In Proceedings of RoboCup International Symposium, Singapore, June 2010.

Learning Footstep Prediction from Motion Capture

Andreas Schmitz, Marcell Missura, and Sven Behnke

University of Bonn,
Computer Science VI, Autonomous Intelligent Systems
Roemerstr. 164, 53117 Bonn, Germany
{schmitz,missura,behnke}@ais.uni-bonn.de
http://ais.uni-bonn.de

Abstract. Central pattern generated walking for bipedal robots has proven to be a versatile and easily implementable solution that is used by several robot soccer teams in the RoboCup Humanoid Soccer League with great success. However, the forward character of generating motor commands from an abstract, rhythmical pattern does not inherently provide the means for controlling the precise location of footsteps. For implementing a footstep planning gait control, we developed a step prediction model that estimates the location of the next footstep in Cartesian coordinates given the same inputs that control the central pattern generator. We used motion capture data recorded from walking robots to estimate the parameters of the prediction model and to verify the accuracy of the predicted footstep locations. We achieved a precision with a mean error of 1.3 cm.

Key words: footstep planning, dynamic walking, machine learning, motion capture

1 Introduction

Central pattern generator based methods and inverse kinematics based methods are two successful approaches to implement controlled dynamic walking for bipedal robots, even though they differ in their core aspects. Central pattern generator based methods [1, 2] generate an abstract, periodic signal stream which is translated to motor commands resulting in rhythmical weight shifting and leg swinging motions. Inverse kinematics based solutions [3, 4, 7] precalculate trajectories in Cartesian coordinates for significant body segments, such as the pelvis and the feet. These trajectories are converted to motor commands by solving the inverse kinematics with the given trajectories as constraints. In the latter case the footstep locations are known in advance: they are determined by the intersections of the foot trajectories with the ground. In the former case, however, footstep locations are not inherently obtainable, since they are indirect results of amplitudes and frequencies of abstract signal patterns.

Our goal is to predict the footstep locations of a central pattern generated walk and to use the predictions to implement a more precise, footstep planning gait control. We present two different approaches to estimate the footstep locations and compare their performance in experimental results.

The remainder of the paper is organized as follows. After reviewing related work, a brief introduction to our gait engine is given in Section 3. Then an overview of the footstep prediction algorithm is presented in Section 4, leading to the detailed descriptions of the two different approaches we implemented: a forward kinematic based approach in Section 5 and a motion capture based approach in Section 6.

2 Related Work

Footstep planning is a fairly new research topic and feasible solutions are scarce in comparison. The most prominent proposals in [8–10] and also [11] are based on the A* algorithm. By imposing a strong discretization on the state space and using only a small, discrete set of actions, these online solutions plan a few steps ahead and are able to deal with dynamic environments. Uneven floor plans are also considered, so that the footstep plans can include stepping over obstacles and climbing stairs. An intruiging alternative solution has been recently shown in [12]. Here, a short sequence of future footsteps is considered to be a virtual kinematic chain as an extension of the robot. Their location is determined using inverse kinematics. The configuration space and the action space are not discretized, but the algorithm is computationally expensive. A computationally more promising method that can plan in a few milliseconds, if the environment is not too cluttered, has been suggested in [13]. The idea is to solve the footstep planning problem mostly with a path planning algorithm. Actual footstep locations are only given in key points, where the walking speed of the robot has to be zero, for example when stepping over an obstacle. The majority of the footstep locations are layed out along the planned paths by the motion generator developed for HRP-2 [3, 6, 5]. The closest related work is [10], where an A* based footstep planning algorithm is adapted for the humanoid robot ASIMO. As the walking algorithm of ASIMO is not precisely known, the authors were forced to reverse engineer a footstep prediction algorithm from observations with a motion capture system.

3 Central Pattern Based Dynamic Walking

In this chapter we introduce the basic concepts of our central pattern based gait generation method in a simplified level of detail. We concentrate on the core modules that are important to understand the footstep prediction model.

