EUROGRAPHICS 2000 / M. Gross and F.R.A. Hopgood
(Guest Editors)

Volume 19 (*2000*), *Number 3*

Motion Balance Filtering

Seyoon Tak, Oh-young Song, and Hyeong-Seok Ko

SNU Human Animation Center, School of Electrical Engineering, Seoul National University

Abstract

This paper presents a new technique called motion balance filtering, which corrects an unbalanced motion to a balanced one while preserving the original motion characteristics as much as possible. Differently from previous approaches that deal only with the balance of static posture, we solve the problem of balancing a dynamic motion. We achieve dynamic balance by analyzing and controlling the trajectory of the zero moment point (ZMP). Our algorithm consists of three steps. First, it analyzes the ZMP trajectory to find out the duration in which dynamic balance is violated. Dynamic imbalance is identified by the ZMP trajectory segments lying out of the supporting area. Next, the algorithm modifies the ZMP trajectory by projecting it into the supporting area. Finally, it generates the balanced motion that satisfies the new ZMP constraint. This process is formulated as a constrained optimization problem so that the new motion resembles the original motion as much as possible. Experiments prove that our motion balance filtering algorithm is a useful method to add physical realism to a kinematically edited motion.

1. Introduction

Balancing is one of the major ingredients that constitute the physical realism of human character animation. Without it, the body does not look situated in a real world. This paper is about balancing the motion of articulated human characters.

In a mechanical system, when the system needs to actuate joints in real-time to maintain the balance (e.g., an inverted pendulum on a moving base), a major portion of control effort may have to be consumed just to balance the system. In such a *reactive balancing* case, even though balancing has an utmost importance, balancing mechanism cannot do any preparation. Whenever an imbalance occurs, however, everything else should be sacrificed to keep the balance.

On the other hand, when the goal is to produce "balanced walking animation", the pattern of walking motion gets a relatively more importance than in the inverted pendulum case. Balancing is just one aspect of the complicated motion. Moreover, if the computation does not need to be done in online, we can analyze the motion to identify unbalanced periods, and adjust the original motion so that the new motion is balanced. Differently from the inverted pendulum, motion change can be kept minimum so that the original motion pattern is preserved, which is possible by employing global optimization. Since it looks at the whole period, imbalances can be *avoided* before they occur. Sometimes, it might be more effective to modify even balanced parts of the motion when we consider the whole motion. We call this type of balancing *planned balancing*.

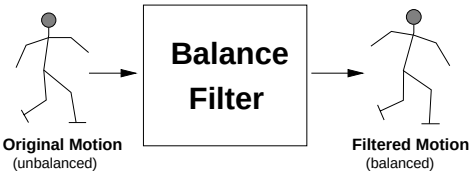


Figure 1: *Motion balance filter.*

In this paper, we propose an algorithm called *motion balance filtering*, which achieves planned balancing of a given motion. The input to the filter (Figure 1) is a motion sequence that is not necessarily dynamically balanced. The balance filter analyzes the motion to identify the unbalanced parts, and corrects the motion to produce a balanced version of it as the output. To keep the motion details of the original motion as much as possible, the process is formulated as a constrained optimization problem. Therefore motion balance filtering is an off-line algorithm.

In addition to the above basic functions, our solution to the planned balancing problem has the following interesting features:

© The Eurographics Association and Blackwell Publishers 2000. Published by Blackwell Publishers, 108 Cowley Road, Oxford OX4 1JF, UK and 350 Main Street, Malden, MA 02148, USA.

- **can handle different support phases uniformly:**
  Human motion can be decomposed into single support and double support phases. Instead of solving each phase separately, we present a formulation that handles both support phases in a uniform manner. Therefore there is no discontinuity at the phase boundaries.

- **can show different balancing styles depending on the size of foot sole:**
  The size of foot sole affects balancing. A person with a large sole can balance more easily and comfortably than a person with a smaller sole or in a pinpoint-shoes. Our algorithm can generate the differences easily by simply changing the foot sole sizes in the simulation. The result is shown in Section 5.

- **can show personified balancing style:**
  Even between two humans with same physical condition, there can exist differences in their balancing manners. Some may prefer to use the waist joint, while others may prefer to use torso joints. Our algorithm allows animators to control these preferences by setting different joint stiffnesses.

- **can add extra kinematic constraints:**
  Since our balancing algorithm is formulated as a constrained optimization problem, a kinematic constraint can be easily added. For example, a limbo walk can be originated from a normal walk by constraining the head below the limbo bar.

The remainder of this paper is organized as follows. In Section 2 we explain the ZMP concept and its relation to balance. In Section 3 we review previous work on balancing. In Section 4 we explain the steps for our balance filtering algorithm. We show the experimental results in Section 5. Finally, Section 6 concludes the paper.

