The 10th Workshop on Humanoid Soccer Robots
15th IEEE-RAS International Conference on Humanoid Robots, Seoul, Korea, November 2015

Analysis and Compensation of Biped Walking Disturbances Caused by Model Abstractions

Oliver Urbann, Jan Hendrik Berlin, Matthias Hofmann, Ingmar Schwarz
Robotics Research Institute
Section Information Technology
TU Dortmund University

***Abstract*—In this paper, we identify and analyze reasons for instabilities in closed-loop walking control. We show how abstractions from the physical robot to models, or from complex models to simplified models in simulation influence the performance of the walking motion. Furthermore, we present new algorithms for compensation of these disturbances. This way they can be lowered in advanced to reduce dependence on sensor feedback stabilization.**

I. Introduction

Biped locomotion is a popular research topic as it provides enhanced mobility of robots in nature (e.g. forest) or in environments designed for humans. However, usability of biped robots is limited due to stability issues that are still an ongoing research topic. There are numerous reasons for that, ranging from external disturbances like obstacles, humans or other robots to internal reasons, e.g. simplified robot models or gear errors. Both classes of errors must be handled differently. External disturbances cannot be foreseen and therefore require active balancing in contrast to internal errors. In theory, all these errors can be integrated into the model, although this is in fact almost impossible.

The contribution of this paper is the analysis of internal reasons for inaccuracies in biped locomotion. We reveal general causes of internal disturbances to be helpful for various researchers and present new algorithms to solve the exposed causes. This paper is independent from specific walking algorithms as we concentrate on the model and not on the way it is utilized to generate the motion. However, our analysis is based on the approach presented at the beginning of the next section.

A. Related Work

Many cutting-edge walking algorithms are based on the 3D linear inverted pendulum mode proposed by Kajita et al. [1] or similar approaches [2]. In contrast to the prediction of the simple model, the walk of a physical robot is less stable. It is common sense that the abstraction from a real robot to a simple model is a major reason for deviations. As a result, numerous researchers propose various extensions to overcome specific flaws in the simple single center of mass (CoM) abstraction. Pratt et al. [3] propose to include a flywheel in the model of the robot to calculate dynamics of a tilting robot. Seven et al. [4] propose utilizing a second CoM for the swinging leg. Buschmann et al. [5] propose to include a third CoM for both legs. Kajita et al. [6] measure the difference between a simulated robot and the predicted dynamics of the 3D-LIPM to compensate the Zero Moment Point (ZMP)[1] error. While this is in fact not an improved model, it corrects the errors of the simple model by utilizing a rigid body simulation and is therefore far less abstract. However, he applied a second preview controller for this compensation wherefore it can only be done offline or online with at least one step delay in case of a change in speed or direction.

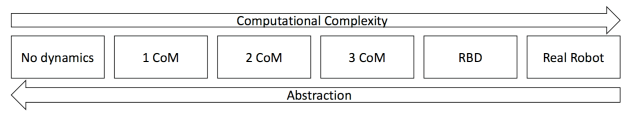

Fig. 1. Various levels of abstraction are known for motion generation to overcome computational complexity. ”No dynamics” denotes motions with manually optimized foot trajectories or similar approaches. The simplest known model for a robot consists of one CoM. More complex models were proposed to include the leg dynamics. RBD is usually utilized by simulators although it is a strong abstraction compared to a real robot.

A model of a robot is always an abstraction of a physical robot but nevertheless utilizable to generate stable walking motions. However, as discussed by Urbann et al. [7], it might require some undesired choices, e.g. a particular step height or a ZMP that is not only inside the support polygon but also of special shape. The latter is one approach of this paper for walking stabilization.

B. Overview

Fig. 1 depicts the discussion of the latter paragraph about various levels of abstractions. The most simple case of physical dynamics is in fact ”no dynamics”. Walking algorithms utilizing static foot trajectories are examples for this category. The more CoMs are used, the better is the reflection of the reality. A sophisticated model is the Rigid Body Dynamics (RBD), e.g. as utilized by the Open Dynamics Engine (ODE)[2]. Here, the CoMs of all links are included along with their inertia matrices and the simulation is able to reflect frictions forces between objects. Joints in ODE can also be torque-driven, which enables the realization of PID controllers. This can be

[1] The Zero Moment Point is the point on the floor where all tipping moments of the robot are zero. If it resides inside the support polygon, the robot is considered as stable.

