Walking Gait Generation with Foot Placement Control for the HOAP-3 Humanoid Robot

Yazhou Huang, Robert Backman and Marcelo Kallmann
UCM School of Engineering Technical Report TR-2011-002
University of California, Merced
{yhuang6,rbackman,mkallmann}@ucmerced.edu

***Abstract*— While biology-inspired approaches for computing walking gaits are often based on joint oscillators, whenever precise control of feet placement is needed a more specific approach based on explicit footstep control is usually employed. This paper presents a simple hybrid approach based on a gait generator procedure that generates a patterned walking gait based on trajectories to be followed by the feet and the center of mass of the robot. While the trajectories can be modulated according to high-level parameters offering realtime control of the produced gait, the trajectories can also be modified in order to achieve precise foot placements. Robust balance is achieved by integrating a reactive realtime module which adjusts the overall posture of the robot according to the readings of the feet pressure sensors. We apply our gait generation method to the HOAP-3 humanoid platform and several results are presented.**

I. Introduction

Walking is a key capability of humanoid robots and at the same time it is one of the most difficult motions to achieve precise control.

This paper presents a hybrid approach for generating walking gaits which is based on a pattern generator that precisely controls feet placement according to parameterized trajectories. We apply our method to the HOAP-3 humanoid robot, which is illustrated in Figure 1.

In order to maintain balance when executing the generated gait in the robot, our method includes a reactive module which adjusts the center of mass of the robot according to the readings of the feet sensors in order to achieve robust balance maintenance.

The effectiveness of our gait generator and balance maintenance methods is demonstrated with several videos which show the humanoid robot successfully walking in different situations, and in particular executing paths among obstacles.

II. Related Work

Research on humanoid locomotion involves many challenging aspects and different approaches are available in the literature for the purpose of humanoid walking gait generation. In general these approaches can be classified into three main categories described below.

In the first category, a limited knowledge of the robot dynamics is used, as for example: the location of the center of mass (CoM), and the total angular momentum. Since the walking controller knows little about the system structure, these approaches usually rely on a feedback control loop such as the Inverted Pendulum Model [1] [5].

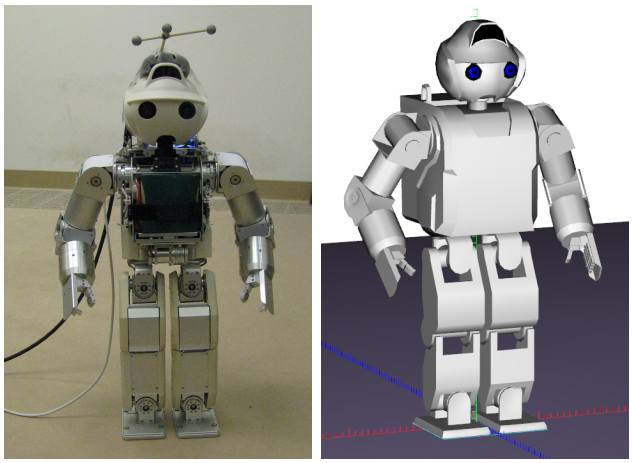

Fig. 1. The HOAP-3 humanoid robot (left) and its virtual model (right).

In contrast, a second category of methods can be identified when precise knowledge of the robot dynamics is availble, for example: the CoM, the Zero Moment Point (ZMP) and inertia of each link of the robot, etc. The performance of these methods will mainly rely on the accuracy of the computational model [2][9] .

The third category is represented by Central Pattern Generators (CPG) which are essentially, in most of the cases, neural networks consisting of various mathematical neuron models. The parameters of the models are optimized with multiple objectives, and the outputs of the neurons are utilized as the target angles of corresponding joints [10] [3] [4]. However, less attention has been given for pattern generators addressing precise placement of each foot step during walk generation, which is a critical aspect when walking among obstacles, which is the typical scenario of approaches based on foot stepping planning [8].

