In Proceedings of the 2006 IEEE International Conference on Robotics and Automation (ICRA), Orlando, Florida, pp. 1597-1603, May 2006.

Online Trajectory Generation for Omnidirectional Biped Walking

Sven Behnke
University of Freiburg, Germany, Computer Science Institute
behnke@informatik.uni-freiburg.de

***Abstract*— This paper describes the online generation of trajectories for omnidirectional walking on two legs. The gait can be parameterized using walking direction, walking speed, and rotational speed. Our approach has a low computational complexity and can be implemented on small onboard computers.**

We tested the proposed approach using our humanoid robot Jupp. The competitions in the RoboCup soccer domain showed that omnidirectional walking has advantages when acting in dynamic environments.

I. Introduction

Omnidirectional locomotion is a concept that has proven to be advantageous in dynamic environments and in restricted spaces. Wheeled omnidirectional vehicles, for instance, provide superior maneuvering capability compared to the more common car-like vehicles. The ability to move into any direction, irrespective of the orientation of the vehicle, and to control the rotational speed at the same time has advantages in many domains. One such domain is robotic soccer. Since omnidirectional drives have been introduced in 2000 by the team Big Read (Cornell) [1] in the RoboCup SmallSize league [2], most teams in the wheeled leagues adopted this approach. It is much easier to position robots for kicking and to outmaneuver opponents when using an omnidirectional drive.

Omnidirectional locomotion is not restricted to wheeled vehicles, though. Spenneberg and Kirchner, for example, implemented omnidirectional walking in a biomimetric robot with eight legs [3]. Porta and Celaya [4] describe an approach to body and leg coordination of a hexapod robot for omnidirectional walking in rough terrain. Omnidirectional walking is also heavily used in the RoboCup Four-legged League. As one of the first, Hengst et al. [5] describe an approach to generate omnidirectional walking for the Sony ERS-110 Aibo dogs.

The above examples motivated us to implement omnidirectional walking for bipedal robots as well. Our robots compete in the RoboCup Humanoid League, a new league added in 2002. Here, bipedal locomotion, balance, and ball manipulation are some of the main research issues.

This paper describes an approach for online trajectory generation that produces fully parameterizable omnidirectional walking for biped robots. Our approach has a low computational complexity. We implemented it on the Pocket PC of our humanoid robot Jupp and tested it in the RoboCup soccer domain. Jupp's design is based on its predecessor Toni [6]. Compared to Toni, Jupp has an additional rotational joint in each thigh, an additional joint in the trunk, extended joint limits, and stronger arms. These new features simplified rotation of the robot on the spot while walking and allowed us to implement getting-up behaviors. We also added a compass module in order to make self-localization easier.

In the next section, we review some of the related work. Section III describes Jupp's mechanical and Section IV its electrical design in detail. Section V describes how Jupp is controlled to achieve omnidirectional walking. Section VI summarizes experimental results obtained with this gait.

II. Related Work

During the past 30 years, many different schemes to control walking bipeds have been implemented, with varying degree of success. Among the most common approaches are trajectory tracking methods [7], [8], [9]. Trajectories for individual joints or for the zero moment point (ZMP) are generated offline, e.g. by solving the dynamic equations of motion. High gain position controllers are used during walking to follow the predefined trajectory.

The ZMP is defined as the point on the ground about which the sum of the moments of all the active forces equals zero. If the ZMP is within the convex hull (support polygon) of all contact points between the feet and the ground, a bipedal robot is dynamically stable [10]. The use of the ZMP to judge stability was a major advance over the center-of-mass projection criterion, which describes static stability.

Trajectory tracking requires precise knowledge of dynamic parameters of the robot to prepare walking. It also relies on a good environment model. Trajectory controlled robots are not robust to disturbances or changes in the environment. Furthermore, trajectory tracking is computationally intensive. The dynamic equations for a robot with many degrees-of-freedom (DOF) can be very difficult to solve, even numerically.

To make online feedback possible, simplified models, such as the inverted pendulum method [11] are used. When a biped robot is supporting its body on one leg its dominant dynamics can be represented by its center of mass, which is connected by a massless telescopic leg to the supporting foot. Other feedback strategies rely on virtual model control [12]. Virtual springs and dampers are placed at strategic locations to control the robot's pitch, height, and speed.

Another approach to incorporate feedback into walking is to entrain so called central pattern generators (CPGs) with the robot-environment dynamics. Neurophysiological studies showed that such neural oscillators drive the walking rhythm

in vertebrates [13]. They also participate in the control of human walking [14]. No dynamic model of the robot or the environment is needed for this method. It is, though, not straightforward to determine the appropriate parameter settings for the oscillator networks to generate a suitable pattern for walking control.

All the above approaches are used to control a given robot. Frequently, constraints, such as a constant height of the robot's center-of-mass (COM) are imposed on the robot motion when generating trajectories. Due to limited knee speed and singularities, this leads to unnatural walking styles, like the walking with bent knees. The problems, like high torques and high energy consumption, associated with bent knees have only recently been addressed by lowering the COM in the double-support phase, making a gait with mostly stretched knees possible [15]. A larger variety walking styles has been realized for planar bipeds [16].

A completely different approach to walking is to utilize the robot dynamics. McGeer showed that planar walking down a slope is possible without actuators and control [17]. Based on his ideas of passive dynamic walking, actuated machines have been built recently [18]. They are able to walk on level ground. Because their actuators only support the inherent machine dynamics, they are very energy-efficient. They are easy to control, e.g. by relying on foot-contact sensors. It has also been shown that similar bipeds can walk in a stable manner by relying on feed-forward control only [19]. Other recent robots also demonstrate that a suitable robot morphology can simplify control [20]. Examples for robots utilizing springs to store energy and to generate dynamics are the robots constructed by Iida and Pfeifer [21] and by Tilden [22].

To our knowledge, omnidirectional walking has not been implemented for bipedal robots. Even the most advanced humanoids, like Sony Qrio [23] and Honda Asimo [24], do not possess the ability to fully parameterize their walking motion.