[2] www.ode.org

utilized by simulations to reduce the gap between RBD and a real robot as done in Gazebo [8]. However, ODE or any simulator can not be a replacement for a model in a motion algorithm.

Between a real robot and RBD is still a big gap, which motivated Urbann et al. to propose the MoToFlex simulator [9]. We review its features in section II and utilize it to evaluate the influence of PID controller, worn out gears, slightly flexible links and the joint motor on the real walk in section III. In this section we additionally compensate the detected instabilities with various algorithms. Section IV concludes this work.

II. Review of MoToFlex

MoToFlex [9] is intended to fill the gap between a simulation based on RBD, e.g. the ODE and real robots. As the name implies, an RBD simulates rigid bodies only. All joints do not reveal any angle errors as they do not include a motor model or a PID based controller like the joints of the Nao by Aldebaran Robotics. MoToFlex therefore realizes the following features (abbreviations are utilized in the following to indicate if a feature is activated or not):

- A motor model **M** with PID controller, limited voltage, impedance **I**, resistor and friction,
- Gears **G** with mass, flexibility and tolerances,
- Flexible bodies **F** to reflect lightweight links of a robot.

The simulation is optimized by Evolutionary Algorithms to fit a physical robot. This also leads to an improved simulation when no additional feature of MoToFlex is utilized, i.e. a basic ODE simulation is executed. With all features activated it can be shown in multiple experiments that MoToFlex is able to reflect various instabilities of a walking robot [9].

III. Analysis and Compensation of Oscillation

In this section we investigate the difference between a model consisting of a single CoM and RBD. We present an algorithm to compensate the error, and are then able to inspect the deviations caused by motors, PID controllers, gears, and flexible links.

A. Setup

To generate a walking motion, the ZMP/IP controller as proposed by Urbann et al. is applied [10], [11], [12], [13]. It utilizes the 3D-LIPM and is therefore representative for walking motions based on a single CoM abstraction. We analyze two observations when walking at 20cm/s with a physical robot: First an oscillation of the trunk during the walk, and second the actual speed is higher than desired, approx. 25 cm/s. The walk is overall stable in terms of it does not fall down. However, the oscillation is not intended and should not occur.

As Fig. 2 depicts, the oscillation of this walk can be reproduced in MoToFlex with comparable amplitude and same frequency. Additionally, we can seen both, in simulation and reality, every second step the amplitude is lower. The same walk with all additional features of MoToFlex switched off (basic ODE simulation) leads to approximately no oscillation. This is further analyzed in section III-D.

Moreover, the robot walks in MoToFlex 21 cm/s which is higher than the desired and expected speed of 20 cm/s. We therefore utilize MoToFlex to find a root cause for the speed increase of the physical robot in the next section.

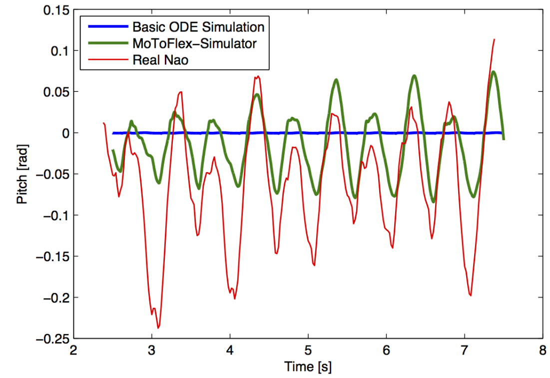

Fig. 2. Oscillation during a walk straight forward at 20 cm/s. While a simulation based on RBD (e.g. ODE) reveals no oscillation during the evaluated walk, MoToFlex reflects the oscillations of the physical robot in amplitude and frequency. Additionally, every second step the amplitude is smaller during the walk of a physical robot which is also reflected by MoToFlex.