3.1 Leg Interface

The leg interface is a low level abstraction layer that allows intuitive control of a leg with three parameters. The leg angle Θ_{Leg} defines the angle of the leg with respect to the trunk, the foot angle Θ_{Foot} defines the inclination of the foot with respect to the transversal plane, and the leg extension η defines the distance between the foot and the trunk (Figure 1). The output of the leg interface are joint angles for the hip, knee and ankle. The leg interface allows independent control of the three parameters and encapsulates the calculation of coordinated joint angles.

$$L(\Theta_{Leg}, \Theta_{Foot}, \eta) = (\Theta_{Hip}, \Theta_{Knee}, \Theta_{Ankle}) \tag{1}$$

The leg can be bent in roll, pitch, and yaw directions as illustrated in Figure 1 (right). The three directional components can be controlled independently from each other. Most importantly, the foot is rotated around its own center and not around the trunk. More detailed information can be found in [1].

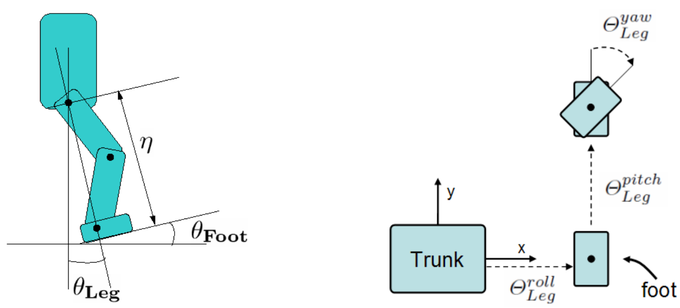

Fig. 1. The leg interface allows independent control of three abstract parameters: the leg angle Θ_{Leg}, the foot angle Θ_{Foot}, and the leg extension η (left). The leg can be bent independently in roll, pitch, and yaw directions (right).

3.2 Central Pattern Generator (CPG)

The CPG generates patterns of rhythmic activity governed by a periodic internal clock called gait phase $-\pi \leq \phi < \pi$. The patterns encode the waveforms of the leg interface parameters. In particular, the leg extension is activated with a sinusoidal function, whereas the phase of the left leg is shifted by π with respect to the right leg (Figure 2, left).

$$P_w = sin(\phi) \tag{2}$$

The antidromic shortening and extending of the legs causes a rhythmic lateral shift of the body weight alternatingly freeing a leg from its support duty. This

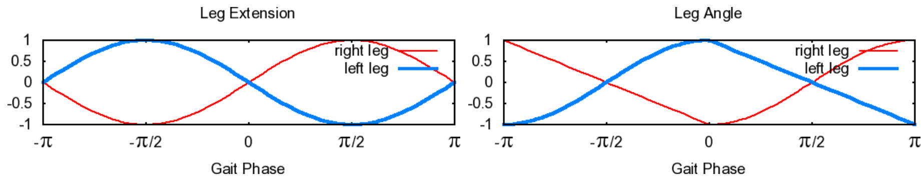

Fig. 2. The CPG waveforms encode the leg extension (left) and leg swing (right) patterns.

leg can be swung. In concert with the leg extension signal P_w, the CPG generates a second activation to swing the free leg.

$$P_s = sin(\phi - \frac{\pi}{2}), \quad -\pi \leq \phi < 0 \tag{3}$$

$$P_s = 1 - \frac{\phi}{\pi}, \quad 0 \leq \phi < \pi \tag{4}$$

As shown in Figure 2 right, the leg is swung forward with a sinusoidal motion and pushed back with a linear motion in the support phase. Support exchange is expected to occur at gait phase $\phi = 0$ from right to left and at gait phase $\phi = -\pi$ from left to right.

3.3 Omnidirectional Gait Control

Walking direction and step size are controlled by modulating the amplitude of the leg swing activation P_s with a gait control vector $g \in [-1, 1]^3$ and applying it to the roll, pitch and yaw component of the leg angle Θ_{Leg}. Omnidirectional walking is achieved by applying the swing signal in all three directions simultaneously with different intensities. For example, a mixture of the pitch and yaw components will result in a curved walk forward, where the yaw intensity determines the curvature of the path. The modulated signals are then transformed by a configuration vector $c \in \mathbb{R}^3$, which is a mapping from CPG signal space to leg angle space expressed in radians. c can be used to adapt the very same CPG patterns to robots of different sizes, for example from the KidSize and the TeenSize class and to fine tune individual robots. In summary, these equations describe the generation of the gait trajectory:

$$\Theta_{Leg}^{roll} = P_s \cdot g_x \cdot c_x + |g_x| \cdot c_x \tag{5}$$

$$\Theta_{Leg}^{pitch} = P_s \cdot g_y \cdot c_y \tag{6}$$

$$\Theta_{Leg}^{yaw} = P_s \cdot g_z \cdot c_z + |g_z| \cdot c_z \tag{7}$$

At the end of the gait control chain the leg interface converts the leg angles and extensions to joint angles, as depicted in Figure 3.