The gait of most bipedal robots is controlled by precomputed trajectories that describe ZMP-stable motion primitives, such as walking straight in forward direction, turning on the spot, and making a step to the side. These macros are usually chained to create more complex trajectories. However, because macros frequently cannot be combined, the robots must make brief stops when changing their walking direction.

In the new Asimo research model, walking patterns are created online. The robot can parameterize foot placement and turning angle according to the current situation. As a result, it can walk smoothly in many, but not all, directions. It can also change its step frequency.

The lack of fully parameterizable biped gaits might be due to the missing pressure from the application side. Most bipeds walk only in research labs and do not have to adapt their walking direction, speed, and rotation to changes in a dynamic environment.

In addition to the expensive larger humanoid robots, which are usually driven by DC-motors and harmonic drive gears, some smaller servo-driven humanoid robots have been developed recently [25], [26], [27]. Servo motors are used for humanoid robots because of their low cost and because of their good weight-to-torque ratio. The servo-driven robots have up to 22DOFs and a size of 30-40cm. We follow this approach in the design of our humanoid robot Jupp, but increase robot size without adding much extra weight.

III. Mechanical Design

Fig. 1 shows our humanoid robot Jupp. It has been designed for the 2005 RoboCup Humanoid League competitions in the KidSize class. As can be seen, Jupp has human-like proportions. Its mechanical design focused on weight reduction. Jupp is 60cm tall and has a total weight of only 2.3kg.

The robot is driven by 19 servo motors: 6 per leg, 3 in each arm, and one in the trunk. The six leg-servos allow for flexible leg movements. Three orthogonal servos constitute the 3DOF hip joint. Two orthogonal servos form the 2DOF ankle joint. One servo drives the knee joint.

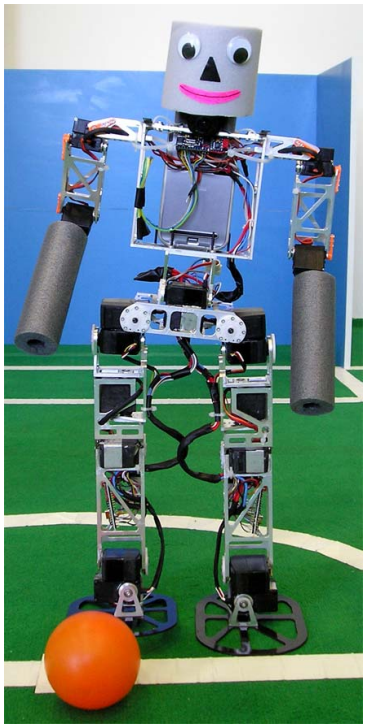


Fig. 1. Frontal view of the humanoid robot Jupp.

We selected the S9152 servos from Futaba to drive the roll and yaw joints of the hips, the knees, and the ankles. These digital servos are rated for a torque of 200Ncm and have a weight of only 85g. The hip yaw joints need less torque. They are powered by DS 8811 servos (190Ncm, 66g). We augmented all servos by adding a ball bearing on their back, opposite to the driven axis. This made a stiff hinge joint construction possible. Jupp's arms do not need to be as strong as the legs. They are powered by SES640 servos (64Ncm, 28g). Two orthogonal servos constitute the shoulder joint and one servo drives the elbow joint.

The robot's skeleton is constructed from aluminum extrusions with rectangular tube cross section. In order to reduce

weight, we removed all material not necessary for stability. Jupp's feet and its forearms are made from sheets of carbon composite material. The elasticity of the feet and the carpet, the robot walks on, helps to maintain non-degenerate foot-ground contact, even when the supporting foot is not parallel to the ground. Jupp's head is made of lightweight foam.

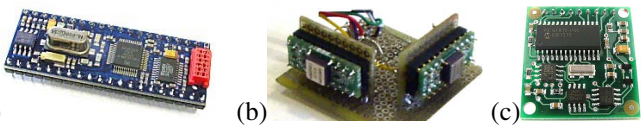
IV. Electronics

Jupp is fully autonomous. It is powered by high-current Lithium-polymer rechargeable batteries, which are located in its lower back. Two Kokam 2000H cells last for about 30 minutes of operation. They can be discharged with 30A and have a weight of only 110g.

The servos are interfaced to three tiny ChipS12 microcontroller boards, shown in Fig. 2(a). One of these boards is located in each shank and one board is hidden in the chest. These boards feature the Motorola MC9S12C32 chip, a 16-bit controller belonging to the popular HCS12 family. We clock it with 24MHz. It has 2kB RAM, 32kB flash, a RS232 serial interface, CAN bus, 8 timers, 5 PWM channels, and 8 A/D converters. We use the timer module to generate pulses of 1...2ms duration at a rate of 180Hz in hardware. These pulses encode the target positions for the servos. Up to eight servos can be controlled with one board. In order to keep track of the actual servo movements, we interfaced their potentiometers to the A/D converters of the HCS12. By analyzing the temporal fine structure of these signals, we estimate not only the current servo positions, but also the PWM duty cycles of their motors.

In addition to these joint sensors, Jupp is equipped with an attitude sensor and a compass. The attitude sensor, shown in Fig. 2(b), is located in the trunk. It consists of a dual-axis accelerometer (ADXL203, ± 1.5g) and two gyroscopes (ADXRS 150/300, $\pm 150/300$ deg/s). The four analog sensor signals are digitized with A/D converters of the HCS12 and are preprocessed by the microcontroller. The compass module, shown in Fig. 2(c), is located in the head of the robot. It is interfaced to the timer module of the HCS12. Using pulse-width modulation, it indicates the robot's heading direction, relative to the earth's magnetic field.

The microcontrollers communicate with each other via a CAN bus at 1MBaud and with a main computer via a RS232 serial line at 115KBaud. Every 12ms, target positions for the servos are sent from the main computer to the HCS12 boards, which generate intermediate targets at 180Hz. This yields smooth joint movements. It is also possible to relax the digital servos. The microcontrollers send the preprocessed sensor readings back. This allows keeping track of the robot's state in the main computer. We use a Pocket PC as main computer [28], which is located in Jupp's chest (see Fig. 1). The FSC Pocket Loox 720 has a weight of only 170g, including the battery. It features a 520MHz XScale processor PXA-272, 128MB RAM, 64MB flash memory, a touch-sensitive display with VGA resolution, Bluetooth, wireless LAN, a RS232 serial interface, and an integrated 1.3 MPixel camera.