In this paper we present a walk gait generator for bipedal humanoid robots with explicit feet and CoM trajectory manipulation in order to achieve realtime parameterization of the produced gait and at the same time precise foot

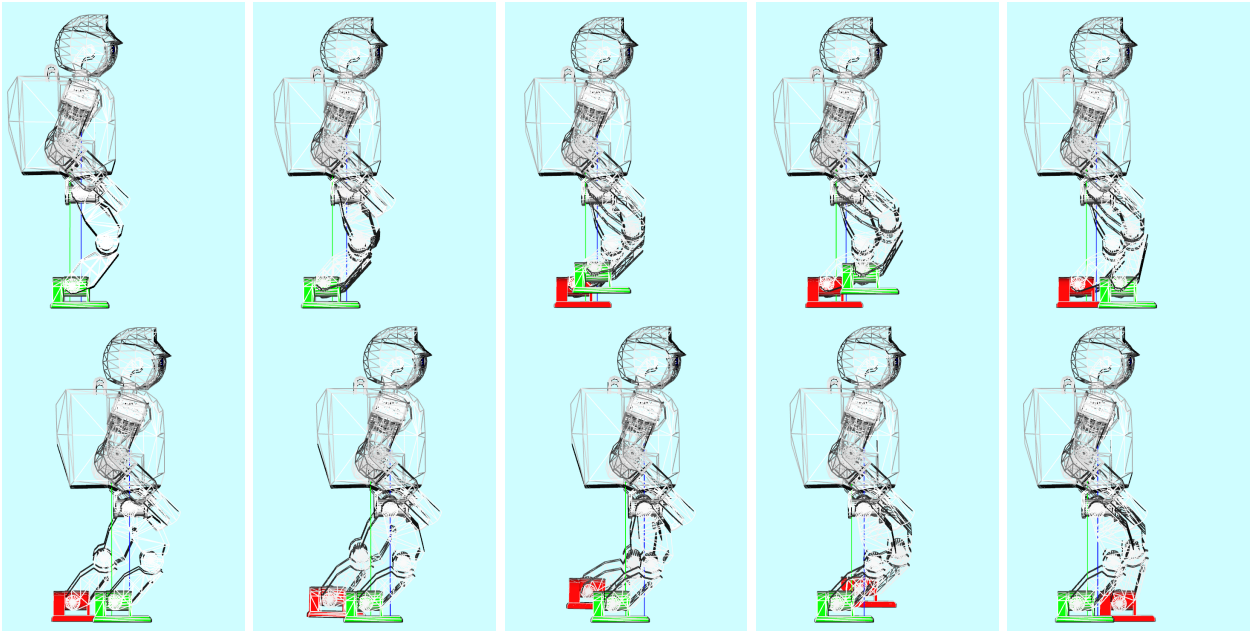

Fig. 2. The snapshots show results of the generated walking gait. The left foot end-effector (red) and the right one (green) follow a predefined curve (Figure 3) to step forward. IK is applied for both end-effectors at every frame to either maintain CoM projection at the center of the supporting footplate, or to shift CoM from one foot to the other. First row: starting from standing pose, shifting CoM projection (green line) to left foot center (single support), making a step with right foot, shifting CoM from left foot to right foot. Second row: making a step with the left foot while maintaining the CoM on top of the right foot center, then alternating the support foot in order to make the next step. The vertical blue line indicates the projection of the root joint of the robot on the floor.

placements. Our method is based on executing trajectories solved by a fast analytical Inverse Kinematics (IK) solution, which was designed specifically for the HOAP-3 humanoid robot.

Our approach offers several advantages, in particular precise feet placement for the walk gait generation can be guaranteed at the maximum level while balance is maintained by placing the CoM floor projection at optimal locations at each frame. As our method is still based on a gait generator we are also able to parameterize the output with high-level intuitive parameters such as: walk speed, step distance, turning angle, step height, etc, that can continuously vary within certain ranges, resulting in a fine control of each step. This precise foot placement with high level steering control is ideal for scenarios involving locomotion among obstacles.

III. Walking Gait Generation

The walking gait generation can be decomposed into a start phase, a walking phase and a return phase. The gait is generated by explicitly moving three joints: the root joint, the left foot, and the right foot. At each time step the left and right feet are considered to be end-effectors to be solved by the IK, while the desired location for the root joint can be freely moved while the IK maintain the feet precisely attached to the floor.

Figure 2 illustrates the process: from the start pose (double support mode), the root joint is first translated to the top of the supporting foot until the CoM projection lies perfectly at the center of the supporting footplate (single support). The other foot is now free to make a step while maintaining the CoM unchanged. Note that the CoM is constantly changing while the robot is moving and that the CoM can not be directly manipulated, but it can be indirectly controlled by moving the root joint position, i.e. moving the root joint to the desired CoM location minus the difference vector between the root joint location and the CoM of the robot in the previous timestep. In order to achieve precise control, the generator applies the IK for a few iterations at each timestep in order to get to a pose that best satisfies the CoM constraint.