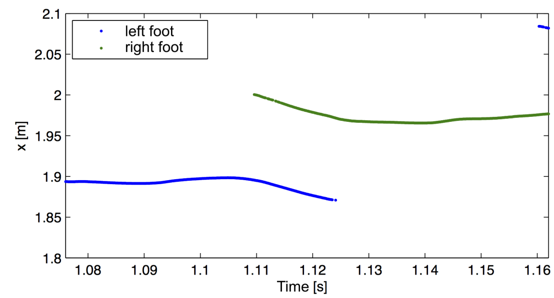

Fig. 3. Positions of feet on ground over time. The position is plotted only in case a ground contact is detected.

B. Coupling of Oscillation and Speed-Up

Fig. 3 illustrates the positions of both feet during one step. If a foot has contact to the ground, a dot marks its position. As can be seen, in the double support phase (both dots are drawn) both feet are slipping backwards. This lowers the actual speed. In Fig. 4 a dot that marks the distance between both feet is drawn if and only if both feet have contact to the ground. As

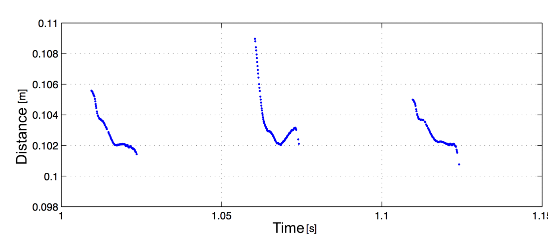

Fig. 4. Distance between feet if both feet have ground contact.

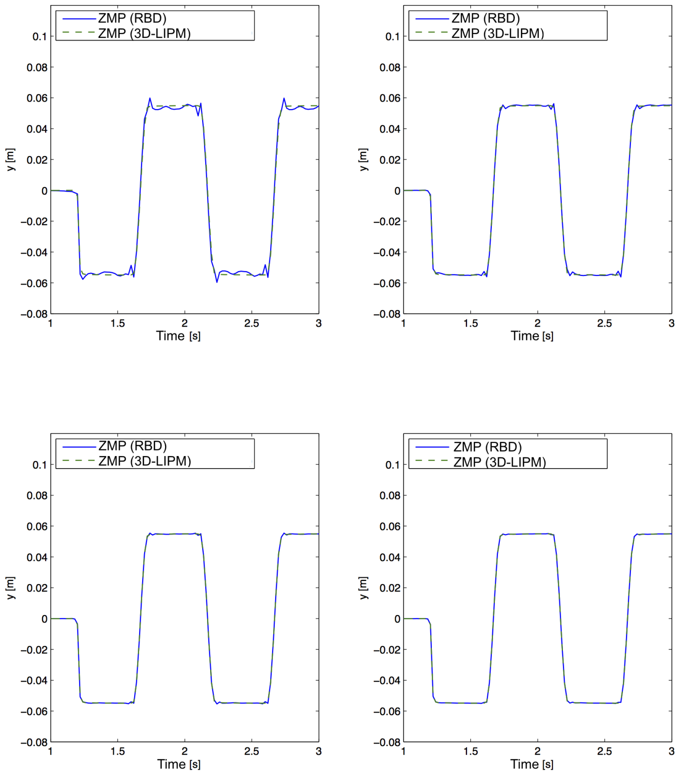

Fig. 5. ZMP deviations between RBD and 3D-LIPM after multiple iterations of Alg. 1.

can be seen, the distance decreases during the double support but is higher than the desired distance of 0.1m. Due to this step size increase the speed of the robot is increased. As this effect is larger than the speed decrease due to slipping the overall speed of the robot is increased as mentioned in section III-A.

C. RBD

Algorithm 1 Compensation of abstraction RBD to 3D-LIPM

$p' = p_r, p_o = 0$
for i=1 to n **do**
  q = ZMP/IP-Controller(p'); // Execute walking algorithm to retrieve target angles q
  p = Newton-Euler(q); // Calculate ZMP using RBD (e.g. Newton-Euler)
  $p_o = p_o + p_r - p$; // Add difference to offset p_o
  $p' = p_r + p_o$; // Get new desired ZMP by adding offset to reference
end for

First part of the analysis is to evaluate the influence of the simplification to one CoM on the ZMP. While in theory the robot walks stable as long as the ZMP does not leave the support polygon, the combination of various instabilities can cause a fall down. It is therefore reasonable to improve all possible sources for ZMP deviations, here the simplification to a single CoM.