(a) (b) (c)

Fig. 2. Electronics: (a) Microcontroller; (b) Attitude sensor; (c) Compass.

This computer runs behavior control, computer vision, and wireless communication. It is equipped with a Lifeview FlyCAM CF 1.3M that has been fitted to an ultra-wide-angle lens. The lens is located at the position of the larynx. The wide field of view of this camera (about $112° \times 150°$) allows Jupp to see at the same time its own feet and objects above the horizon.

V. Behavior Control

We control Jupp using a framework that supports a hierarchy of reactive behaviors [29]. This framework allows for structured behavior engineering. Multiple layers that run on different time scales contain behaviors of different complexity. This framework forces the behavior engineers to define abstract sensors that are aggregated from faster, more basic sensors. One example for such an abstract sensor is the robot attitude that is based on the readings of accelerometers and gyros. Abstract actuators give higher-level behaviors the possibility to configure lower layers in order to eventually influence the state of the world. One such abstract actuator would be the desired walking direction, which configures the gait engine, described below, implemented in the lower control layers.

The framework also supports an agent hierarchy. For Jupp, we use three levels of this hierarchy: individual joint, body part, and entire robot. This structure restricts interactions between the system variables and thus reduces the complexity of behavior engineering. The lowest level of this hierarchy, the control loop within the servo, has been implemented by the servo manufacturer. It runs at about 300Hz for the digital servos. We monitor targets, actual positions, and motor duties.

At the next layer, we generate target positions for the individual joints of a body part at a rate of 83.3Hz. We make sure that the joint angles vary smoothly. This layer implements an interface that describes the behavior of body parts. As an example, we detail the interface of a leg below.

A. Leg Interface

The entire leg can be positioned relative to the trunk using leg extension (the distance from the hip joint to the ankle joint), leg angle (angle between the pelvis plate and the line from hip to ankle), and foot angle (angle between foot plate and pelvis plate).

Let $\theta_{\mathbf{Leg}} = (\theta^{\mathrm{r}}_{\mathrm{Leg}}, \theta^{\mathrm{p}}_{\mathrm{Leg}}, \theta^{\mathrm{y}}_{\mathrm{Leg}})$ denote the desired leg angle with the convention that the leg pitch angle $\theta^{p}_{Leg} = 0$ if the leg is parallel to the trunk and $\theta^{p}_{Leg} > 0$ if the leg is in front of the trunk. Similarly, the leg roll angle $\theta^{\mathrm{r}}_{\mathrm{Leg}} = 0$ if the leg is parallel to the trunk, and $\theta^{\mathrm{r}}_{\mathrm{Leg}} > 0$ if the leg is moved outwards in the lateral plane. The leg yaw angle $\theta^{y}_{Leg} = 0$ if the foot points in forward direction. $\theta^{y}_{Leg} >$ twists the leg, such that the foot points outwards. Furthermore, $\theta_{\mathbf{Foot}} = (\theta^{\mathrm{r}}_{\mathrm{Foot}}, \theta^{\mathrm{p}}_{\mathrm{Foot}})$ denotes the desired foot angle, $\theta_{\mathbf{Foot}} = (0, 0)$ if the foot is

parallel to the pelvis plate. Finally, $-1 \leq \gamma \leq 0$ denotes the desired leg extension, with the convention $\gamma = 0$ if the leg is fully extended and $\gamma = -1$ if the leg is shortened to $\eta_{min} = 0.875$ of its original length. The target relative leg length η can be computed as $\eta = 1 + (1 - \eta_{min})\gamma$. The knee angle $\theta_{\mathrm{Knee}} = -2 \cdot \mathsf{acos}(\eta)$ shortens the leg, but would also change the leg and foot angles. Because the thigh and shank of the robot have the same length, if the leg is not twisted ($\theta^{\mathrm{y}}_{\mathrm{Leg}} = 0$), we can subtract $0.5 \cdot \theta_{\mathrm{Knee}}$ from $\theta^{\mathrm{p}}_{\mathrm{Hip}}$ and from $\theta^{\mathrm{p}}_{\mathrm{Ankle}}$ to compensate this effect. The leg is twisted using the thigh joint: $\theta_{\mathrm{Thigh}} = \theta^{\mathrm{y}}_{\mathrm{Leg}}$. For twisted legs, ($\theta_{\mathrm{Thigh}} \neq 0$), the knee angle must be rotated, before subtracting it from the hip. With the rotation matrix

$$R_{\theta_{\mathrm{Thigh}}} = \begin{pmatrix} \mathsf{cos}(\theta_{\mathrm{Thigh}}) & \mathsf{sin}(\theta_{\mathrm{Thigh}}) \\ -\mathsf{sin}(\theta_{\mathrm{Thigh}}) & \mathsf{cos}(\theta_{\mathrm{Thigh}}) \end{pmatrix}$$

the hip-update becomes

$$\begin{pmatrix} \Delta\theta^{\mathrm{r}}_{\mathrm{Hip}} \\ \Delta\theta^{\mathrm{p}}_{\mathrm{Hip}} \end{pmatrix} = R_{\theta_{\mathrm{Thigh}}} \begin{pmatrix} 0 \\ -0.5 \cdot \theta_{Knee} \end{pmatrix}.$$

The roll and pitch components of the leg angle $(\theta^{r}_{\mathrm{Leg}}, \theta^{p}_{\mathrm{Leg}})$ are added to $\theta_{\mathbf{Hip}}$. For untwisted legs, they must be subtracted from $\theta_{\mathbf{Ankle}}$ to keep the foot angle unchanged. For ($\theta_{\mathrm{Thigh}} \neq 0$), the ankle-update must be rotated against θ_{Thigh}:

$$\boldsymbol{\Delta}\theta_{\mathrm{Ankle}} = -R^{-1}_{\theta_{\mathrm{Thigh}}} \begin{pmatrix} \theta^{r}_{\mathrm{Leg}} \\ \theta^{p}_{\mathrm{Leg}} \end{pmatrix}.$$