Notably, the equation in sagittal direction (6) differs from the lateral and yaw directions (5,7). In sagittal direction the legs are encouraged to swing fully from

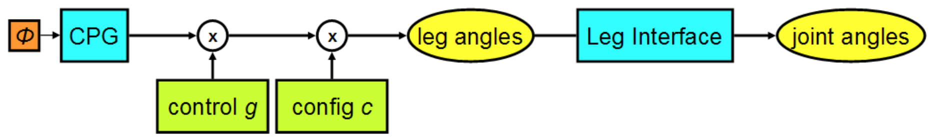

Fig. 3. The gait trajectory generation chain. The CPG is always active and produces periodic activation signals clocked by the gait phase ϕ. After modulation by the gait control g, the signals are mapped to leg angle space with the configuration c. The leg interface converts the leg angles to joint angles that are passed to the robot.

front to back. Positive and negative leg angles are likewise allowed. In lateral direction, however, the legs would collide. To avoid negative leg angles a positive value is added to the leg roll angle proportionally to the lateral component of the gait control vector causing the legs to spread out when walking in lateral direction. As a result, two different step sizes occur: a long step, when the leading leg is swung in the direction the robot is moving and a short step, when the other leg is pulled in to meet the leading leg at a leg angle close to zero. The same is true for the yaw direction.

4 The Step Prediction Models

For the implementation of a step planning algorithm, a forward model

$$F(g_x, g_y, g_z, \phi) = (p_x, p_y, p_\theta) \tag{8}$$

is required that maps a gait control vector g to an expected footstep location and orientation $p \in \mathbb{R} \times \mathbb{R} \times [-\pi, \pi]$ in Cartesian coordinates. We define p to be the location and orientation of the footstep in the local coordinate frame of the support foot. As outlined in the previous Section, a gait control vector g can produce two different step sizes. The gait phase ϕ has to be included as parameter for reasons of disambiguation. The value of ϕ is determined by keeping track of the support leg and knowing at which gait phase the next step will occur.

We developed two strategies to obtain this mapping. As an analytic approach we used a kinematic model of the robot to calculate the forward kinematics from given joint angles. Alternatively, we collected training data from a motion capture device and used linear regression to learn the mapping F. Both approaches are described in detail in the following sections.

5 Kinematic Model Approach

The kinematic model requires a precise skeleton of the robot that we acquired from the CAD construction blueprints. Predictions are made by applying the joint angles in the moment of a step to the kinematic model and calculating the position and orientation of the swing foot relative to the support foot in Cartesian space (Figure 5). To get hold of the joint angles in the moment of a step for a specific gait control vector g, we set the appropriate gait phase ϕ (0 or $-\pi$) and execute the gait trajectory generation chain (Figure 3). The CPG will output the signals that it would produce in the moment of a step. Using the configuration c of a specific robot and the gait control vector g, we acquire the desired joint angles.

Due to their analytic nature, the predictions can be calculated very efficiently. However, some sources of error are inevitable. Inaccuracies in the skeleton cannot be completely avoided as well as the fact that the robot does not perfectly obey the commanded joint angles. There is always mechanical wear, backlash in the gears and undesired elasticities that cause the physical system to deviate from theory. We decided to take an alternative approach and learn from data collected from the actual physical system. This approach is presented in the next Section.

6 Machine Learning Approach

As an alternative course, we collected ground truth data with a motion capture system to learn from the footsteps as they really happen. Two KidSize robots were equipped with reflective markers in groups of three or four on the head, the hip, and the feet, as shown in Figure 4. With both robots we recorded approximately five minutes of more or less random walking speeds and directions, trying to explore the entire gait control space. The output of the motion capture device are trajectories of the reflective markers, which we synchronized with a recording of the gait control vector. In data post processing, we calculated the centroids of every marker group to represent the head, the hip and the feet with only one point. The centroids were used for further post processing. From the hip marker group we calculated the orientation of the robot with respect to the global vertical axis. The orientation describes in which direction the robot is facing in the world coordinate frame, but this is not necessarily equal to the walking direction. Using the feet marker groups we also computed the orientation of both feet relative to the global orientation of the robot.