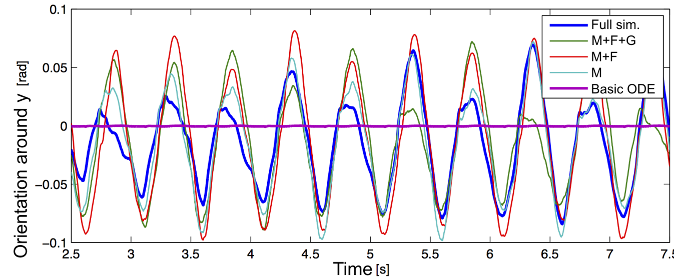

Fig. 6. Simulation of walk at 20 cm/s with various features of MoToFlex enabled.

We apply the compensation algorithm shown in Alg. 1. It is an accumulation of the difference between the desired ZMP p_r calculated by 3D-LIPM[3] and RBD p. This sum is afterwards added to the modified ZMP p' and the walk is recomputed. This is reiterated until the ZMP according to RBD matches the desired as depicted in Fig. 5. Afterwards p' is applied as new reference ZMP.

D. Simulation

The motors, gears and flexible bodies of MoToFlex cause the difference of the ZMP between RBD and a physical robot. As the measurement of the ZMP in simulation is problematic, we continue to analyze the oscillation. Fig. 6 depicts the oscillation with different features of MoToFlex switched on or off. The oscillation with all features (**M+F+G+I**) is comparable to Fig. 2 and comparable to the simulation of just **M**. It can be concluded that the parameters of the simulation found by the Evolutionary Algorithm are optimized such that **F+G** cancels out the effect of **I**. Without **I** but with **F+G** the phase of the oscillation is shifted about 0.5 s but still high. As the simulation of **F** has a low influence as well, **M+G** is solely responsible for the oscillation. Moreover, the gains for **G** in simulation are near to zero[4] and therefore it follows that almost only the PID controllers are responsible for the oscillation. Additionally, Fig. 7, 8 and 9 illustrate that higher desired torques lead to larger deviations in the direction of the torque. Consequently, PID controllers and the abstraction to a single CoM are an important cause for internal errors of a walking robot and are responsible for typical body oscillations in walking direction. However, it has to be noted that this holds for the simulation, which is optimized to match the walk of a physical robot, although this is not a guarantee that the results are transferable. Nevertheless, while the results are possibly obvious, a comprehensive analysis is the base for possible solutions presented in the next section.

E. Compensation on Physical Robot

In this section we present compensation methods comparable to the algorithm 1 but here for a physical robot. All experiments were done while walking forward only, as the

[3]The ZMP based on 3D-LIPM is equal to the desired ZMP as the walk is determined utilizing this abstraction.

[4]Gains were optimized by Evolutionary Algorithms to match the real walk.

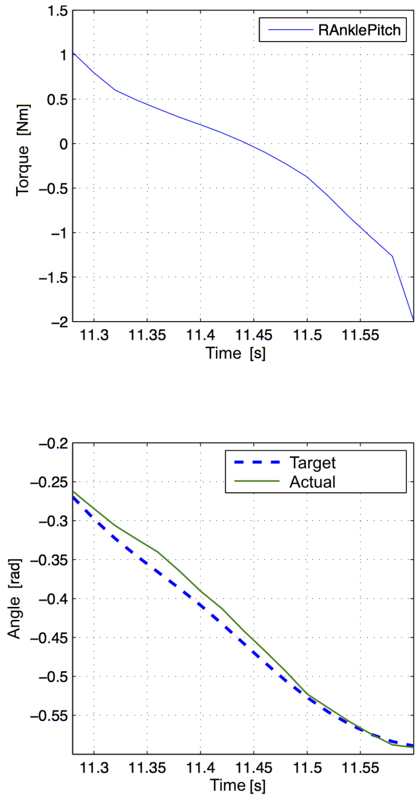

Fig. 7. Target angle of right ankle pitch and the required torque. The deviation between actual and target angle is coupled to the torque required to achive the movement.