When making a step, the foot end-effector follows a predefined curve (see Figure 3) that lifts the foot off the ground and moves forward (also sideways if stepping with a turning angle). The root joint is then maintained at the same hight, and turns in the same way as the stepping foot but only for half of the angle, see Figure 4. This increases the chances for solving the IK problem due to the limited range of motion in the leg joints of the HOAP-3 robot. The step length, turning angle, foot lift height are among several parameters which can be continuously controlled during the gait generation.

If the parameters change drastically so that the IK can not solve for the produced trajectory due to reachability or joint range limits constraints, the generator on the fly adjusts the end-effectors until solutions can be found. This is usually done by adding rotation increments to adjust the stepping foot (the current end-effector) and also by translating it away from the body so that it can be lifted higher (in the case of not being reachable). If no IK solutions can found for the

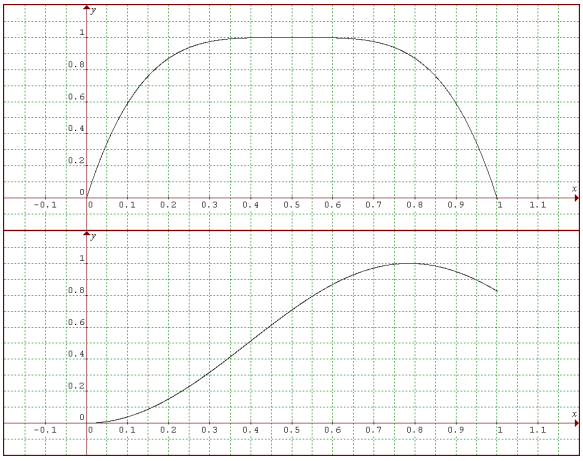

Fig. 3. Curves used by the walking gait generator: foot lift curve $y = 1-(2x-1)^4, x \in [0,1]$ (top), and the adjustment of the leg base during single support (see Figure 6) $y = 1/2sin(4t-\pi/2)+1/2, t \in [0,1]$ (bottom).

given set of parameters, the parameters will still be limited with the priority of maintaining balance.

The generator outputs one pose at each time interval, and it sends it immediately to a low-level communication module. The module interpolates from the current configuration of the robot, which is received by a network connection at a fixed frame rate (60 fps). See section VI for details. By varying the time interval, turning angle and step length, the generator can be easily be applied for applications such as footstep placement for path following, or walking interactive control from a joystick, etc.

IV. Balance Control

When the generated walking gaits are simply applied to the robot with no further corrections, the robot is able to walk however the balance is usually lost when the CoM transition becomes too fast. We implemented two simple controllers to solve the balance maintenance problem: leg base tilt correction and CoM offset correction. The humanoid looses balance during a fast gait mainly because it lacks a fast response to translate the CoM before entering (and during) a single support phase.

We apply an additional rotation for the supporting leg for balance compensation, which largely follows the sinusoidal curve shown in Figure 3-bottom. The x-axis is the time and the y-axis is the magnitude of such compensation, which makes the CoM projection overshoot over the desired target and increase the transition speed. The magnitude of the maximum overshoot angle and the parameters of this curve were adjusted during several walking tests until a suitable shape was obtained.

The CoM offset is a time-variant 2-D vector. It adds on top of the supporting footplate center as the desired location of the CoM projection, allowing subtle balance tuning during single support phase. Four pressure sensors are located below each footplate, from which the Center of Pressure (CoP) can be acquired and it is then used as feedback for on-line adjustment of CoM offset. A generic proportional-derivative feedback control of the offset vector is applied in order to reduce the oscillation of CoP.

The results of the balance control are shown in 5. As we can clearly see, the CoP trajectories obtained with the walking generator with the balance controller active have much less variations than the curves obtained without using the balance controller. Note that the shape of the curves exhibit a lot of noise as they were reconstructed with the real readings obtained from the pressure sensors, which are noisy and have varying range of response. Nevertheless the improvement in the trajectories can be clearly observed when the balance controller is activated. In Figure 5 the robot performs about 8 steps going straight, and readings from the 8 pressure sensors located at the bottom of the feet are used to reconstructed the CoP trajectory.