Finally, the foot angle $\theta_{\mathbf{Foot}}$ is rotated against θ_{Thigh} and added to $\theta_{\mathbf{Ankle}}$. This yields:

$$\theta_{\mathbf{Hip}} = \begin{pmatrix} \theta^{r}_{\mathrm{Leg}} \\ \theta^{p}_{\mathrm{Leg}} \end{pmatrix} + R_{\theta_{\mathrm{Thigh}}} \begin{pmatrix} 0 \\ -0.5 \cdot \theta_{Knee} \end{pmatrix}, \quad (1)$$

$$\theta_{\mathbf{Ankle}} = \begin{pmatrix} 0 \\ -0.5 \cdot \theta_{Knee} \end{pmatrix} + R^{-1}_{\theta_{\mathrm{Thigh}}} (\theta_{\mathbf{Foot}} - \begin{pmatrix} \theta^{r}_{\mathrm{Leg}} \\ \theta^{p}_{\mathrm{Leg}} \end{pmatrix}). \quad (2)$$

The leg interface represented by $\theta_{\mathbf{Leg}}, \theta_{\mathbf{Foot}}$, and γ is a more abstract actuator space than the space spanned by the individual joint angles $\theta_{\mathbf{Hip}} = (\theta^{\mathrm{r}}_{\mathrm{Hip}}, \theta^{\mathrm{p}}_{\mathrm{Hip}}), \theta_{\mathrm{Thigh}}, \theta_{\mathrm{Knee}}$, and $\theta_{\mathbf{Ankle}} = (\theta^{\mathrm{r}}_{\mathrm{Ankle}}, \theta^{\mathrm{p}}_{\mathrm{Ankle}})$. It simplifies the implementation of dynamic walking, because its dimensions are less dependent than the individual joints angles. By changing only one target, e.g. the target leg extension γ, multiple joints are actuated in a coordinated way.

B. Central Clock

A central clock $-\pi \leq \phi_{\mathrm{Trunk}} < \pi$ running in the trunk determines the step frequency ψ. Both legs derive their own gait phase $-\pi \leq \phi_{\mathrm{Leg}} < \pi$ by shifting the trunk phase by $\pm\pi/2$. Based on its gait phase, each leg generates trajectories for its leg extension, leg angle, and foot angle.

C. Omnidirectional Walking

The target walking direction, speed, and rotation is specified by the vector $\mathbf{v_{Robot}} = (v^{\mathrm{x}}_{\mathrm{Robot}}, v^{\mathrm{y}}_{\mathrm{Robot}}, v^{\theta}_{\mathrm{Robot}})$, where $v^{\mathrm{y}}_{\mathrm{Robot}}$ denotes the speed in forward direction, $v^{\mathrm{x}}_{\mathrm{Robot}}$ is the lateral speed, and $v^{\theta}_{\mathrm{Robot}}$ determines the robot's rotation around the vertical axis.

The three key ingredients for generating omnidirectional walking are lateral shifting of the robot's center of mass, shortening of the leg which is not needed for support and movement of the legs in walking direction. The shortened leg is moved quickly into the walking direction. At the same time, the supporting leg has maximal extension and is moved slowly against the walking direction.

The swing amplitudes $\mathbf{a_{Robot}} = (a^{\mathrm{r}}_{\mathrm{Robot}}, a^{\mathrm{p}}_{\mathrm{Robot}}, a^{\mathrm{y}}_{\mathrm{Robot}})$ for the leg motion in target direction are derived from the target speed vector as follows:

$$(a^{\mathrm{r}}_{\mathrm{Robot}}, a^{\mathrm{p}}_{\mathrm{Robot}}) = (\mathsf{asin}(\frac{v^{\mathrm{x}}_{\mathrm{Robot}}}{\psi \cdot l}), \mathsf{asin}(\frac{v^{\mathrm{y}}_{\mathrm{Robot}}}{\psi \cdot l})), \quad (3)$$

$$a^{\mathrm{y}}_{\mathrm{Robot}} = \frac{v^{\theta}_{\mathrm{Robot}}}{\psi}, \quad (4)$$

where l denotes the leg length.

In the following, we describe the generation of the individual components of omnidirectional walking. The design of the trajectories was based on kinematics and dynamic constraints. All curves are as smooth as possible to limit accelerations. The parameters have been determined using feedback from the real robot by first walking with zero speed, next walking into each of the three directions with increasing speeds, and finally observing combinations of walking directions.

• Shifting: Because a sinusoid moves between two extreme points with the smallest accelerations, the lateral shifting of the robot's center of mass is done in a sinusoidal way:

$$\theta_{\mathrm{Shift}} = a_{\mathrm{Shift}} \cdot \mathsf{sin}(\phi_{\mathrm{Leg}}), \quad (5)$$

where $a_{\mathrm{Shift}} = 0.12 + 0.08 \cdot ||(a^{\mathrm{r}}_{\mathrm{Robot}}, a^{\mathrm{p}}_{\mathrm{Robot}})|| + 0.7 \cdot |a^{\mathrm{r}}_{\mathrm{Robot}}|$ is the shifting amplitude. The shifting amplitude increases with the gait speed as well as with the lateral speed. Both, the leg and foot roll angles are used to shift the robot:

$$\theta_{\mathrm{LegShift}} = \theta_{Shift} \quad (6)$$

$$\theta_{\mathrm{FootShift}} = -0.5 \cdot \theta_{\mathrm{Shift}}. \quad (7)$$

This keeps the upper body upright in the lateral plane.

