University of Pennsylvania
ScholarlyCommons

Technical Reports (CIS) Department of Computer & Information Science

January 1993

Curved Path Human Locomotion That Handles Anthropometrical Variety

Hyeongseok Ko
University of Pennsylvania

Norman I. Badler
University of Pennsylvania, badler@seas.upenn.edu

Follow this and additional works at: https://repository.upenn.edu/cis_reports

Recommended Citation

Hyeongseok Ko and Norman I. Badler, "Curved Path Human Locomotion That Handles Anthropometrical Variety", . January 1993.

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-93-13.

This paper is posted at ScholarlyCommons. https://repository.upenn.edu/cis_reports/464
For more information, please contact repository@pobox.upenn.edu.

Curved Path Human Locomotion That Handles Anthropometrical Variety

Abstract

Human locomotion simulation along a curved path is presented. The process adds a small constant cost (*O*(1)) to any pre-existing straight line walking algorithm. The input curve is processed by the *foot print generator* to produce a foot print sequence. The resulting sequence is scanned by the *walking motion generator* that actually generates the poses of the walking that realizes such foot prints. The two primitives INITIALIZE_STEP and ADVANCE_STEP are used for walking motion generation. INITIALIZE_STEP is activated with the input parameters *walker, next_foot_print, left_or_right*, and *step_duration*, just before each step to precompute the trajectories of the center of the body and the ankles. ADVANCE_STEP is called with a *normalized time* to generate the actual pose at that moment. The normalized time is a logical time, covering zero to one during a complete step.

Keywords

animation, human locomotion, walking path, foot prints, algorithm, generalization, trajectory, time complexity

Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-93-13.

This technical report is available at ScholarlyCommons: https://repository.upenn.edu/cis_reports/464

Curved Path Human Locomotion That Handles Anthropometrical Variety

MS-CIS-93-13
GRAPHICS LAB 54

Hyeongseok Ko
Norman I. Badler

University of Pennsylvania
School of Engineering and Applied Science
Computer and Information Science Department
Philadelphia, PA 19104-6389

February 1993

Curved Path Human Locomotion that Handles Anthropometrical Variety

Hyeongseok Ko, Norman I. Badler
Department of Computer and Information Science
University of Pennsylvania
200 South 33rd Street
Philadelphia, PA 19104-6389
e-mail : ko@graphics.cis.upenn.edu,
badler@central.cis.upenn.edu
Phone : (215) 898-1976
Fax : (215) 898-0587

January 1993

Abstract

Human locomotion simulation along a curved path is presented. The process adds a small constant cost ($O(1)$) to any pre-existing straight line walking algorithm. The input curve is processed by the *foot print generator* to produce a foot print sequence. The resulting sequence is scanned by the *walking motion generator* that actually generates the poses of the walking that realizes such foot prints. The two primitives **INITIALISE_STEP** and **ADVANCE_STEP** are used for walking motion generation. **INITIALIZE_STEP** is activated with the input parameters *walker*, *next_foot_print*, *left_or_right*, and *step_duration*, just before each step to pre-compute the trajectories of the center of the body and the ankles. **ADVANCE_STEP** is called with a *normalized time* to generate the actual pose at that moment. The normalized time is a logical time, covering zero to one during a complete step.

Keywords: animation, human locomotion, walking path, foot prints, algorithm, generalization, trajectory, time complexity.

1 Introduction

In dealing with human behaviors in a three dimensional animation, the curved path locomotion problem naturally arises since the walking path is usually a curve rather than a straight line. A curved path locomotion system can be very useful to human animators if it can automatically generate the walking motion of human figures from any position and direction to any other position and direction.

There have been many research groups interested in human locomotion [9, 16, 15, 14, 22, 21, 20, 10, 19]. However, most of them have focused on *linear path locomotion* (LPL) in which walking is restricted to the sagittal plane. There have been studies on other kinds of legged locomotion such as hopping, jumping, and multi-leg coordination [18, 12, 7].

Bruderlin and Calvert built a keyframeless locomotion system for straight walking paths [3, 2]. They generated every single frame based on both dynamics and kinematics. Walking was controlled by three primary parameters: step length, step frequency, and velocity. Various walking styles could be produced by changing the walking attributes.