2. Zero Moment Point (ZMP)

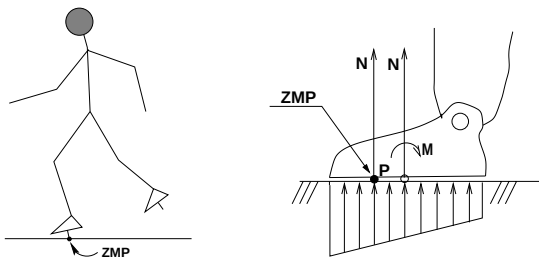


Figure 2: *(a) ZMP in a zero foot sole. (b) ZMP in a non-zero foot sole.*

The *zero moment point* (ZMP) is the key concept to understand our balancing algorithm. In robotics the concept has already been used to balance biped robots [24, 17]. ZMP used in dynamic motion analysis is analogous to the *center of gravity* (COG, the projection of the center of mass on the ground plane) used in statics. As COG should be kept within the supporting area for static balance, ZMP should be kept within the supporting area for dynamic balance [24, 13].

Balancing of a human in motion does not result from the fact that the foot soles have non-zero areas, but it essentially results from the complicated coordination of the body parts. People can walk in special shoes with pinpoint supports as illustrated in Figure 2(a). Clearly, no actuation – when the supporting point is modeled as a hinge joint – can be exerted at the supporting point. So, the point is called *zero moment point* [24].

ZMP can be extended to non-zero foot soles. Even though the distribution of ground reaction force (pressure) under the supporting foot is quite complicated, in general, it can be replaced with an equivalent pair of force N and moment M that act on a single point P within the foot sole (Figure 2(b)), as far as the dynamic motion of the whole mechanical system is concerned. Here M depends on the choice of P. We define the ZMP as P such that M vanishes to zero.

ZMP can be even further extended to the double support phase. In this case the *support area* – a generalization of the (single) supporting foot – is the convex hull of two supporting feet. The distribution of ground reaction force is a little peculiar; the pressure is zero everywhere except for the two sole area.

As the foot pressure changes during a motion, ZMP draws a certain trajectory. For example, in a walking step, ZMP translates from the heel to the tiptoe of the supporting foot [24]. For a motion to be physically valid, ZMP should be within the support area. In the pinpoint shoe example, if the tip turns out not the ZMP, that means there should have been an extra moment actuating at the tip to realize the motion, which is physically impossible.

We define balance in terms of ZMP: *a given motion is balanced when and only when the ZMP trajectory is kept within the support area.* The definition applies to both single and double support phase. Note that this definition is purely based on physical validness, and works only for the planned balancing.

In reactive balancing, all motions are happening in the real world, and thus are physically valid. Therefore the above definition for the planned balancing cannot be used. Instead, the balance should be based on whether the system actually falls or not. Under this definition, we cannot conclude a motion is unbalanced even if the system apparently looks staggering unless it eventually falls.

In dealing with kinematic motions given in the graphical world, therefore, the balance based on ZMP seems to make more sense. Normally the initial motion given to our system never falls unless the falling is the part of the motion. So

© The Eurographics Association and Blackwell Publishers 2000.

the balance from the reactive balancing does not help much. On the other hand, the ZMP-based balance can tell us which portion of the motion is out of balance, and even give a vector quantity that tells us how much and in which direction the motion should be adjusted.

Note that *balancing*, an effort to keep the balance during a motion, has a different meaning in this paper. In this paper, balancing means modifying an original motion into a balanced version of it. It is done by correcting the given motion so that the whole ZMP trajectory goes within the supporting area.

3. Previous Work

In this section we review the approaches developed for balancing biped robots or animated figures. We also review motion editing techniques, since our planned balancing is a sort of motion editing: in converting a given motion into a balanced one, it is important to preserve original motion features.

3.1. Biped Robot Control in Robotics

Many robotics researchers [20, 15, 22, 19, 10, 17, 9, 8] have studied dynamics and motion control of biped walking robots. They were mainly concerned with realizing robust locomotion considering the reactive force at the foothold. ZMP has been widely used as an index of stability. Most work took the approach to maintain ZMP within the stable region by compensating upper body motion [22, 10, 17, 8].

Fukuda et al. [10] obtained ZMP trajectory by placing four sensors at each sole. Then, the joint motion was determined using the recurrent neural networks so that ZMP should not move out of the supporting area of the robot.

Park et al. [17] presented ZMP trajectory control scheme for stable biped locomotion. The trajectory was determined by employing fuzzy logic on the leg trajectories. Based on the resulting trajectory, the trunk and swing leg motion were compensated to stabilize the locomotion. They showed that a moving ZMP (rather than a fixed ZMP) increases the locomotion stability, and guarantees natural motion.

Dasgupta and Nakamura [8] proposed an algorithm to produce feasible walking motion of a robot from human motion capture data. First, they heuristically set a desired ZMP trajectory referring to the data, and then corrected the motion of a selected joint so that the resulting motion approximately matches the desired ZMP trajectory. Assuming the joint motion is represented in Fourier series, they formulated an optimization problem to determine the unknown coefficients.

The robots studied by the above groups had simple structures (typically of 7 or 8 DOFs). The resulting motion was slow and quite primitive compared to human motion. Their work mainly focused on developing a stable control algorithm.