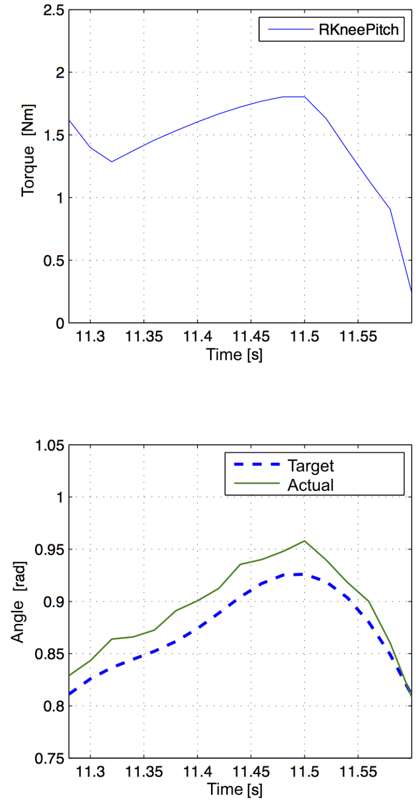

Fig. 8. Target angle of right knee pitch and the required torque. The deviation between actual and target angle is coupled to the torque required to achive the movement.

oscillation is not visible during rotation or a walk sideways. The results from the previous section gives a strong hint to the reason for this: with larger step sizes the torque on the foot joint raises and therewith the angle error leading to an oscillation of the body. This behavior is not included in the model resulting in a gap between the model and the physical robot.

The idea is to minimize the deviations between the simplified model and the more complex system. In contrast to the previous section, the complex system is the real robot that reveals the oscillation as shown in Fig 2. As it is impossible to predict the physical behavior of the real robot, the compensation must rely on measurements and adapt the desired ZMP accordingly.

Algorithm 1 performs well in compensation of the abstraction from RBD to 3D-LIPM. The idea of the algorithm here is therefore comparably. The sum of the difference between desired and actual ZMP is added to the desired ZMP of the next step, see Algorithm 2. A filter with gain α_1 is applied to mitigate bias from the measurements. Afterwards the difference is calculated and added to the sum. A second gain α_2 is utilized to avoid harsh reactions. After one step (here 50 time frames[5]) the sum for the specific time frame is added to the desired ZMP.

[5]One time frame has a duration of 10ms.

Algorithm 2 Compensation of measured ZMP deviations with bias filter

Init offset array $p_o[1 \ldots T]$ and b with 0
for i=1 to n **do**
  q = ZMP/IP-Controller(p'); // Execute walking algorithm to retrieve target angles q
  p = Walk(q); // Measure ZMP from walking robot
  $b = \alpha_1 b + (1-\alpha_1)(p_r - p)$; // Calculate bias b
  $p' = p_r + p_o[i \bmod T + 1]$; // Get new desired ZMP by adding offset to reference
  $p_o[i \bmod T + 1] = \alpha_2 p_o[i \bmod T + 1] + (1-\alpha_2)p_r - p + b$; // Add difference to offset p_o and remove bias
end for

Fig. 10 depicts the outcome of this algorithm. The measured ZMP (actual ZMP) shows large deviations compared to the original desired ZMP (not depicted) which is a linear function with positive slope. The compensation (Algorithm 2) is applied resulting in the shown target ZMP which is calculated as intended. However, the measured ZMP of each step is similar to the previous step. We conclude that a modified ZMP derived from measurements does not appropriately match the deviations and present in the following a different approach.

Since the dynamics of a periodical oscillation is mainly supplied from the dynamics of the previous oscillation, it is sufficient to stop a tilt into one direction. The tilt into the other