Boulic *et al.* tried a generalization of experimental data based on the normalized velocity of walking [1]. They put a correction phase (inverse kinematics) to handle the possible constraint violation of the computed values. In that process they introduced the *coach concept*, which basically chooses among the multiple inverse kinematic solutions one that is the closest to the original motion.

Another straight line walking animation technique was developed by Ko and Badler [11]. In this approach, steps of arbitrarily anthropometrically scaled human figures ([8]) at an arbitrary step length are obtained through a generalization of the measured data of one particular subject and step length. The constraints are enforced within the generalization process, obviating the correction phase. Also the original walking style is maintained during the generalization. By acquiring a multiple set of measurements, several walking styles can be shown in one animation scene.

Girard discussed the turning problem in [6]. He interprets the stepping (*liftoff*) as a way to exert an impulse on the human body in running. Each impulse contributes an acceleration to the whole body movement. He computed the impulse that is required to drive the center of the body along a given curve. By the impulse (which includes rotational torque as well as upward force) at the liftoff, the body gets a rotational torque and the *whole* body rotates in the air.

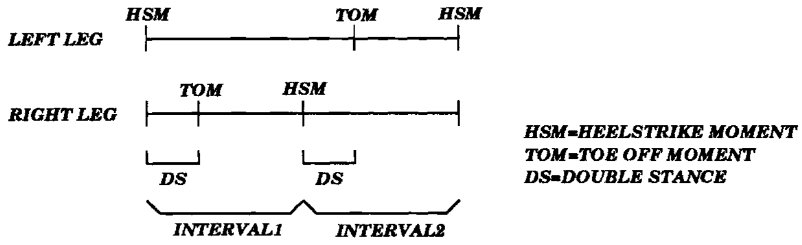

Figure 1: The Phase Diagram of Human Walk

In walking, however, everything that turning entails is done while at least one of the feet is on the ground. So there are more constraints to be satisfied than in running. The whole body does not turn rigidly at the same angle: the body is rather *twisted* during the motion. Moreover, the ankle and hip joints have an important role in generating a natural motion.

In building a curved path locomotion (CPL) system, we tried to utilize pre-existing LPL systems. That is, an LPL system is used as a subsystem to our CPL system. Our generalization algorithm from LPL to CPL was based on the intuition that there should be a smooth transition between linear and curved path locomotion: if the curvature is not large, the curved path walking generated by our CPL system should be close to the linear path walking given by the underlying LPL system. In particular, if the given curve is actually a straight line, the resulting CPL should match that of the underlying LPL system. *No assumptions were made about the underlying LPL system, therefore most LPL systems can be generalized into CPL ones by our algorithm.* Clearly the underlying LPL will determine the stylistics of the resulting CPL.

Two assumptions are made here: A revolute joint with a *fixed axis* is assumed at the ball of the foot. This is one of the major differences between the real human foot and our model. In real walking, the axis of flexion at the ball of the stance foot changes its orientation according to the current stepping direction. Another assumption is there should be no sliding between the foot and the floor. Even though these two things actually happen in real walking, they are difficult to predict and model.

At a certain moment, if a leg is between its own heelstrike (beginning) and the other leg's heelstrike (ending), it is called the *stance leg.* If a leg is between the other leg's heelstrike (beginning) and its own heelstrike (ending), it is called the *swing leg.* For example, in Figure 1, left leg is the stance leg during interval 1, and right leg is the stance leg during interval 2. Thus at each moment

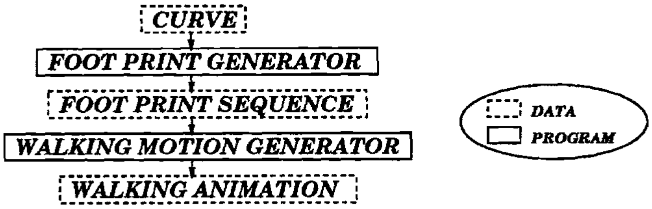

Figure 2: The Two Subproblems of the CPL System

we can refer to a specific leg by either stance or swing leg with no ambiguity. The joints and segments in a leg will be referred to using prefixes *swing* or *stance*. For example, swing ankle is the ankle in the swing leg.