3.2. Balance Control in Computer Animation

Phillips and Badler [18] described techniques for interactively controlling bipedal figures using kinematic constraints. They associated the COM logically with lower torso region of the figure, and used it as the end-effector of the constraint. In realtime interaction, they maintained the COM within supporting polygon for static balance.

Boulic et al. [4] proposed an algorithm to control COM for any tree-structured articulated figure in a multiple support context. By distributing the body mass in a proper manner, realistic static postures could be obtained. They considered the COM as an end-effector, and controlled it using inverse kinematics. They named the approach *inverse kinetics*.

Recently, Aydin et al. [2] proposed a balance control algorithm in interactive environments. Their approach controlled the root joint (a similar concept as ZMP) in order to produce balanced posture when the environmental interaction is not restricted to only gravitational force or ground reaction. Assuming that the inertia effects are negligible and external forces/torques are known, they computed the torque of the root joint during the *forward torque control phase*. Then they calculated the joint rotation angles that reduce the torque at the root joint to zero during the *inverse torque control phase*.

The above approaches are valid only when there is no motion or the motion is very slow. In general, when a kinematically generated motion needs to be balanced, clearly dynamic balance has to be considered.

Ko [13, 14] proposed a balance control technique for human locomotion. He introduced the *balance vector*, the torque at the ZMP computed by inverse dynamics, as a degree of imbalance. Then balancing was achieved by translating or rotating the pelvis and torso so that the torque at the ZMP vanishes to zero. His algorithm was effective in keeping the balance of only a specific type of motion – human walking.

3.3. Motion Editing

There have been proposed several approaches to reuse or edit existing motion library. Bruderlin et al. regarded a motion as time-varying signals, and applied many signal processing techniques such as multi-resolution filtering, dynamic time-warping, motion displacement mapping [5]. Witkin et al. introduced the motion warping technique [26]. They directly manipulated the motion curve in the time domain. Unuma et al. [23] edited the motion in the frequency domain, and could generate locomotion with different emotions.

Gleicher [11, 12] proposed a motion editing algorithm to retarget a captured motion to human figures of different anthropometric scales. Lee and Shin [16] enhanced Gleicher's work by employing a hierarchical curve fitting technique and human-specific inverse kinematics solver. Recently, Choi and Ko developed the *on-line motion retargeting* technique [6]

© The Eurographics Association and Blackwell Publishers 2000.

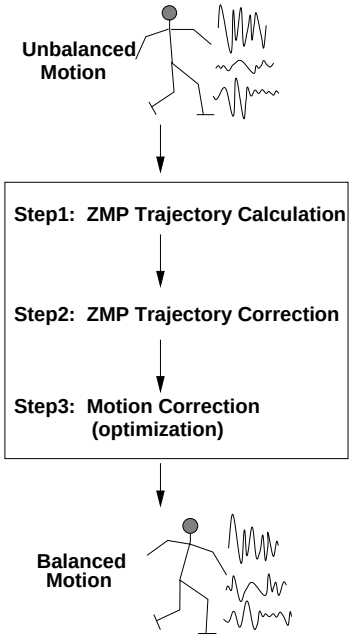


Figure 3: *Steps for balance filtering.*

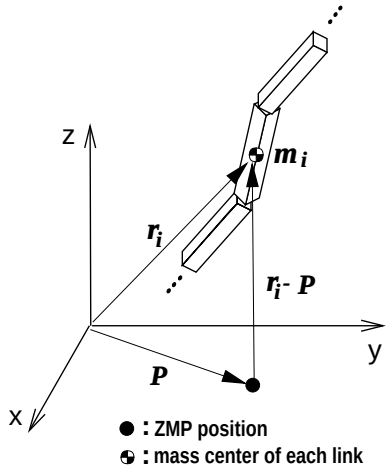


Figure 4: *ZMP calculation.*

which enables captured motion to be retargeted in real-time to a character while the motion is being captured.

The above techniques are based purely on kinematics. They do not consider the physical aspects such as balance.

4. Motion Balance Filtering

Figure 3 shows the motion balance filtering process which consists of three steps. In Step 1, we analyze the motion data to obtain the ZMP trajectory of the given motion. In Step 2, if there are some portions of ZMP trajectory that go outside the supporting area, they are projected within the area. In Step 3, we modify the motion so that the ZMP trajectory of the new motion coincides with the one obtained in Step 2. Those steps are further explained below.

4.1. Spline Fitting of Joint Angle Data

Inverse dynamics, which is the main procedure for computing the ZMP trajectory in Step 1, needs to have the joint angle accelerations. When the (discrete) raw joint angle data is used in calculating the acceleration, the result is quite sensitive to the noise. In the original data the noise is nearly unnoticeable since its amplitude is very small. When the angle is doubly differentiated, however, the amplitude due to noise increases proportionally to the square of its frequency. Filtering the raw data with an appropriate low-pass filter is a common solution to this problem [25].