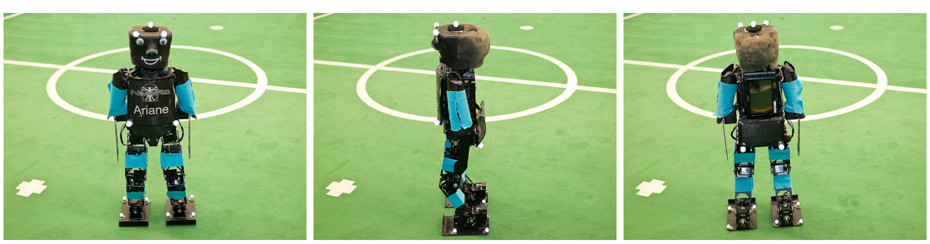

Fig. 4. Groups of reflective markers were used to indicate the head, the hip and each of the feet.

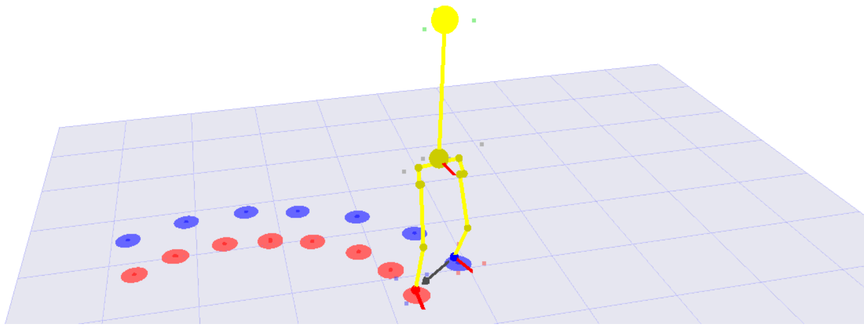

Fig. 5. Visualization of the data obtained from the motion capture device: the marker cloud, the kinematic model fitted into the cloud and performing a step, the orientation of the robot and the feet, and some of the extracted footstep locations on the floor. The arrow between the feet illustrates the vector we extract from the kinematic model.

To produce the training data, single steps had to be identified. We used the feet centroids to extract two features: the height of the feet h_l and h_r and the velocities of the feet v_l and v_r calculated from two consecutive frames. A step is recognized when the feet have approximately the same height and the same velocity:

$$|h_l - h_r| + |v_l - v_r| < 0.01\,. \tag{9}$$

Altogether we identified approximately 3000 footsteps and matching gait control vectors. Figure 5 shows a visualization of the marker cloud, the kinematic model fitted into the cloud, the orientation of the trunk and the feet, and some of the extracted footstep locations on the floor.

As mentioned in Section 3.1, the leg interface implements independent control of the footstep location and orientation. This allows the assumption that instead of the three dimensional function F (8), three independent one dimensional functions can be learned. However, when performing a step, the fixed frame of reference is the support foot and not the coordinate frame of the trunk. The g_z component of the gait control vector applies a rotation to the footstep location by an angle α and consequently p_x and p_y both depend on g_z. This is shown in Figure 6 (a). To tackle this problem, we introduce a footstep $q = (q_x, q_y, \alpha)$ in the reference frame of the trunk (Figure 6 (b)). q_x and q_y are defined as the distances between the feet in x and y directions and α is the orientation of the swing foot with respect to the trunk. In this frame of reference q_x depends only on g_x, q_y only on g_y, and α only on g_z. These are the mappings that we learn. We salvage (q_x, q_y, α) from the identified steps in the motion capture data. Knowing the reference frame of the trunk from the hip marker group, we calculate (q_x, q_y) from the difference between the foot coordinates of the fitted skeleton. α is equal to the swing foot orientation that we calculated from the foot marker group. We found the support foot angle and the swing foot angle to be symmetrical enough,

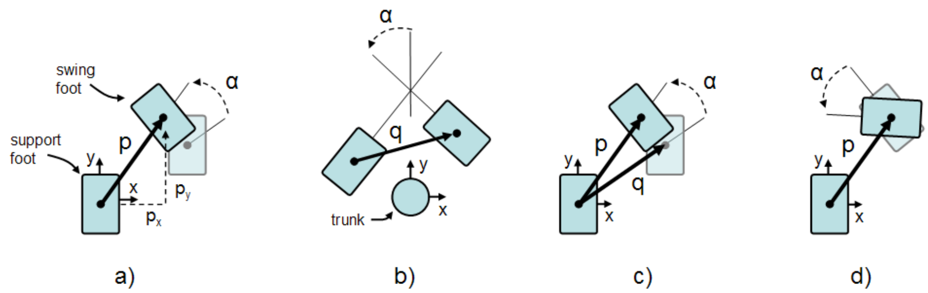

Fig. 6. The g_z component of the gait control vector rotates the footstep location by the angle α around the support foot and influences p_x and p_y (a). The footstep q in the trunk frame, however, is independent from the rotation (b). q is mapped to the support foot related footstep p with a rotation by α (c) and an additional rotation of the swing foot again by the angle α (d).