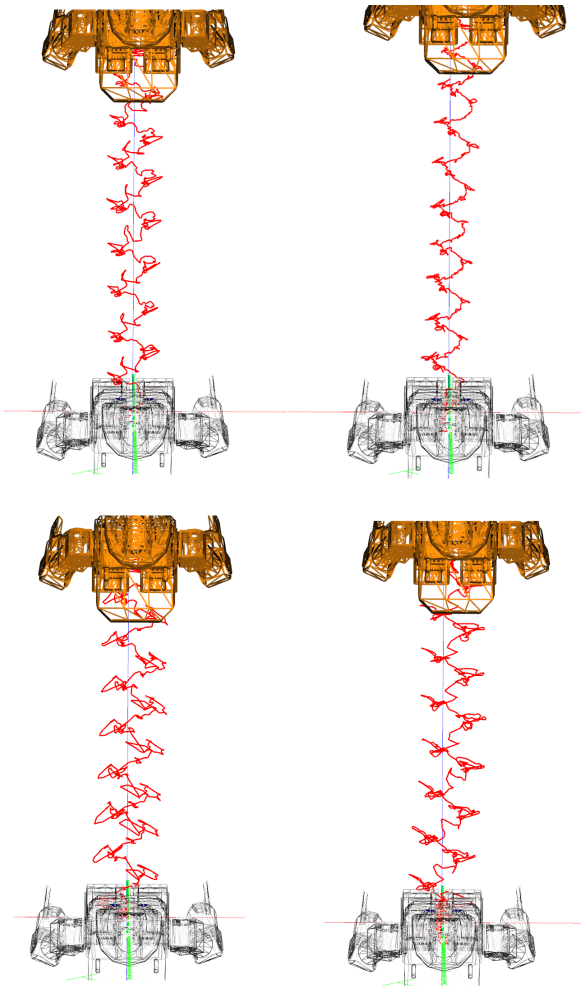

Fig. 5. Comparison of CoP trajectories with and without on-line adjustments: upper row: slow walk, lower row: fast walk; left column: without adjustment, right column: with on-line adjustment, which obviously reduces the oscillation

The results obtained with the balance controller greatly improve the motion and clearly produce a much smoother

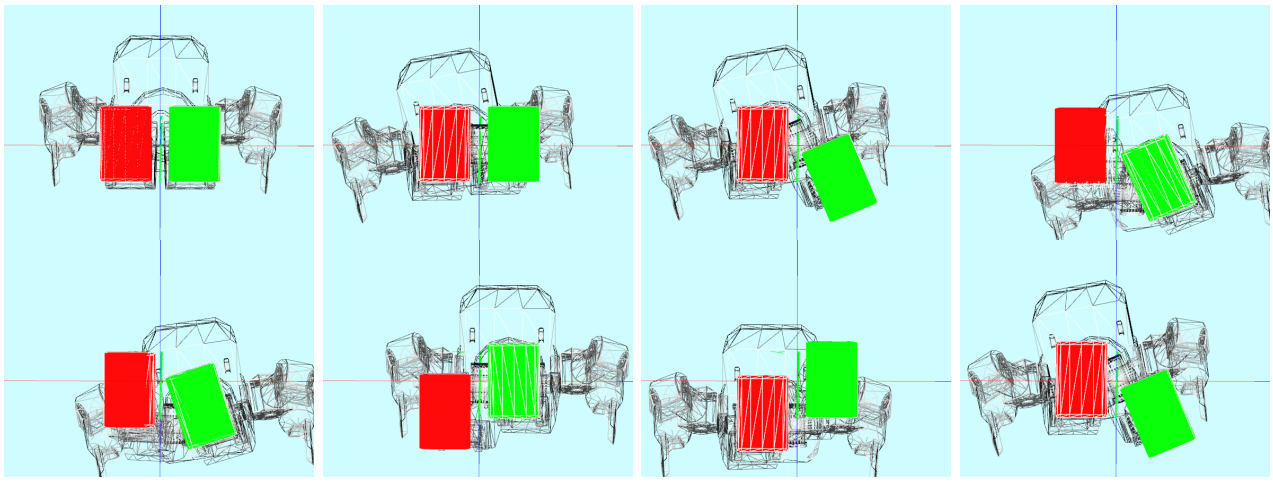

Fig. 4. Snapshots of making a right turn (bottom view). First row: starting from a standing posture, the CoM projection is shifted to the left footplate center (single support), and a right-turn stepping with the right foot is made. The upper body only turns half of the angle. Second row: shifting the CoM from the left foot to the right foot and then making a forward step with the left foot while maintaining the CoM on top of the right foot center and rotating back the upper body.

and balanced walking motion, which is demonstrated in examples with the robot which are available in the video accompanying this paper.