• Shortening: As the robot shifts to a side, the opposite leg is not needed to support the weight. It can be shortened. The time course of the shortening is determined by the shortening phase:

$$\phi_{\mathrm{Short}} = v_{\mathrm{Short}}(\phi_{\mathrm{Leg}} + \pi/2 + o_{\mathrm{Short}}), \quad (8)$$

where $v_{\mathrm{Short}} = 3.0$ determines the duration of the shortening and $o_{\mathrm{Short}} = -0.05$ determines the phase shift of the shortening relative to the lateral weight shifting. A cosine now produces smooth transitions between the fully extended leg and the shortened leg:

$$\gamma_{\mathrm{Short}} = \begin{cases} -a_{\mathrm{Short}} 0.5(\mathsf{cos}(\phi_{\mathrm{Short}}) + 1) & \text{if } -\pi \leq \phi_{\mathrm{Short}} < \pi \\ 0 & \text{otherwise,} \end{cases}$$

where the shortening amplitude $a_{\mathrm{Short}} = 0.2 + 2 \cdot ||(a^{\mathrm{r}}_{\mathrm{Robot}}, a^{\mathrm{p}}_{\mathrm{Robot}})||$ increases with the gait speed. To lift the foot at the end pointing into the walking direction, a similar term is used:

$$\theta_{\mathrm{FootShort}} = \begin{cases} -a^{\mathrm{p}}_{\mathrm{Robot}} .125(\mathsf{cos}(\phi_{\mathrm{Short}}) + 1) & \text{if } -\pi \leq \phi_{\mathrm{Short}} < \pi \\ 0 & \text{otherwise.} \end{cases}$$

This avoids accidental contact between the leading part of the foot and the ground during swing.

• Loading: Immediately after the leg is fully extended and the heel landed, it is shortened a second time, in order to facilitate loading of this leg:

$$\phi_{\text{Load}} = v_{\text{Load}} \cdot \text{piCut}(\phi_{\text{Leg}} + \pi/2 - \pi/v_{\text{Short}} + o_{\text{Short}}) - \pi, \quad (9)$$

where $v_{\text{Load}} = 3$ determines the duration of the second sortening. The function piCut(.) maps its argument to the range $[-\pi, \pi)$ by adding multiples of 2π. The amplitude of the second shortening depends on the sagittal amplitude:

$$a_{\text{Load}} = 0.025 + 0.5 \cdot (1 - \cos(|a^{\text{p}}_{\text{Robot}}|)). \quad (10)$$

The shortening is also computed using a cosine:

$$\gamma_{\text{Load}} = \begin{cases} -0.5(\cos(\phi_{\text{Load}}) + 1) & \text{if } -\pi \leq \phi_{\text{Load}} < \pi \\ 0 & \text{otherwise} \end{cases}. \quad (11)$$

• Swinging: After the leg has been unloaded and shortened, it is moved quickly into the walking direction. This swing is reversed slowly during the rest of the gait cycle. The time couse of the swing is described by the swing phase:

$$\phi_{\text{Swing}} = v_{\text{Swing}}(\phi_{\text{Leg}} + \pi/2 + o_{\text{Swing}}), \quad (12)$$

where $v_{\text{Swing}} = 2.0$ is the swing speed and $o_{\text{Swing}} = -0.15$ is the phase shift of the swinging. While the swinging is sinusiodal, the reverse motion is linear:

$$\theta_{\text{Swing}} = \begin{cases} \sin(\phi_{\text{Swing}}) & \text{if } -\pi/2 \leq \phi_{\text{Swing}} < \pi/2 \\ b(\phi_{\text{Swing}} - \pi/2) - 1) & \text{if } \pi/2 \leq \phi_{\text{Swing}} \\ b(\phi_{\text{Swing}} + \pi/2) + 1) & \text{otherwise.} \end{cases} \quad (13)$$

The speed of the reverse motion is:

$$b = -(2/(2 \cdot \pi \cdot v_{\text{Swing}} - \pi).$$

The swing is done with the leg angle and balanced partially with the foot angle:

$$\begin{aligned} \theta^{\text{r}}_{\text{LegSwing}} &= \text{ls} \cdot a^{\text{r}}_{\text{Robot}} \cdot \theta_{\text{Swing}}, & (14)\\ \theta^{\text{p}}_{\text{LegSwing}} &= a^{\text{p}}_{\text{Robot}} \cdot \theta_{\text{Swing}}, & (15)\\ \theta^{\text{y}}_{\text{LegSwing}} &= \text{ls} \cdot a^{\text{y}}_{\text{Robot}} \cdot \theta_{\text{Swing}}, & (16)\\ \theta^{\text{r}}_{\text{FootSwing}} &= 0.25 \cdot a^{\text{r}}_{\text{Robot}} \cdot \theta_{\text{Swing}} & (17)\\ \theta^{\text{p}}_{\text{FootSwing}} &= \text{ls} \cdot 0.25 \cdot a^{\text{p}}_{\text{Robot}} \cdot \theta_{\text{Swing}}, & (18) \end{aligned}$$

where the leg sign is ls=-1 for the left leg and ls=1 for the right leg.

• Balance: The robot is balanced by tilting it with every step in the sagittal and lateral plane and by adding offsets to the leg and foot angles:

$$\begin{aligned} \theta^{\text{r}}_{\text{FootBal}} &= 0.5 \cdot \text{ls} \cdot a^{\text{r}}_{\text{Robot}} \cdot \cos(\phi_{\text{Leg}} + 0.35), & (19)\\ \theta^{\text{p}}_{\text{FootBal}} &= 0.02 + 0.08 \cdot a^{\text{p}}_{\text{Robot}} & (20)\\ &\quad -0.04 \cdot a^{\text{p}}_{\text{Robot}} \cdot \cos(2 \cdot \phi_{\text{Leg}} + 0.7), & (21)\\ \theta^{\text{r}}_{\text{LegBal}} &= .01 + \text{ls} \cdot a^{\text{r}}_{\text{Robot}} + |a^{\text{r}}_{\text{Robot}}| + .1 \cdot a^{\text{y}}_{\text{Robot}}. & (22) \end{aligned}$$

The leg roll angle makes sure that the legs do not collide while walking to the side or while rotating.