so that they do not have to be modeled separately. The trunk oriented footstep q is then transformed to the support foot oriented footstep p with a step transform function T as depicted in Figure 6 (c) and (d). (q_x, q_y) is rotated in the frame of the support foot by the angle α. Then, the swing foot is rotated around it's own origin again by the angle α. The transform function T is given by:

$$(p_x, p_y) = (q_x, q_y) \cdot R(\alpha) \tag{10}$$

$$p_\theta = 2\alpha \tag{11}$$

where R denotes a rotation matrix.

Moreover, we implemented another simplification of the learning task. Figure 7 shows a decomposition of function F. Given g and ϕ as input, the gait control chain (Figure 3) can be used to calculate the leg angles at the moment of the next step. The step function S is the actual physical step. It maps the leg angles to a footstep q in the reference frame of the trunk. The step transform T translates q to the footstep p in the reference frame of the support foot. Only S needs to be learned and can be approximated by three simple, independent, one dimensional functions:

$$q_x = q_x(\Theta_{Leg}^{roll}), \tag{12}$$

$$q_y = q_y(\Theta_{Leg}^{pitch}), \tag{13}$$

$$\alpha = \alpha(\Theta_{Leg}^{yaw}). \tag{14}$$

Using the gait chain as an executable building block also allows us to incorporate different robot configurations just by exchanging them in the gait control chain. We expect better adaptability of the algorithm to different individuals of the same robot type.

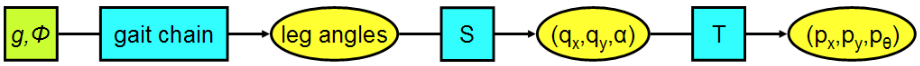

Fig. 7. A decomposition of function F. Given g and ϕ, the gait control chain calculates the leg angles at the moment of the next step. The physical step S maps the leg angles to the footstep q in the reference frame of the robot. The step transform T translates q to the footstep p in the reference frame of the support foot.

7 Experimental Results

Figure 8 presents data from Mocap and the approximated functions for q_x, q_y and α (eqs. 12-14). All three relationships show a strong linear character and can easily be fitted in the most simple manner with linear functions. Naturally, one would expect the functions of leg angles to step sizes to be sinusoids. The most likely explanation for the seemingly linear coherence is that our robots are taking relatively small steps and sinusoid functions can be well approximated with linear ones, as long as their argument stays close to zero. The function mapping Θ_{Leg}^{yaw} to α is almost identity. The small deviation from identity must originate from the error the physical system makes when at the commanded leg angle. With this deviation accounted for we can expect an improvement in our footstep predictions.