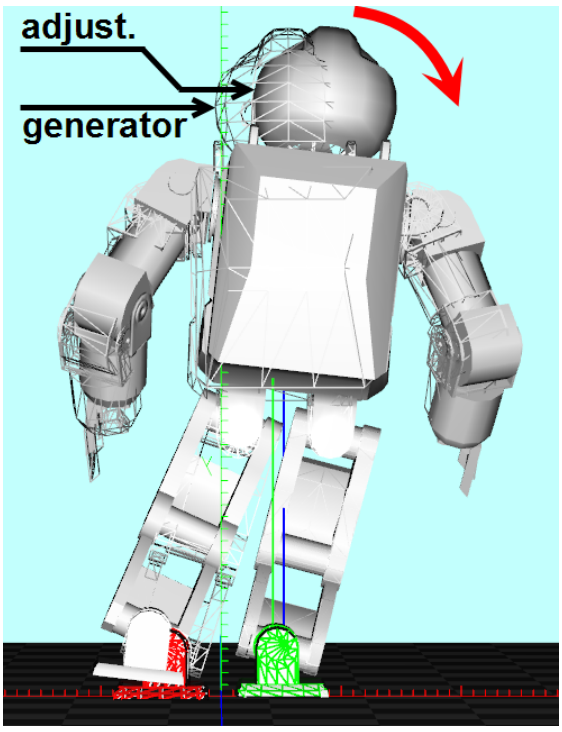

Fig. 6. Adjustment is made to reduce lateral oscillation during single support.

V. Finding Paths with Clearance

Our walking gait generator is suitable for precise walking control and we present as example the use of a path planner to compute trajectories among obstacles which can be executed by our walking controller.

Given an initial point p_{init}, a goal point p_{goal}, and a clearance distance r, the collision-free path $P(p_{init}, p_{goal}, r) = (p_0, p_1, \ldots, p_n)$ is described as a polygonal line with vertices p_i, $i \in \{1, \ldots, n\}$, describing the solution path. As the path turns around obstacles with distance r from the obstacles, every corner of the path is a circle arc which is approximated with points, making sure that the polygonal approximation remains of r clearance from the obstacles.

In order to compute $P(p_{init}, p_{goal}, r)$, we have extended an existing triangulation-based path finding method [6, 7] with the capability of generating paths with guaranteed clearance r from the obstacles. The original method is based on three parts: first a constrained Delaunay triangulation is performed for exactly describing the polygonal obstacles, then an A* search over the adjacency graph connecting the free triangles is performed until a sequence of triangles joining p_{init} and p_{goal} is found, and finally the shortest path inside the triangle sequence can be computed with a simple linear pass using the *funnnel* algorithm.

In order to extend the method for computing paths with r-clearance, several modifications to the main algorithm were needed: 1) each expansion during the A* search was extended to include tests for checking if the triangle being expanded can be traversed by a disc of radius r without collisions, 2) a triangle refinement procedure was also needed in order to correctly handle all sizes and shapes of the triangles being traversed, 3) the linear *funnel pass* described in the original method [6, 7] was extended to work with circles of radius r and several special cases happening at the beginning and end of the path were also specifically tested and handled. We omit details about these extensions since their explanation are overly long and not directly related to our present work on walking control. Our main purpose here is to present that collision free paths with guaranteed clearance r can be efficiently computed and are suitable for

executaion with the proposed walking gait generator.

Figure 7 shows a collision-free path and channel produced by our method. The figure also shows the node expansions performed during the search for the path. In this way we perform the needed global search for a path very efficiently in the triangulation, avoiding resolution-dependent methods such as when using grid-based subdivisions. Note that the number of vertices n for describing the polygonal obstacles is much smaller than the number of cells needed in grid-based representations, and this reduced number of cells employed by our method makes our approach for path finding very efficient.