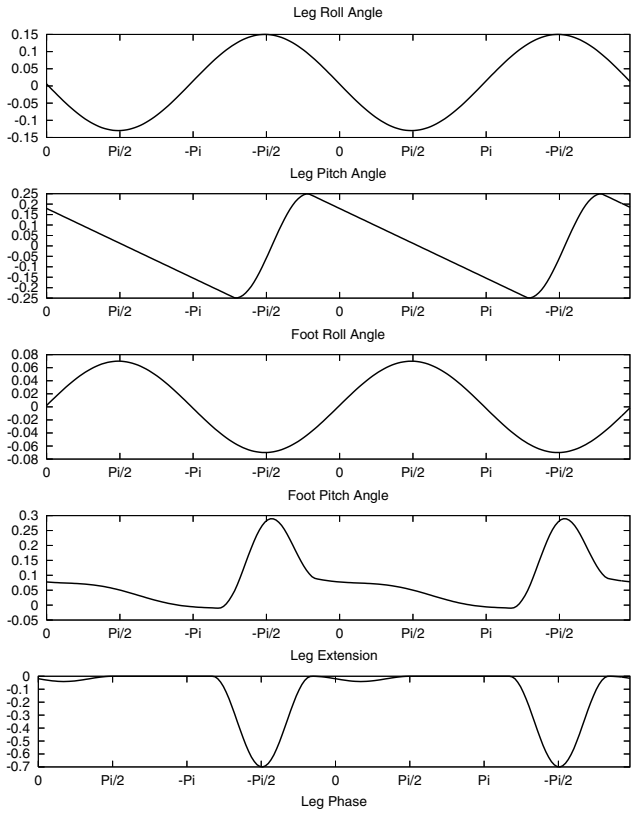


Fig. 3. Trajectories generating forward walking. See text for details.

• Output: The individual components of the walking motion are combined as follows:

$$\begin{aligned} \theta^{\text{r}}_{\text{Leg}} &= \theta^{\text{r}}_{\text{LegSwing}} + \theta_{\text{LegShift}} + \theta^{\text{r}}_{\text{LegBal}}, & (23)\\ \theta^{\text{p}}_{\text{Leg}} &= \theta^{\text{p}}_{\text{LegSwing}}, & (24)\\ \theta^{\text{y}}_{\text{Leg}} &= \theta^{\text{y}}_{\text{LegSwing}}, & (25)\\ \theta^{\text{r}}_{\text{Foot}} &= \theta^{\text{r}}_{\text{FootSwing}} + \theta_{\text{FootShift}} + \theta^{\text{r}}_{\text{FootBal}}, & (26)\\ \theta^{\text{p}}_{\text{Foot}} &= \theta^{\text{p}}_{\text{FootSwing}} + \theta_{\text{FootShort}} + \theta^{\text{p}}_{\text{FootBal}}, & (27)\\ \gamma &= \gamma_{\text{Short}} + \gamma_{\text{Load}}. & (28) \end{aligned}$$

The resulting trajectories are depicted in Fig. 3, Fig. 4, and Fig. 5 for the cases of forward walking ($a_{\text{Robot}} = (0, 0.25, 0)$), walking to the side ($a_{\text{Robot}} = (0.0625, 0, 0)$), and turning on the spot ($a_{\text{Robot}} = (0, 0, 0.25)$), respectively.

The arms of the robot are moved in a similar way as its legs. An arm moves synchronously with its contralateral leg.

VI. Results

Fig. 6, Fig. 7, and Fig. 8 show image sequences of the robot walking in forward direction, to the side, and turning on the spot, respectively. A video showing the smooth transition between and combination of these walking directions can be downloaded from www.NimbRo.net.

It can be observed that Jupp walks dynamically with relatively large steps. The upper body of the robot swings laterally, but tilts only little in the lateral and sagittal planes. Jupp keeps the knee of the stance leg straight. When walking forward with

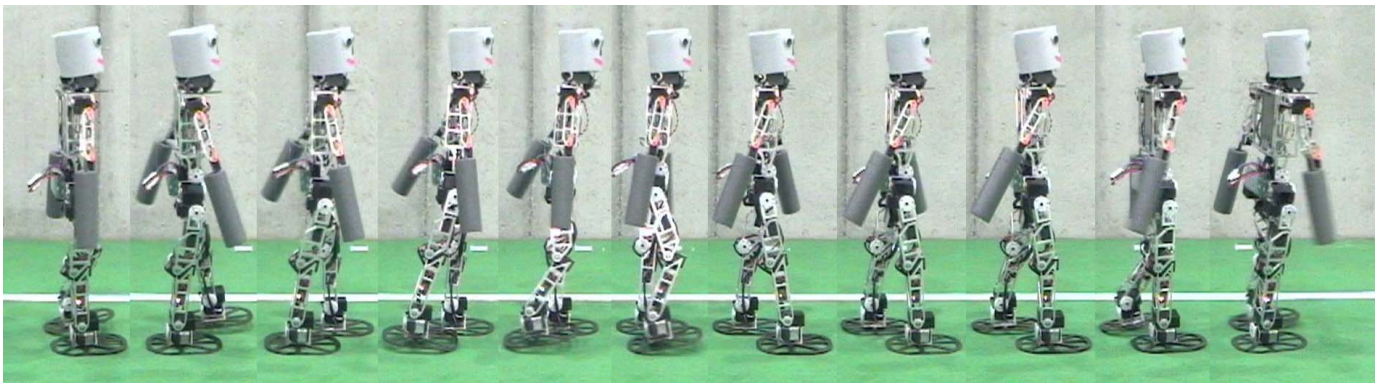


Fig. 6. Image sequence extracted from video of forward walking robot (every second frame).

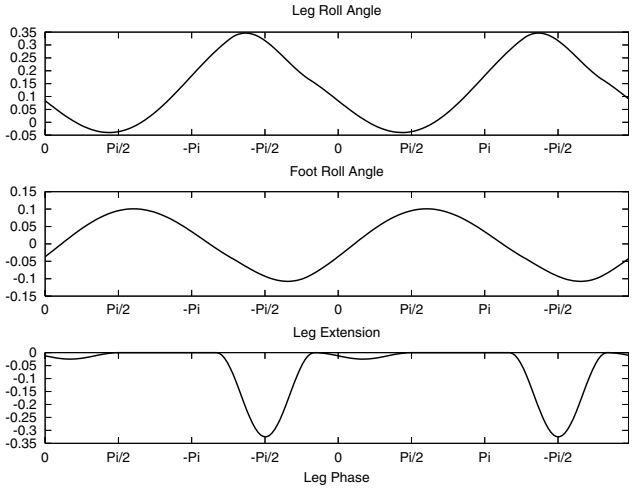


Fig. 4. Trajectories generating walking to the side. See text for details.