In this paper, we solve the noise problem by fitting the motion data into a cubic B-spline [21]. For this, we employ least squares spline approximation technique with MATLAB spline toolbox [1]. When the raw motion data is given along with the knot sequence, a spline curve is determined in a way that minimizes the mean-squares. Once the motion is represented in an analytic form, we can differentiate the equation twice and obtain the second derivatives. At this time the noise problem is mostly removed.

4.2. ZMP Trajectory Calculation

Computing ZMP according to the original definition given in Section 2 is not practical, since it is not easy to measure the distribution of ground reaction force over time. Fortunately, there is another way to compute ZMP that uses inverse dynamics. In the motion of an articulated figure, each body segment contributes to the moment at the foot. According to *D'Alambert's principle* [3], the moment contribution at point **P** (Figure 4) is given by $m_i(\mathbf{r}_i - \mathbf{P}) \times (-\ddot{\mathbf{r}}_i + \mathbf{g})$, where m_i is the mass of the i-th link, $\mathbf{r}_i$ is the position of the mass center of i-th link. Therefore ZMP can be obtained by solving the following equation for **P**:

$$\sum_i m_i(\mathbf{r}_i - \mathbf{P}) \times (-\ddot{\mathbf{r}}_i + \mathbf{g}) = 0. \tag{1}$$

There is an infinite number of solutions for Equation (1). (The dimension of the solution space is one.) We can obtain a unique solution by fixing z value of **P** to zero, which corresponds to constraining **P** into the ground plane. The solution $\mathbf{P} = (x_{zmp}, y_{zmp}, 0)$ is given by

$$x_{zmp} = \frac{\sum_i m_i(\ddot{z}_i - g_z)x_i - \sum_i m_i(\ddot{x}_i - g_x)z_i}{\sum_i m_i(\ddot{z}_i - g_z)} \tag{2}$$

$$y_{zmp} = \frac{\sum_i m_i(\ddot{z}_i - g_z)y_i - \sum_i m_i(\ddot{y}_i - g_y)z_i}{\sum_i m_i(\ddot{z}_i - g_z)}, \tag{3}$$

© The Eurographics Association and Blackwell Publishers 2000.

where (x_i, y_i, z_i) and $(\ddot{x}_i, \ddot{y}_i, \ddot{z}_i)$ are the position and acceleration of the mass center of i-th link. Calculating Equations (2) and (3) at each time tick produces ZMP trajectory.

4.3. ZMP Trajectory Correction

If the ZMP trajectory obtained in Section 4.2 goes out of the supporting area, it means the motion is not dynamically balanced, thus the motion has to be modified. But first we have to modify the ZMP trajectory so that it goes in the supporting area.

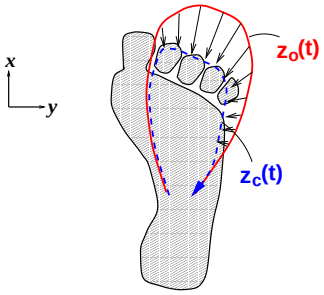


Figure 5: *Projecting ZMP trajectory into the support area: the solid and dashed lines represent the original ZMP trajectory $\mathbf{z}_o(t)$ and corrected ZMP trajectory $\mathbf{z}_c(t)$, respectively.*

We obtain the workable ZMP trajectory $\mathbf{z}_c$ in Figure 5 by modifying the original trajectory $\mathbf{z}_o$ *minimally*; we simply project the ZMP trajectory portions lying outside the supporting area to the boundary as shown in the figure. Although the method seems reasonable, the resulting motion may be objectionable from the point of stability. (In robotics, people control the system so that ZMP trajectory is placed well inside the supporting area for stability.) If a more stable motion is needed, we project ZMP trajectory to the inner side of the area.

4.4. Motion Correction

In this section, we address the problem of modifying original motion so that ZMP trajectory of the new motion coincides with $\mathbf{z}_c$. Due to the high degrees of freedom in the body, there exist an infinite number of solutions. The redundancy is exploited in our motion correction step for preserving the original motion pattern. It is formulated as a constrained optimization problem:

$$\textit{minimize}\ g(\theta) \quad \textit{subject to}\ \mathbf{f}(\theta) = \mathbf{c}, \tag{4}$$

where θ is a vector representing the joint angles, g is the objective function, $\mathbf{f}$ is a vector function of constraints, and $\mathbf{c}$ the given constraint vector.

In our balance filtering, the ZMP constraint obtained in

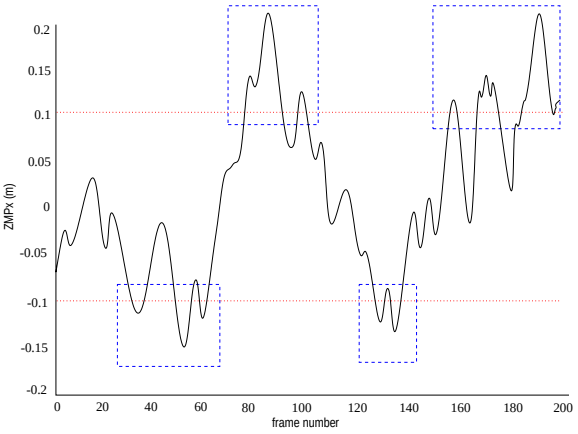


Figure 6: *ZMP trajectory in a sway motion: the two dotted lines represent support area boundaries. The regions enclosed by rectangles represent optimization windows.*

Section 4.3 is used to set up $\mathbf{f}(\theta) = \mathbf{c}$. For the objective function, the following equation[12], can be used:

$$g(\theta) = \int (\theta(t) - \theta_o(t))^2\, dt, \tag{5}$$

where $\theta(t)$ and $\theta_o(t)$ are the joint angle vectors of the new and original motions, respectively.