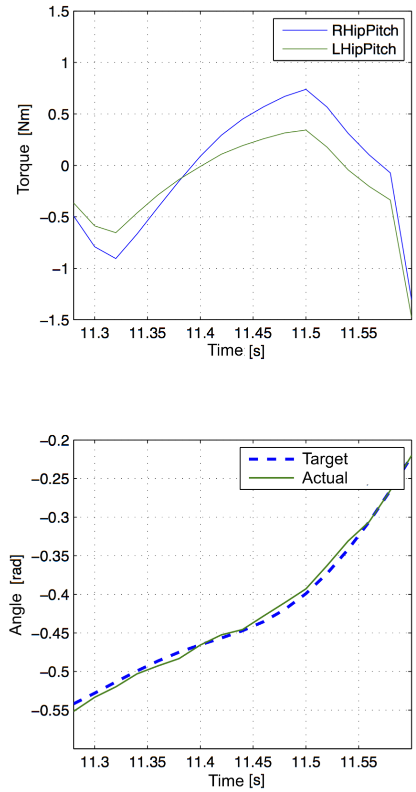

Fig. 9. Target angle of right hip pitch and the required torque. The deviation between actual and target angle is coupled to the torque required to achive the movement.

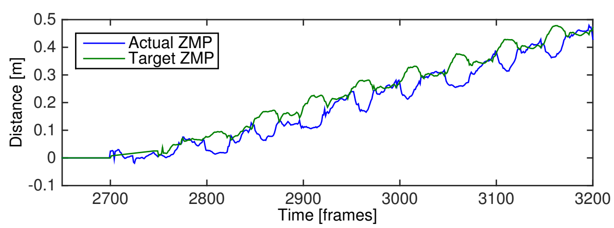

Fig. 10. Measured and desired ZMP after modification on a physical robot to reduce the error of the actual ZMP.

direction should then be canceled as well. Hence, we focus on the oscillation instead of the measured ZMP and apply two different offset functions to the desired ZMP: A sawtooth and a half sine. Fig. 11 depicts the actual ZMP of an unmodified walk with a linear ZMP function as reference. The form is comparable to a half sine which is therefore the modifier for the desired ZMP, resulting in the target ZMP as depicted here. As a result, the actual ZMP after modification is closer to a linear function. The sawtooth function is applied analogously. The frequency of the modifier (half sine and sawtooth) is coupled to the step duration. The phase of the modifier must be optimized manually.

Fig. 12 illustrates the results in the frequency domain. The largest amplitude can be measured at 2 Hz. We therefore concentrate our optimization to reduce the oscillation at this

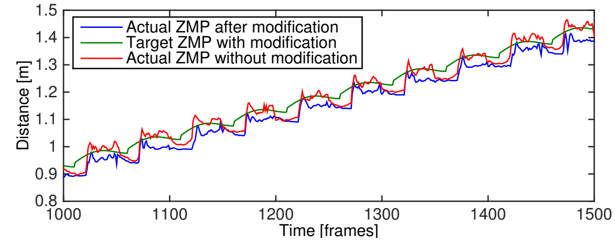

Fig. 11. Desired ZMP modified by a half sine function with impact on the measured ZMP.

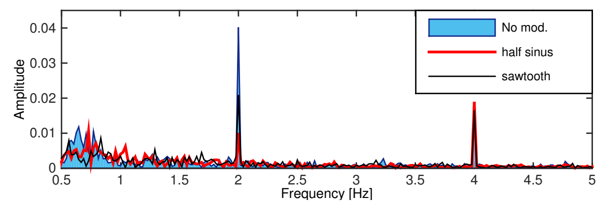

Fig. 12. Spectrum of the oscillation of a physical robot without and with two different modifications of the desired ZMP. The relevant peak of the oscillation is at 2 Hz.

frequency. Both functions (half sine and sawtooth) perform well, while the half sine shows the best results. As Fig. 13 depicts, the standard deviation of the oscillation is reduced as well, or in other words, the oscillation is successfully lowered.

Nevertheless, even with compensation a significant oscillation is not avoidable. Various reasons are possible:

- The form of the compensation function (half sine, sawtooth) does not fit exactly the error.
- The amplitude is not matched or varies over time.
- The error is a result of the limited torques of the joints.

In the latter case a compensation this way is impossible as requesting higher torques will not lead to higher torques, actually.