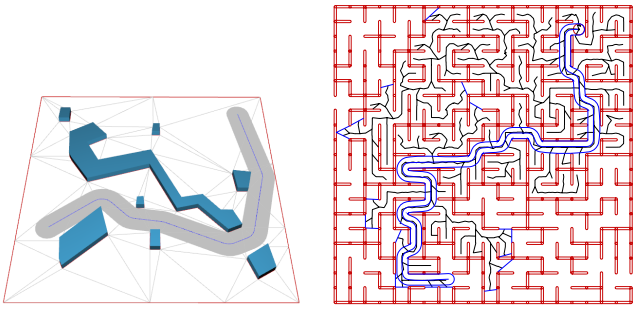

Fig. 7. The left image illustrates a computed collision-free path and channel with given clearance radius. The right image shows the expanded nodes of the A* search during the path computation.

VI. Realtime Execution in the HOAP Platform

There are mainly three components to the network interface controlling the robot: a user interface client application, a server application running on the robot's on board computer, and a real-time module that has dedicated access to the motor control board.

The user interface is running on a dedicated computer, and is where the motion is generated. For each frame of motion the posture of the robot is converted from joint angles to encoder values and is put into a message struct along with the step size for the real-time module. The main method to control the overall speed of the robot is by adjusting the step size for each posture sent. The message is then sent over the network to the server on the robotson board computer. When the server receives a posture message from the UI it relays it directly to the real-time module. When the real-time module receives a message it places it into a circular queue. The real-time module starts interpolating the postures and sends each increment to the motor control board.

For each step the real-time module pulls a posture from the queue along with the step size. The number of interpolation steps is determined by the largest difference in motor positions between the current and next posture divided by the step size. The increment for each motor is given by the difference in position divided by the number of steps. This ensures that for each move all of the motors start and end at the same time and the overall motion is continuous. Each time a posture is taken from the queue a data log is created that includes the pressure sensors in the feet,accelerometers,gyros and the corresponding encoder values. This information is then sent back through the server to the user interface for display and optimization.

There is also a separate computer which is dedicated to tracking the robot and any obstacles. A separate server is set up on this computer which enables retrieval of the global position and orientation of the robot and the obstacles any time that it is queried. With the combination of the sensor data log from the robot and the global position and orientation from the Tracking system an accurate depiction of the robot and its environment can be made in real-time.

Fig. 8. Overall system architecture for controlling the robot.

VII. Results

Figure 9 illustrates the effectiveness of the reactive balance control applied in conjunction with the walking gait generator. Figure 10 shows an example of a path planned among obstacles which was computed by the described triangulation-based decomposition planning approach. The figure also shows several snapshots of the robot executing the path among obstacles.

The path execution of Figure 9 is performed by adjusting in realtime the gait generator parameters such that feet placements following the given path are achieved. The generated walking gait is able to precisely execute the given path. Our results indicate that the proposed overall approach for walking generation is well suited for such types of tasks requiring fine walking control.

VIII. Conclusions and Future Work

This report describes a simple walking gait generator method that generates parameterized walking gaits with precise feet placements. We successfully applied our gait generation method to the HOAP-3 humanoid platform and several results were presented. As future work we intend to add dynamic balance mechanisms in order to improve the smoothness and reliability of the walking.

Acknowledgments: the presented work was partially funded by NSF award BCS-0821766.

References

[1] A. Goswami, B. Thuilot, B. Espiau, and L. G. Gravir. A study of the passive gait of a compass-like biped robot: Symmetry and chaos. *International Journal of Robotics Research*, 17:1282–1301, 1998.

[2] Q. Huang, K. Yokoi, S. Kajita, K. Kaneko, H. Arai, N. Koyachi, and K. Tanie. Planning walking patterns