In conventional spacetime constraint methods, entire motion is considered in the objective function, and thus entire motion is modified. In our balance filtering, however, it seems more reasonable to modify only the unbalanced part, keeping the rest unchanged. For this, we adopt the selective spacetime window technique[7]. Figure 6 shows the trajectory of ZMP relative to the support area boundaries. We activate our balancing algorithm only for the parts where the trajectory is outside of the boundary. In the figure, the parts processed by the algorithm is indicated with rectangular boxes. In *planned* balancing, it turns out more effective to form the boxes to include some of the balanced parts around as shown in Figure 6. It enables balancing to be done with less amount of effort by *preparing earlier*. In our implementation, we used optimization windows with 5 or 6 extra frames before and after the unbalanced part.

Actual objective function we used in the motion correction step was the discrete form

$$g(\theta) = \sum \|\theta(t) - \theta_o(t)\|_M, \tag{6}$$

$$M = \begin{bmatrix} m_{11} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & m_{JJ} \end{bmatrix}, \tag{7}$$

where $\|\cdot\|_M$ is the matrix M-norm, which is defined by

© The Eurographics Association and Blackwell Publishers 2000.

$\|\mathbf{x}\|_M = \mathbf{x}^t\mathbf{M}\mathbf{x}$ for a given vector $\mathbf{x} \in \mathbb{R}^J$. Here $\mathbf{x}$ has the dimension of joint angle space.

The matrix $\mathbf{M}$ in Equation (7) is a positive-definite matrix, which is typically diagonal if the coupling factors among joints do not exist. The matrix value is set by animators to control the stiffness at the joints. For example, a character can be made to use a particular joint j more actively than other joints by setting m_{jj} with a smaller value. It turns out that letting animators control the joint stiffness is quite helpful for making the final result look more human-like motion. After surveying the original motion, animators can control more specific aspects of the new motion by setting different values for $\mathbf{M}$, without destroying the dynamic soundness.

4.4.1. Imposing Additional Constraints

In many human motions, the constrained optimization above produces an unacceptable result. If the character is doing the limbo game, the head should be maintained below the limbo bar. If a character is dancing, the foot must be on the floor when it is in the support phase. Optimization under simple ZMP constraints cannot guarantee such conditions. In our algorithm such conditions can be easily accommodated by simply augmenting the constraint part with an appropriate set of additional constraints. The experimental results will be shown in Section 5.

4.4.2. Handling Double Support Phases

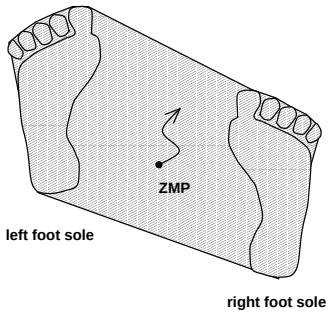


Figure 7: *Supporting area in double support phase.*

In the double support phase, everything is basically the same except that the supporting area is a lot wider. In Step 1, ZMP trajectory is calculated with the same ZMP equations, i.e., Equations (2) and (3). In Step 2, the supporting area becomes the convex hull of two feet areas as shown in Figure 7, thus the ZMP trajectory is projected into a larger area. In Step 3, to maintain the double support, we simply impose a constraint that two feet should be positioned at the fixed locations during the double support phase.

5. Experiments

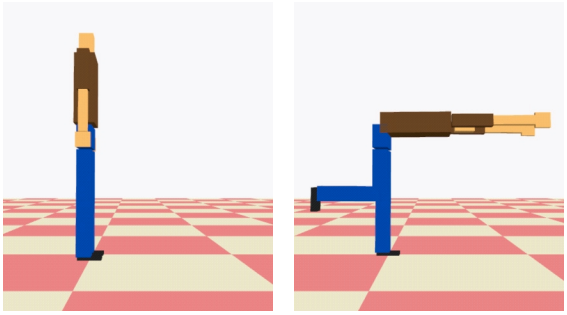
In this section we report the experimental results of our motion balance filtering algorithm that is implemented on Silicon Graphics Octane MXI workstation. We used an articulated human figure of 25 DOFs. The length and mass of each link, which are listed in Table 1, were taken from Winter's book [25].

link	length (cm)	mass (kg)
pelvis	15.5	7.1
torso	48.0	17.75
head	14.0	4.05
upper arm	27.0	1.4
lower arm	28.0	1.1
upper leg	30.0	5.0
lower leg	42.0	2.325
foot	16.0	0.725

Table 1: *Physical properties of human model.*

We performed experiments on two motions: (1) forward stretching motion and (2) limbo walking. (1) was prepared by kinematic editing; (2) was obtained from motion capture. We ran our motion balance filtering algorithm on those motions with different parameter values. The results were recorded into animation Clips#1~#10, which are available at `http://graphics.snu.ac.kr/demo/mbf/`. All the animations were recorded at normal speed: 30 frames per second.