IV. Conclusion

This paper presents an analysis and compensation of instabilities with internal reasons, primarily caused by model abstractions. The analysis presents inaccuracies in multiple steps from a simple model containing only one CoM to a physical robot. The most problematic elements are the

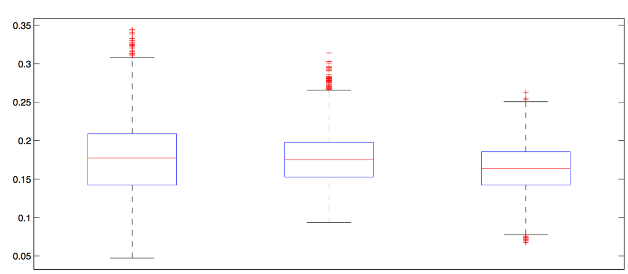

Fig. 13. Box plot of the oscillation of a physical robot without and with two different modifications.

abstraction from RBD to a single CoM and additionally PID controllers. The latter actually create torques without regard to the desired dynamic that is only known to upper layers of control.

Possible compensations are presented and compared. The compensation for the RBD to 3D-LIPM abstraction performs as expected while the real robot can be stabilized by applying a sawtooth or half sine function to the desired ZMP.

The compensation based on measurements failed as it only depends on the measured ZMP. However, both manually applied functions showed satisfying results. Therefore a future research topic is the automation of the manual application of arbitrary functions. A promising approach are neural networks learning to predict the ZMP based on the measurements. As neural networks can be applied in linear controller algorithms like Model Predictive Control, this way the model includes multiple aspects besides the CoM.

REFERENCES

[1] Kajita, S., Kanehiro, F., Kaneko, K., Yokoi, K., Hirukawa, H.: The 3D linear inverted pendulum mode: a simple modeling for a biped walking pattern generation. In: Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on. Volume 1. (2001) 239 –246 vol.1

[2] Graf, C., Röfer, T.: A closed-loop 3D-LIPM gait for the RoboCup Standard Platform League humanoid. In Pagello, E., Zhou, C., Behnke, S., Menegatti, E., Röfer, T., Stone, P., eds.: Proceedings of the Fifth Workshop on Humanoid Soccer Robots in conjunction with the 2010 IEEE-RAS International Conference on Humanoid Robots, Nashville, TN, USA (2010)

[3] Pratt, J.E., Carff, J., Drakunov, S.V., Goswami, A.: Capture point: A step toward humanoid push recovery. In: Humanoids, IEEE (2006) 200–207

[4] Seven, U., Erbatur, K.: An inverted pendulum based approach to biped trajectory generation with swing leg dynamics. In: 2007 7th IEEE-RAS International Conference on Humanoid Robotics. (2007) 44

[5] Buschmann, T., Lohmeier, S., Bachmayer, M., Ulbrich, H., Pfeiffer, F.: A collocation method for real-time walking pattern generation. In: Humanoid Robots, 2007 7th IEEE-RAS International Conference on. (Nov 2007) 1–6

[6] Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., Hirukawa, H.: Biped walking pattern generation by using preview control of zero-moment point. In: ICRA, IEEE (2003) 1620–1626

[7] Urbann, O., Schwarz, I., Hofmann, M.: Flexible linear inverted pendulum model for cost-effective biped robots. In: Humanoid Robots (Humanoids), 2015 15th IEEE-RAS International Conference on. (Nov 2015) to appear

[8] Koenig, N., Howard, A.: Design and use paradigms for gazebo, an open-source multi-robot simulator. In: Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on. Volume 3., IEEE (2004) 2149–2154

[9] Urbann, O., Kerner, S., Tasse, S.: Rigid and Soft Body Simulation Featuring Realistic Walk Behaviour. In: RoboCup 2011: Robot Soccer World Cup XV Preproceedings, RoboCup Federation (2011)

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

[11] Czarnetzki, S., Kerner, S., Urbann, O.: Applying dynamic walking control for biped robots. In: RoboCup 2009: Robot Soccer World Cup XIII. Springer Berlin Heidelberg (2010) 69–80

[12] Urbann, O., Tasse, S.: Observer based biped walking control, a sensor fusion approach. Autonomous Robots **35**(1) (2013) 37–49

[13] Urbann, O., Hofmann, M.: Modification of foot placement for balancing using a preview controller based humanoid walking algorithm. In: Proceedings RoboCup 2014 International Symposium, Eindhoven, Netherlands (2014)