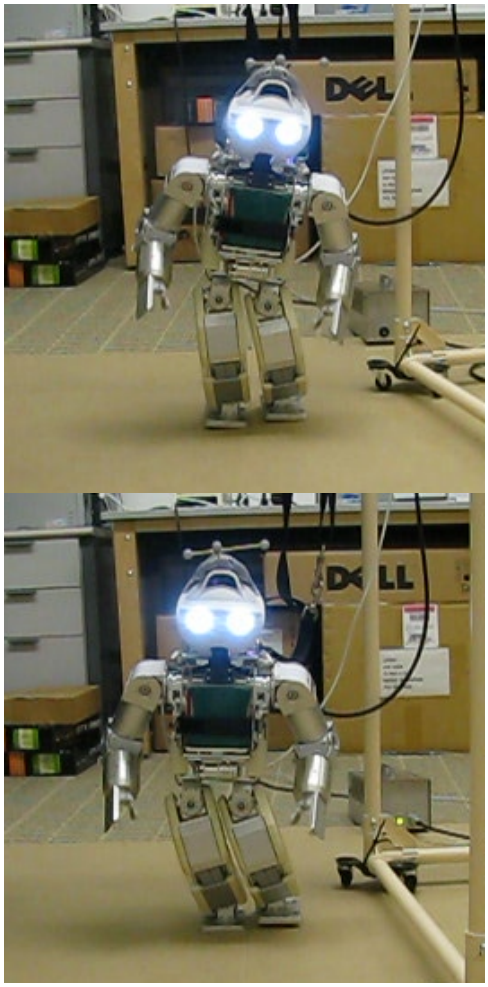

Fig. 9. When running without the reactive balance control module the robot often looses balance (top). With the balance control the robot remains stable with a good stance (bottom).

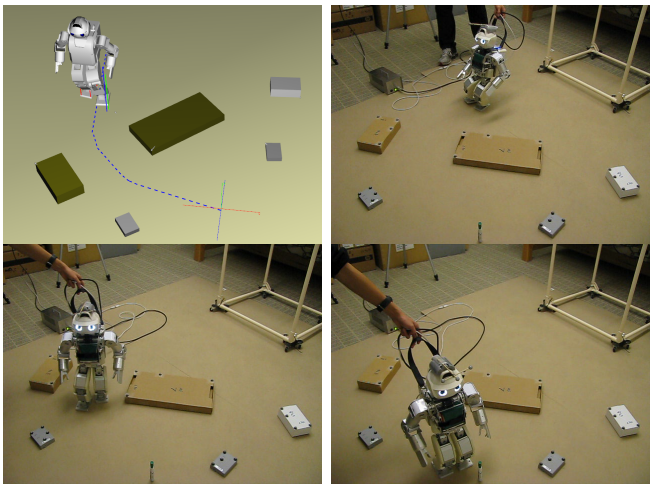

Fig. 10. A path planning and execution example.

for a biped robot. *Robotics and Automation, IEEE Transactions on*, 17(3):280–289, Jun 2001. ISSN 1042-296X. doi: 10.1109/70.938385.

[3] H. Inada and K. Ishii. Behavior generation of bipedal robot using central pattern generator(cpg) (1st report: Cpg parameters searching method by genetic algorithm). In *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 3, pages 2179–2184 vol.3, Oct. 2003. doi: 10.1109/IROS.2003.1249194.

[4] H. Inada and K. Ishii. Bipedal walk using a central pattern generator. *International Congress Series*, 1269: 185 – 188, 2004. Brain-Inspired IT I. Invited papers of the 1st Meeting entitled Brain IT 2004.

[5] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Yokoi, and H. Hirukawa. A realtime pattern generator for biped walking. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 31–37, 2002.

[6] M. Kallmann. Shortest paths with arbitrary clearance from navigation meshes. In *Proceedings of the Eurographics / SIGGRAPH Symposium on Computer Animation (SCA)*, 2010.

[7] M. Kallmann, H. Bieri, and D. Thalmann. Fully dynamic constrained delaunay triangulations. In H. M. L. L. G. Brunnett, B. Hamann, editor, *Geometric Modelling for Scientific Visualization*, pages 241–257. Springer-Verlag, Heidelberg, Germany, first edition, 2003. ISBN 3-540-40116-4.

[8] J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue. Online footstep planning for humanoid robots. In *in Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA03*, pages 932–937, 2003.

[9] J. H. Park. Fuzzy-logic zero-moment-point trajectory generation for reduced trunk motions of biped robots. *Fuzzy Sets and Systems*, 134(1):189 – 203, 2003. ISSN 0165-0114. doi: DOI:10.1016/S0165-0114(02)00237-3.

[10] J. Shan, C. Junshi, and C. Jiapin. Design of central pattern generator for humanoid robot walking based on multi-objective ga. In *Intelligent Robots and Systems, 2000. (IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, volume 3, pages 1930–1935 vol.3, 2000. doi: 10.1109/IROS.2000.895253.