5.1. Forward Stretching Motion



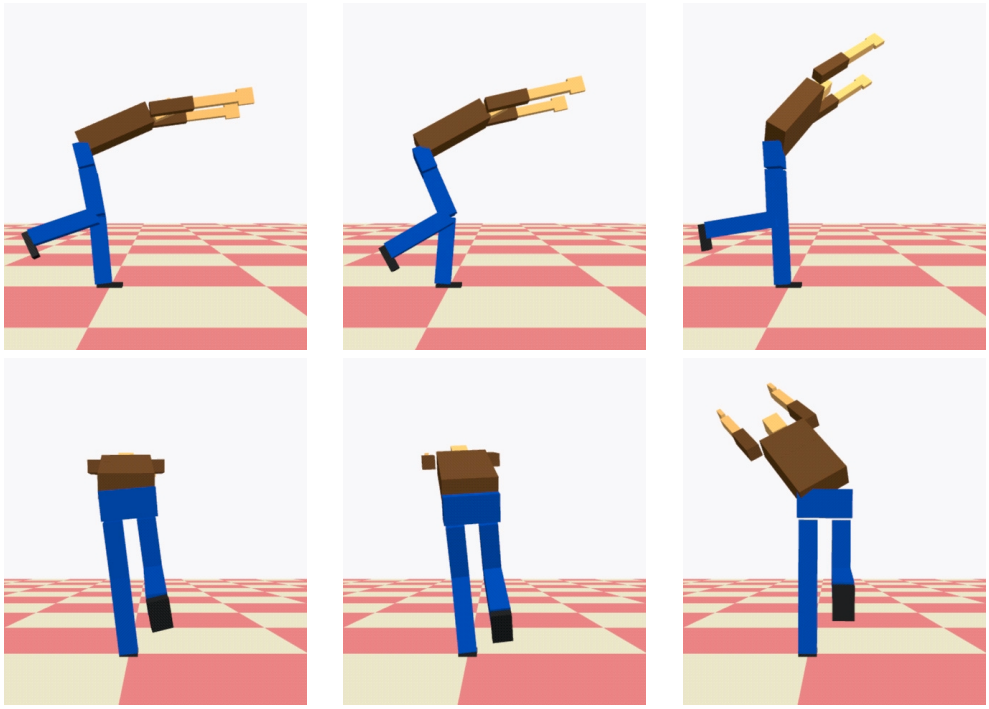
(a) First frame (b) Last frame

Figure 8: *Keyframes of forward stretching motion.*

In this experiment the original motion (the forward stretching motion, 30 frames long) was prepared from kinematic editing, which is shown in Clip#1. We simply interpolated two keyframes, from the upright-standing posture (Figure 8(a)) to the posture shown in Figure 8(b). Although the motion looks O.K., it is not possible in real world. The character can not make such a stop, since both ZMP and COG are outside of the supporting area.

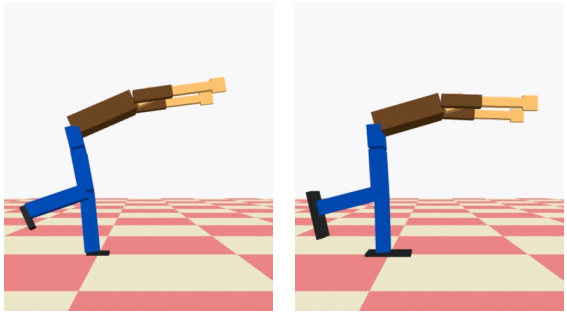
We followed the three steps in Section 4. i.e., we solved

© The Eurographics Association and Blackwell Publishers 2000.



(a) With uniform stiffness (b) With small knee stiffness (c) With small waist stiffness

Figure 9: *Snapshots from forward stretching motion.*



(a) With normal foot sole (b) With large foot sole

Figure 10: *Different foot sole size.*

the constrained optimization problem under the ZMP constraint. The result was, however, quite different from what we expected. It produced a (natural) falling motion shown in Clip#2. It was because there was no other constraint than the ZMP constraint; the motion actually satisfied the given ZMP constraint.

So, we added another constraint that all the link velocities should be zero at the first and last frames. It means that the character should be in a static balance at the last frame. At this time we got a reasonable result which is shown in Clip#3.

In the previous experiment (Clip#3), an identity matrix was used for **M**. i.e., the stiffness was uniform over all joints. By controlling the stiffness values, we got the results shown in Clips#4 and #5. Clip#4 was obtained with a small knee stiffness, and Clip#5 was obtained with a small waist stiffness. Figure 9 compares the effects due to the stiffness matrix. The snapshots were taken at Frame 28.

