking on the spot, it was pushed from the side shortly after the time 11:0. The smoothing effect of the predictive filter can best be seen by comparing the rx and mx velocity data. The filter discards the high velocity peaks at the support exchange that differ strongly from the pendulum model and lead to bad predictions. The blending factor shows its highest peaks right after the support exchanges and after the push. The model adapts nicely to the new pendulum trajectory caused by the push, but eliminates the jittery noise shortly before the time 12:0. At the first step after the push, the rx and mx signals are slightly out of synchronization. The mx model steps earlier than the robot, but synchronization is quickly restored.

5 Experimental Results

We have implemented the Capture Step Framework on a bipedal robot with a weight of 7.5 kg and a height of 107 cm. TeenSize soccer robot Dynaped has demonstrated improved walking capabilities during a technology demonstration at the GermanOpen in Magdeburg in April 2014, as can be seen in the video [18], Fig. 7. Dynaped is equipped with a low cost, two-axis inertial measurement unit and Dynamixel EX-106 servo motors that we operate in a compliant mode. This makes the walk elastic and smooth, but also imprecise. We run the gait generation process with an update frequency of 83.3 Hz. Each iteration requires a computation time of 0.12 ms on a 1.3 GHz single core CPU. Fig. 6 shows experimental data from the Dynaped robot. The experiment started with the

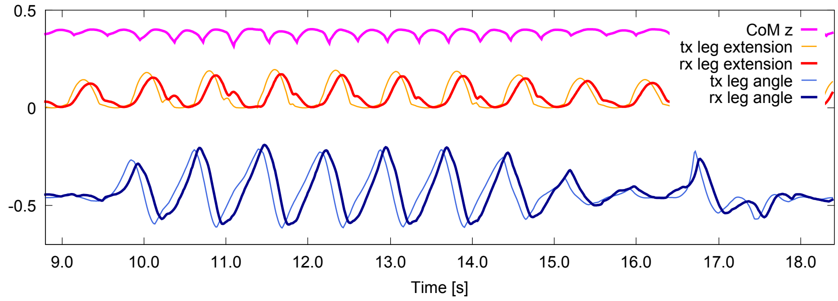

Fig. 6. CoM height, leg extension and leg angle during walking.

robot walking in place. Then the robot accelerated and walked forward with its maximum velocity for approximately four seconds. When the robot came to a stop, it was immediately pushed from the back. The plot displays the CoM height on the top, and demonstrates the non-planar motion of the CoM. The data stream in the center shows the actuation signal (tx) for the leg extension as well as the signal received back from the robot (rx). The compliant actuation can be seen in the moments of the floor contact, where the swing leg automatically absorbs the impact force and the received signal deviates strongly from the actuation signal. The data stream in the bottom shows the motion signal of the leg angle. There is an evident delay between the activation signal and the received signal.

6 Conclusions

We have contributed a robust omnidirectional gait generation method that is composed of an open-loop central pattern generator and a linear inverted pendulum based balance controller. Even though the balance controller is simplified to a point mass model, the controller is able to recover from disturbances that are strong enough to tilt the robot into an oblique pose using analytically computed step timing and foot placement adaptation. The direction or the magnitude of a disturbance does not need to be sensed. Our method can operate in a high latency environment with imprecise actuation using no more than an attitude sensor, joint position feedback and an inaccurate kinematic model. At the same time, common restrictions are lifted, such as bent knees, planar center of mass motion, and ground-aligned feet. We demonstrated the capabilities of our approach in a

Fig. 7. Dynaped regaining balance after a push from the back by stepping forward [18].

public presentation on a real robot. Its low requirements and robustness make the Capture Step Framework an ideal candidate to be implemented on humanoid soccer robots. In future work, we are planning to incorporate additional control laws into our balance controller to cope with the effects of angular momentum. Furthermore, we are investigating the possibility of applying machine learning algorithms to improve the efficiency of the capture steps.

References

1. M. Missura and S. Behnke. Omnidirectional capture steps for bipedal walking. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2013.
2. S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, and K. Yokoi. Biped walking pattern generation by using preview control of zero-moment point. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2003.
3. P.-B. Wieber. Trajectory free linear model predictive control for stable walking in the presence of strong perturbations. In *Humanoids*, 2006.
4. K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka. The development of Honda humanoid robot. In *ICRA*, 1998.
5. S. Kajita, M. Morisawa, K. Miura, S. Nakaoka, K. Harada, K. Kaneko, F. Kanehiro, and K. Yokoi. Biped walking stabilization based on linear inverted pendulum tracking. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2010.
6. I.-W. Park, J.-Y. Kim, J. Lee, and J.-H. Oh. Mechanical design of humanoid robot platform KHR-3 (KAIST humanoid robot 3: HUBO). In *Humanoids*, 2005.
7. H. Diedam, D. Dimitrov, P.-B. Wieber, K. Mombaur, and M. Diehl. Online walking gait generation with adaptive foot positioning through linear model predictive control. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2008.
8. M. Morisawa, F. Kanehiro, K. Kaneko, N. Mansard, J. Sola, E. Yoshida, K. Yokoi, and J.-P. Laumond. Combining suppression of the disturbance and reactive stepping for recovering balance. In *IROS*, 2010.
9. B. J. Stephens and C. G. Atkeson. Push recovery by stepping for humanoid robots with force controlled joints. In *Humanoids*, 2010.
10. J. Urata, K. Nishiwaki, Y. Nakanishi, K. Okada, S. Kagami, and M. Inaba. Online decision of foot placement using singular LQ preview regulation. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2011.
11. J. Englsberger, C. Ott, M. A. Roa, A. Albu-Schäffer, and G. Hirzinger. Bipedal walking control based on capture point dynamics. In *IROS*, 2011.
12. M. Missura and S. Behnke. Self-stable omnidirectional walking with compliant joints. In *Workshop on Humanoid Soccer Robots*, Atlanta, USA, 2013.
13. S. Behnke. Online trajectory generation for omnidirectional biped walking. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2006.
14. C. Graf, A. Härtl, T. Röfer, and T. Laue. A robust closed-loop gait for the standard platform league humanoid. In *Workshop on Humanoid Soccer Robots*, 2009.
15. 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 *Humanoids*, 2011.
16. H. Dong, M. Zhao, and N. Zhang. High-speed and energy-efficient biped locomotion based on virtual slope walking. *Autonomous Robots*, 30(2), 2011.
17. M. Missura and S. Behnke. Dynaped demonstrates lateral capture steps. `http://www.ais.uni-bonn.de/movies/DynapedLateralCaptureSteps.wmv`.
18. M. Missura and S. Behnke. Walking with capture steps. `http://www.ais.uni-bonn.de/movies/WalkingWithCaptureSteps.wmv`.