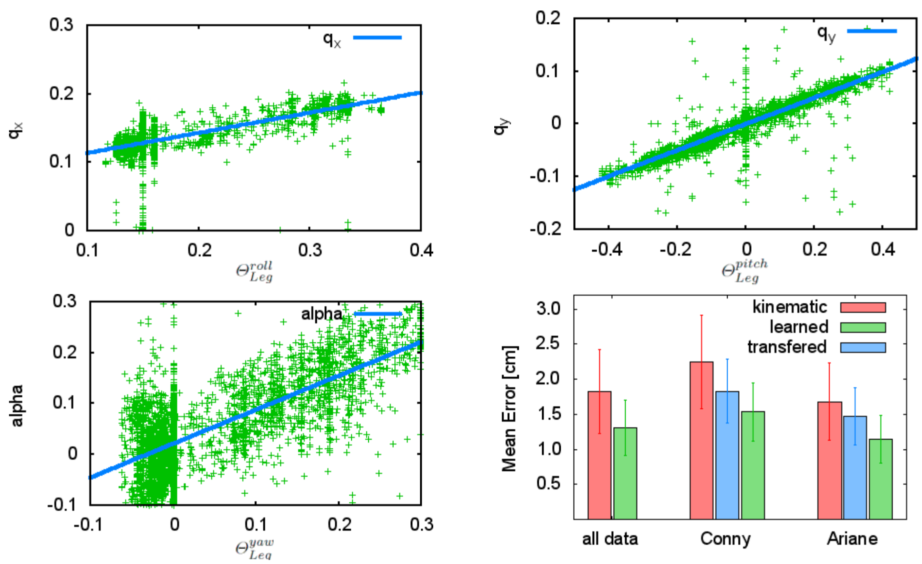

Fig. 8. Plots of the collected data and the approximated functions for q_x (upper left), q_y (upper right), and α (lower left). All three relationships show a linear character. The comparison of the prediction accuracies of the analytic model and the machine learning model on the full step set and the robot specific experiments are shown in the lower right.

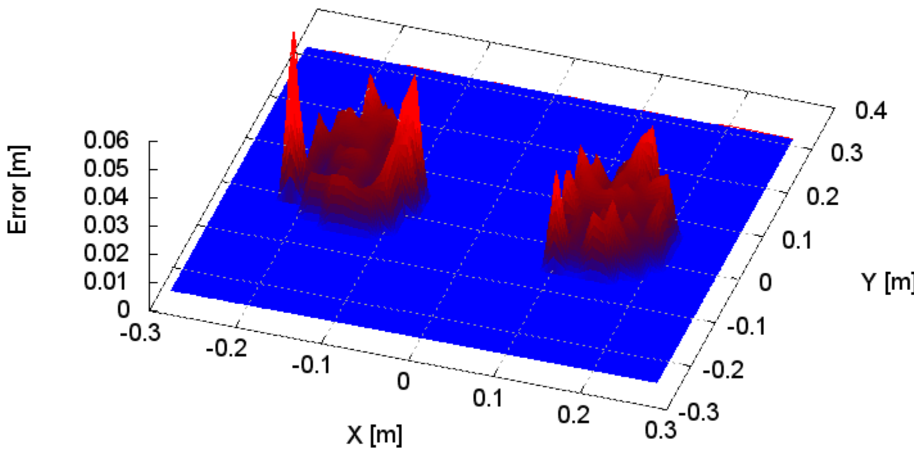

Fig. 9. The working spaces of the left foot and right foot. The height of the graph shows the measured error at each footstep location.

We used the motion capture data to compare the performance of the forward kinematic approach and the machine learning approach. The error of a single prediction is measured by the Euclidean distance between the predicted footstep and the ground truth footstep taken from the motion capture data. We present figures of the mean error and standard deviation measured in three different experiments in Figure 8 lower right. In the first experiment we compared the performance of the forward kinematic model and the machine learning approach on the complete set of approximately 3000 footsteps. The performance of the machine learning approach was measured by 4-fold cross validation. In the second and third experiment we compared the two methods exclusively on the step set produced by only one of the robots, e.g. Conny and Ariane respectively. Additionally, we evaluated transfered models that were trained with 500 steps randomly sampled exclusively from the step set of the other robot. All trained models outperformed the forward kinematic approach. Since the forward kinematic is the same for both robots, the difference in accuracy between the Conny experiment and the Ariane experiment can be explained by a more instable walking of Conny, which is more difficult to predict due to occasional stumbling. Even though the best results were achieved on Ariane (1.1 cm), we believe this model to be overly adapted to this robot and use the mode trained on the full step set as reference, which achieved an accuracy with a mean error of approximately 1.3 cm.

Figure 9 shows a representation of the working spaces of the feet. Every step was plotted by assuming the support foot location to be at the origin. The height describes the measured error. In the base stance of the robot, even if the gait control vector g is 0, the feet are 12 cm apart. These "zero" steps of the robot are roughly in the respective centers of the two areas. A not unexpected tendency of the prediction error growing with the deviation from the zero step can be observed. The diameters of the working spaces are approximately 20 cm and the avarage step size was determined to be 14.4 cm. Comparing the mean error of 1.3 cm to the avarage step size leads to the conclusion, that our predictions are in avarage 90.97% accurate. Future work will show if this accuracy is enough for a step planning algorithm to hit a small target such as a tennis ball.