All the results shown in Clips#3~#5 were obtained with a normal foot size. In Clips#7, stretching was done with a large foot. Figure 10 compares the snapshots taken at Frame 28.

5.2. Limbo Walking

It is very difficult to generate a plausible looking limbo walking when only kinematic techniques are employed, since the dynamic balance cannot be taken care of properly. For example, simply bending the waist joint backward (shown in Figure 11(a) and in Clip#9) produces an unacceptable result. In this experiment, we show limbo walking can be easily produced by our algorithm.

© The Eurographics Association and Blackwell Publishers 2000.

(a) From kinematic editing (b) From balance filtering (c) From motion capture

Figure 11: *Comparison of limbo walking generated from different algorithms.*

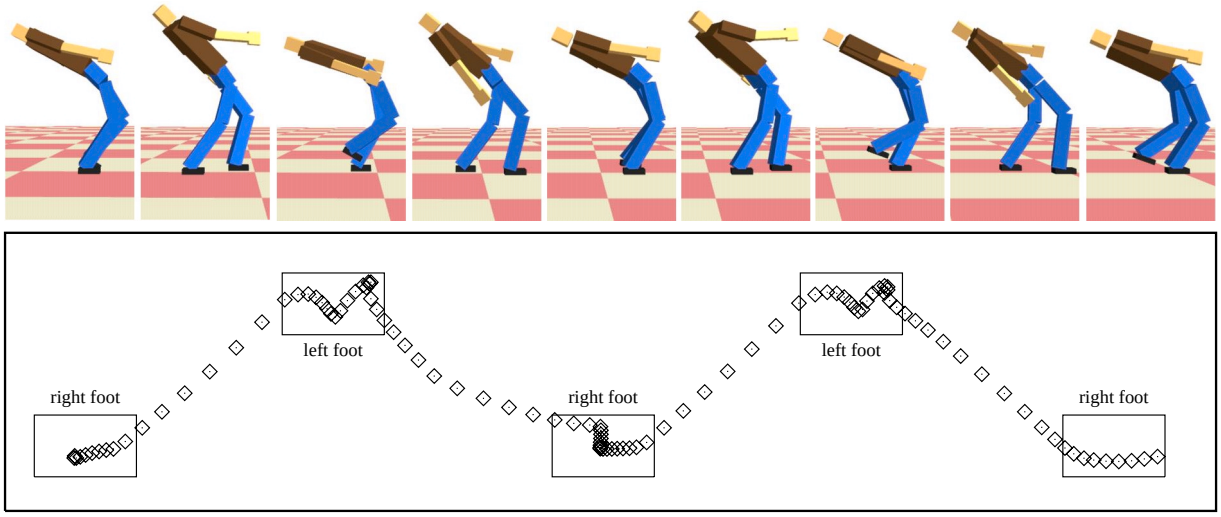


Figure 12: *Snapshots and ZMP trajectory taken from a limbo walking.*

We used a normal walking motion (shown in Clip#8) as the original motion, which was captured using a magnetic equipment. The height of the head was constrained so that the character can go under the limbo bar placed at 110*cm* above the ground. Also, the waist was constrained so that the character does not bend forward. The resulting motion is shown in Clip#10. Comparing with the result in Clip#9 we can see the dynamic balance is quite well taken care of.

The ZMP trajectory was traced while our algorithm was generating the four limbo steps. The result is shown in Figure 12 along with nine snapshots. The 1st, 3rd, 5th, 7th, and 9th snapshots are from single support phases; 2nd, 4th, 6th, and 8th snapshots are from double support phases. During single support phases, the ZMP trajectory stays well within the rectangles that represent the footprints. During double support phases, we can observe from the figure that the ZMP trajectory stays well within the support area – the convex hull of two footprints.

To see how our result compares with real limbo walking, we captured a real limbo walking. Figure 11 shows the three snapshots taken from the kinematically generated motion, the result of our algorithm, and the real limbo walking, respectively.

6. Conclusion

Balancing is one of the most important dynamic aspects of human motion. Without it, the motion does not look happening in real world. In this paper, we presented an algorithm, motion balance filtering, that can be used to *filter* a given motion so that the new version is dynamically balanced. We have shown the results of several experiments to demonstrate the usefulness of the algorithm.

© The Eurographics Association and Blackwell Publishers 2000.

We should make a note that our algorithm achieves *planned balancing*. It can produce an unnatural result if it is applied to a wrong situation. For example, when random forces act on a character in an *ad hoc* manner, according to the algorithm, the character will balance as if he knows all the forces beforehand. (In such a case reactive balancing has to be considered.) Our algorithm can produce good results when we have to deal with the motion that is quite accustomed to people through long experience.

Even though a human figure was used throughout the paper, our algorithm is applicable to any articulated figures. Therefore it can be easily integrated into any existing animation systems as a post processing unit to add physical realism to a kinematically edited motion.

Acknowledgment

This paper was supported by Creative Research Initiatives of the Korean Ministry of Science and Technology. This work was also partially supported by ASRI (Automation and Systems Research Institute), Seoul National University, and the Brain Korea 21 Project.