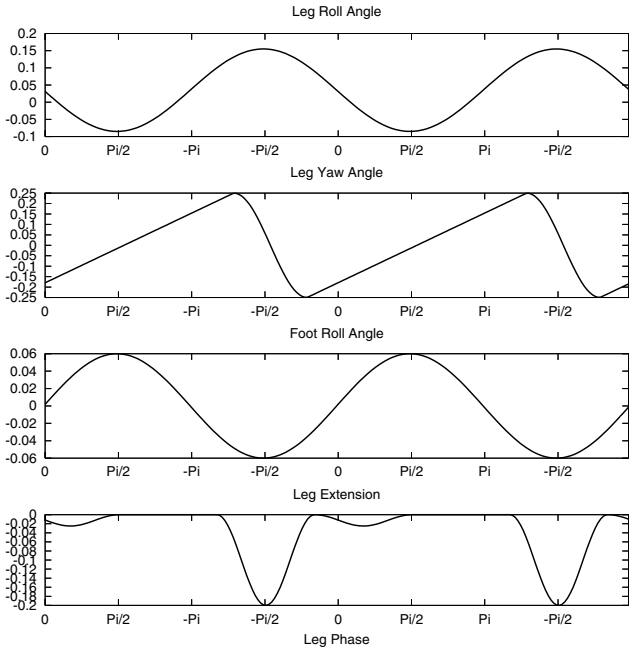


Fig. 5. Trajectories generating turning on the spot. See text for details.

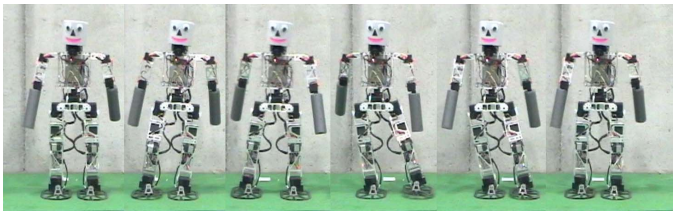


Fig. 7. Image sequence extracted from video of robot walking to the side (every fourth frame).

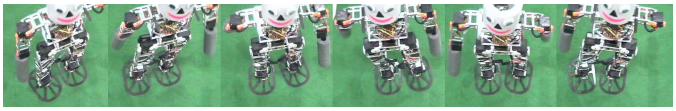


Fig. 8. Image sequence extracted from video of robot turning on the spot (every fourth frame).

$a^{\mathrm{p}}_{\mathrm{Robot}} = 0.25$, an average step size of 9.14cm was measured. With the step frequency of 1.573Hz this results in a speed of 14.4cm/s. When walking to the side with $a^{\mathrm{r}}_{\mathrm{Robot}} = 0.0625$, an average step size of 2.41cm was measured, yielding a lateral speed of 3.8cm/s. The rotation while turning with $a^{\mathrm{y}}_{\mathrm{Robot}} = 0.25$ is 15°/step, resulting in a turning speed of 23.6°/s. Energy consumption in the joints can be estimated from the servo temperatures after walking longer distances. While the hip-servos had the highest temperature, the knee servos were cool and the ankle servos had an intermediate temperature. This indicates that the straight stance leg relieves the knee joint from high torques.

VII. Conclusions

This paper described the online generation of trajectories for omnidirectional walking on two legs. The presented approach has a low computational complexity. We implemented it on the Pocket PC of our 19DOF humanoid robot Jupp. The resulting walking speed in forward direction was 14,4cm/s, which is quite high, given Jupp's size of 60cm.

The omnidirectional gait was not optimized for speed, but for stability and parameterizability. Because the gait is fully parameterizable, higher-level behaviors needed for playing soccer, such as approaching the ball, can smoothly vary

walking direction, walking speed, and rotational speed. This allows precise positioning of the robot for kicking, while rotating the robot towards the target for kicking.

In contrast to the chaining of predefined motion macros, our approach generates smooth robot trajectories, without the need to stop walking when changing direction or turning. It is also possible to continuously compensate for deviations from the desired walking direction using visual feedback. These features make our omnidirectional gait suitable for the use in the RoboCup soccer domain.

In addition to omnidirectional walking, we implemented kicking, getting-up behaviors, visual perception, self-localization, communication, and basic soccer skills for Jupp, which will be detailed elsewhere.

At RoboCup 2005 in Osaka, Japan, Jupp, its twin Sepp, and its larger sibling Max faced the world's best humanoid soccer robots. Both 60cm robots used the same omnidirectional gait. The parameters of the gait engine were adapted to the 70cm robot Max within few minutes. As team NimbRo our robots performed very well. Our robots were walking faster than almost all competitors, could maneuver around obstacles, walk around the ball while looking at it, and position themselves precisely for kicking. Consequently, they scored many goals. Our robots walked very stable when not disturbed. After collisions with other robots, some falls occurred, but our robots got up [30] and continued play. In the technical challenge, they came in 2nd and 3rd. Max won the penalty kick competition in the MediumSize class. The KidSize robots Jupp and Sepp reached the final in the 2 vs. 2 soccer games. Team Osaka won this exciting game 2:1. In the overall Best Humanoid ranking, our robots came in 2nd (KidSize) and 3rd (MediumSize), next only to the titleholder, Team Osaka.

It is not hard to transfer the described gait to other bipeds. We used slightly adapted versions of the described gait with success for robots ranging from 35cm to 120cm in size.