Investigating further possibilities to improve the prediction accuracy, we considered discarding the independency assumption and using a non-linear regression method to learn the function

$$f(\Theta_{Leg}^{roll}, \Theta_{Leg}^{pitch}, \Theta_{Leg}^{yaw}) = (q_x, q_y, \alpha) \tag{15}$$

in one sweep. For this attempt to be fruitful, some dependency has to exist between the three parameters and each of the output dimensions. We examined the correlation between the leg angles produced by the gait and the deviation Δq, which is the difference between the robot centered footstep q and the expected footstep predicted by our linear estimators. Table 1 contains the correlation coefficients. Apart from a small influence of Θ_{Leg}^{yaw} on q_x, no correlation between the parameters and the footstep deviations can be identified. Consequently, our linear approach has already exhausted the learning problem. A more complex approach cannot be expected to produce significantly better results on the same data.

8 Conclusions

We presented a footstep prediction algorithm for a central pattern generated omnidirectional walk. We were able to decompose the footstep prediction into three independent functions that could be approximated with linear regression performed on motion capture data. We achieved a prediciton accuracy with a mean error of 1.3 cm. The learned model outperformed the analytic, forward kinematic based solution and showed some transferability to different individuals of the same robot model. We claim that our linear approach cannot be significantly improved with a more complex technique. In future work we plan

	Δq_x	Δq_y	$\Delta\alpha$
Θ_{Leg}^{roll}	0.022	0.043	0.037
Θ_{Leg}^{pitch}	0.070	0.028	0.020
Θ_{Leg}^{yaw}	0.241	0.013	0.001

Table 1. Correlation coefficients between the leg angles in roll, pitch and yaw directions and the deviation of the footstep q in the reference frame of the trunk.

to use the footstep predictions for a step planning gait control and test the algorithms in real robot soccer games.

Acknowledgments

Funding for the project is provided by Deutsche Forschungsgemeinschaft (German Research Foundation, DFG) under grants BE 2556/2-2,/4.

References

1. S. Behnke, *Online Trajectory Generation For Omnidirectional Biped Walking*, In ICRA, 2006
2. M. Zhao, J. Zhang, H. Dong, Y. Liu, L. Li and X. Su, *Humanoid Robot Gait Generation Based on Limit Cycle Stability*, In RoboCup 2008: Robot Soccer World Cup XII, 2008, pp. 403-413, Springer
3. 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*, 2003, In ICRA, 2003, pp. 1620-1626
4. Q. Huang, K. Yokoi, S. Kajita, K. Kaneko, H. Arai, N. Koyachi and K. Tanie, *Planning Walking Patterns for a Biped Robot*, In IEEE Transactions on Robotics and Automation, 2001, Vol. 17, pp. 280-289
5. K. Kaneko and F. Kanehiro and S. Kajita and H. Hirukawa and T. Kawasaki and M. Hirata and K. Akachi and T. Isozumi, *Humanoid Robot HRP-2*, In ICRA, 2004, pp.1083-1090
6. S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi and H. Hirukawa, *The 3D linear inverted pendulum mode: a simple modeling for a bipedwalking pattern generation*, In IROS, 2001, Vol. 1, pp. 239-246
7. M. Friedmann, J. Kiener, S. Petters, H. Sakamoto, D. Thomas and O. von Stryk, *Versatile, high-quality motions and behavior control of humanoid soccer robots*, In Workshop on Humanoid Soccer Robots of the 2006 Humanoids, 2006, pp. 9-16
8. J. Chestnutt and J. Kuffner, *A Tiered Planning Strategy for Biped Navigation*, In Humanoids, 2004
9. J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba and H. Inoue, *Online Footstep Planning for Humanoid Robots*, In ICRA, 2003, pp. 932-937
10. J. Chestnutt, M. Lau, K.M. Cheung, J. Kuffner, J.K. Hodgins and T. Kanade, *Footstep Planning for the Honda ASIMO Humanoid*, In ICRA, 2005
11. J.S. Gutmann, M. Fukuchi and M. Fujita, *Real-time path planning for humanoid robot navigation*, In Proc. of 19th Int. Conf. on Artificial intelligence, 2005, pp. 1232-1237
12. O. Kanoun, E. Yoshida and J.P. Laumond, *An Optimization Formulation for Footsteps Planning*, In Humanoids, 2009
13. Y. Ayaz, T. Owa, T. Tsujita, A. Konno, K. Munawar and M. Uchiyama, *Footstep Planning for Humanoid Robots Among Obstacles of Various Types*, In Humanoids, 2009