References

1. *MATLAB Spline Toolbox, version 2.0*. The Mathworks, Inc., 1997. 4
2. Yahya Aydin and Masayuki Nakajima. Balance control and mass centre adjustment of articulated figures in interactive environments. *The Visual Computer*, 15:113–123, 1999. 3
3. Ferdinand P. Beer and E. Russell Johnston Jr. *Vector Mechanics for Engineers - Dynamics*. McGraw-Hill, second edition, 1990. 4
4. Ronan Boulic, Ramon Mas, and Daniel Thalmann. Position control of the center of mass for articulated figures in multiple support. In *6th Eurographics Workshop on Animation and Simulation*, pages 130–143. Eurographics, September 1995. 3
5. Armin Bruderlin and Lance Williams. Motion signal processing. In *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 97–104, August 1995. 3
6. Kwangjin Choi and Hyeongseok Ko. On-line motion retargetting. In Bob Werner, editor, *Pacific Graphics '99 Proceedings*, pages 32–42, October 1999. 3
7. Michael F. Cohen. Interactive spacetime constraint for animation. In *Computer Graphics (SIGGRAPH '92 Proceedings)*, pages 293–302, July 1992. 5
8. A. Dasgupta and Y. Nakamura. Making feasible walking motion of humanoid robots from human motion capture data. In *Robotics and Automation '99 Proceedings*, volume 2, pages 1044–1049, 1999. 3
9. Y. Fujimoto and A. Kawamura. Simulation of an autonomous biped walking robot including environmental force interaction. *IEEE Robotics and Automation Magazine*, pages 33–42, 1998. 3
10. T. Fukuda, Y. Komata, and T. Arakawa. Stabilization control of biped locomotion robot based learning with gas having self-adaptive mutation and recurrent neural networks. In *Robotics and Automation '97 Proceedings*, volume 1, pages 217–222, 1997. 3
11. Michael Gleicher. Motion editing with spacetime constraints. In *Proceedings of the 1997 Symposium on Interactive 3D Graphics*, pages 139–148, 1997. 3
12. Michael Gleicher. Retargetting motion to new characters. In *Computer Graphics (SIGGRAPH '98 Proceedings)*, pages 33–42, July 1998. 3, 5
13. Hyeongseok Ko. *Kinematic and Dynamic Techniques for Analyzing, Predicting, and Animating Human Locomotion*. PhD thesis, University of Pennsylvania, Department of Computer and Information Science, Philadelphia, PA 19104-6389, May 1994. MS-CIS-94-31. 2, 3
14. Hyeongseok Ko and Norman I. Badler. Animating human locomotion in real-time using inverse dynamics, balance and comfort control. *IEEE Computer Graphics and Applications*, 16(2):50–59, March 1996. 3
15. Andrew L. Kun and W. Thomas Miller. Adaptive dynamic balance of an experimental biped robot. In *1996 International Conference on Robotics and Automation*, 1996. 3
16. Jehee Lee and Sung Young Shin. A hierarchical approach to interactive motion editing for human-like figures. In *SIGGRAPH 99 Conference Proceedings*, Annual Conference Series. ACM SIGGRAPH, Addison Wesley, August 1999. 3
17. Jong H. Park and Yong K. Rhee. Zmp trajectory generation for reduced trunk motions of biped robots. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'98)*, pages 90–95. IEEE/RSJ IROS, October 1998. 2, 3
18. Cary B. Phillips and Norman I. Badler. Interactive behaviors for bipedal articulated figures. In *Computer Graphics (SIGGRAPH '91 Proceedings)*, pages 359–362, July 1991. 3
19. Jerry Pratt, Peter Dilworth, and Gill Pratt. Virtual model control of a bipedal walking robot. In *1997 International Conference on Robotics and Automation*, 1997. 3
20. Marc H. Raibert. *Legged Robots that Balance*. MIT Press, 1986. 3
21. David F. Rogers and J. Alan Adams. *Mathematical El-*

© The Eurographics Association and Blackwell Publishers 2000.

ements for Computer Graphics. McGraw-Hill, international edition, 1990. 4

22. Ching-Long Shih. Analysis of the dynamics of a biped robot with seven degrees of freedom. In *Robotics and Automation '96 Proceedings*, volume 4, pages 3088–3013, 1996. 3

23. Munetoshi Unuma, Ken Anjyo, and Ryozo Takeuchi. Fourier principles for emotion-based human figure animation. In *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 91–96, August 1995. ACM-0-89791-701-4. 3

24. M. Vukobratović. *Biped Locomotion*. Scientific Fundamentals of Robotics 7, Communications and Control Engineering Series. Springer-Verlag, Berlin, New York, 1990. 2

25. David A. Winter. *Biomechanics and Motor Control of Human Movement*. Wiley, New York, second edition, 1990. 4, 6

26. Andrew Witkin and Zoran Popovic. Motion warping. In *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 105–107, August 1995. 3

© The Eurographics Association and Blackwell Publishers 2000.