Our future work will go into two directions: preventing falls and automatic parameter optimization. To prevent falls, we use attitude feedback to detect external disturbances. Slowing down, tilting the robot against the disturbance, and lunge steps are possible postural responses to improve stability. The gait engine has been designed with few meaningful parameters. While we manually adjusted these parameters so far, based on systematic walking experiments, it should be possible to devise an automatic framework for parameter adjustment, e.g. by using evolutionary or reinforcement learning techniques.

ACKNOWLEDGMENT

The author would like to thank Michael Schreiber, who constructed Jupp's mechanics, Jörg Stückler who implemented the compass interface and designed getting-up behaviors, Johannes Schwenk, who designed getting-up behaviors, and Hauke Strasdat, who implemented computer vision algorithms. This work was supported by grant BE 2556/2-1 of Deutsche Forschungsgemeinschaft (DFG).

REFERENCES

[1] R. D'Andrea, "The Cornell RoboCup robot soccer team: 1999-2003," in *Handbook of Networked and Embedded Control Systems*, ser. Control Engineering. A Birkhäuser book, 2005.
[2] T. Kalmar-Nagy, R. D'Andrea, and P. Ganguly, "Near-optimal dynamic trajectory generation and control of an omnidirectional vehicle," *Robotics and Autonomous Systems*, vol. 46, pp. 47–64, 2004.
[3] D. Spenneberg and F. Kirchner, "Omnidirectional walking in an eight legged robot," in *Proc. Symp. on Robotics and Automation (ISRA)*, 2000.
[4] J. M. Porta and E. Celaya, "Body and leg coordination for omnidirectional walking in rough terrain," in *Proc. of CLAWAR*, 2000.
[5] B. Hengst, D. Ibbotson, S. B. Pham, and C. Sammut, "Omnidirectional locomotion for quadruped robots," in *RoboCup 2001: Robot Soccer World Cup V*. Springer, 2002, pp. 368–373.
[6] S. Behnke, J. Müller, and M. Schreiber, "Toni: A soccer playing humanoid robot," in *Proc. of 9th RoboCup Int. Symp., Osaka*, 2005.
[7] Q. Huang, K. Yokoi, S. Kajita, K. Kaneko, H. Arai, N. Koyachi, and K. Tanie, "Planning walking patterns for a biped robot," *IEEE Tr. on Robotics and Automation*, vol. 17, no. 3, pp. 280–289, 2001.
[8] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka, "The development of Honda humanoid robot," in *Proc. of ICRA*, 1998, pp. 1321–1326.
[9] F. Pfeiffer, K. Löffler, and M. Gienger, "The concept of jogging johnnie," in *Proc. of ICRA*, vol. 3, 2002, pp. 3129–3135.
[10] M. Vukobratovic and B. Borovac, "Zero-moment point – thirty five years of its life," *Int. J. of Humanoid Robotics*, vol. 1(1), pp. 157–173, 2004.
[11] S. Kajita, F. Kanehiro, K. Kaneko, *et al.*, "Biped walking pattern generation by a simple three-dimensional inverted pendulum model," *Advanced Robotics*, vol. 17(2), pp. 131–147, 2003.
[12] J. Pratt, C.-M. Chew, A. Torres, *et al.*, "Virtual model control: An intuitive approach for bipedal locomotion," *Int. J. of Robotics Research*, vol. 20(2), pp. 129–143, 2001.
[13] S. Grillner, "Neurobiological bases of rhythmic motor acts in vertebrates," *Science*, vol. 228, no. 4696, pp. 143–149, 1985.
[14] J. B. Nielsen, "How we walk: Central control of muscle activity during human walking," *Neuroscientist*, vol. 9, no. 3, pp. 195–204, 2003.
[15] M. Morisawa, S. Kajita, K. Kaneko, *et al.*, "Pattern generation of biped walking constrained on parametric surface," in *Proc. of ICRA*, 2005.
[16] D. Sharon and M. van de Panne, "Synthesis of controllers for stylized planar bipedal walking," in *Proc. of ICRA*, 2005.
[17] T. McGeer, "Passive dynamic walking," *International Journal of Robotics Research*, vol. 9, no. 2, pp. 68–82, 1990.
[18] S. Collins, A. Ruina, R. Tedrake, and M. Wisse, "Efficient bipedal robots based on passive-dynamic walkers," *Science 307*, pp. 1082–1085, 2005.
[19] K. D. Mombaur, *Stability Optimization of Open-loop Controlled Walking Robots*, ser. Fortschritt-Bericht. VDI, 2001.
[20] K. Matsushita, M. Lungarella, C. Paul, and H. Yokoi, "Locomoting with less computation but more morphology," in *Proc. of ICRA*, 2005.
[21] F. Iida and R. Pfeifer, "Self-stabilization and behavioral diversity of embodied adaptive locomotion," in *Embodied Artificial Intelligence*, ser. LNCS, vol. 3139, 2004, pp. 119–129.
[22] M. W. Tilden, "Neuromorphic robot humanoid to step into the market," *The Neuromorphic Engineer*, vol. 1, no. 1, p. 12, 2004.
[23] Sony, "Dream Robot QRIO. http://www.sony.net/qrio."
[24] Honda, "ASIMO. http://world.honda.com/asimo."
[25] Kondo Kagaku Co., Ltd., "KHR-1. http://www.kondo-robot.com."
[26] Vstone Co., Ltd., "http://www.vstone.co.jp."
[27] C. Zhou and P. K. Yue, "Robo-Erectus: a low-cost autonomous humanoid soccer robot," *Advanced Robotics 18(7)*, pp. 717–720, 2004.
[28] S. Behnke, J. Müller, and M. Schreiber, "Using handheld computers to control humanoid robots," in *Proc. of DARH, Yverdon*, 2005.
[29] S. Behnke and R. Rojas, "A hierarchy of reactive behaviors handles complexity," in *Balancing Reactivity and Social Deliberation in Multi-Agent Systems*. Springer, 2001, pp. 125–136.
[30] J. Stückler, J. Schwenk, and S. Behnke, "Getting back on two feet: Reliable standing-up routines for a humanoid robot," in *Proc. of The 9th Int. Conf. on Intelligent Autonomous Systems (IAS-9), Tokyo*, 2006.