2 Overview

The curved path walking problem is decomposed into the two subproblems: *foot print generation (FPG)* and *walking motion generation (WMG)* (Figure 2). Foot print generator generates a foot print sequence that follows the given curve. Walking motion generator scans the foot print sequence and produces a walking step for each foot print in the sequence. Even though the WMG is the more essential part of the CPL system, experiment shows that FPG part is also very important for realistic walking, especially under our two assumptions. FPG will be dealt with in Section 3

There are two primitives for the WMG: **INITIALIZE_STEP(walker, next_foot_print, left_or_right, step_duration)** and **ADVANCE_STEP(walker,normalized_time)**. Each step is initialized by **INITIALIZE_STEP**, which precomputes the higher level part so that **ADVANCE_STEP** can be done in $O(1)$ time later. In **INITIALIZE_STEP**, **walker** specifies the walker to be initialized for the step. This way there can be multiple walkers in the same scene. **next_foot_print** is the location of the foot print to be achieved by the current step. **left_or_right** designates which foot (and leg) is used in the current step. The duration of the step is given by **step_duration**.

ADVANCE_STEP generates the walking poses of **walker** at the given **normalized_time**. **normalized_time** is the *logical* time: **ADVANCE_STEP(walker,0.0)** gives the pose at the beginning of the current step and **ADVANCE_STEP(walker,1.0)** gives the one at the end of the current step. By increasing **normalized_time** from zero to one, the walking motion of a whole

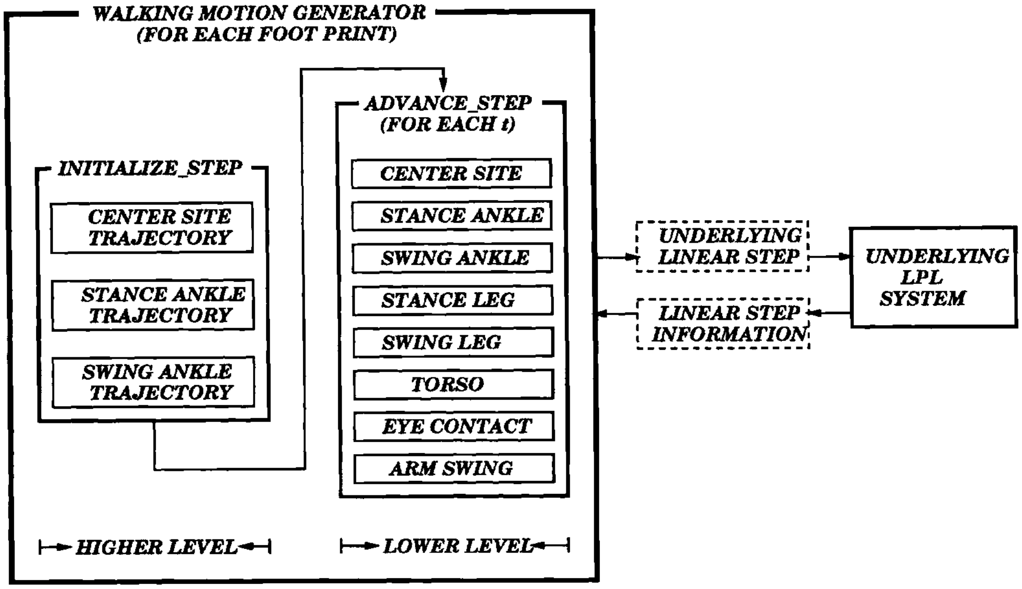

Figure 3: The Overview of the Walking Motion Generation

step can be generated. The step size Δt of **normalized_time** can be adjusted. It is a very effective way to adapt to the various machine speeds for realtime interactive walking display. The concept of normalized time proved to be intuitive and easy to use for the animators.

The overall structure of our WMG algorithm is depicted in Figure 3. The *center site* is defined as the mid-point of the two hip joints. In the movement of the lower body, the trajectories of the center site and the ankles determine the basic outline of the walking. These trajectories are precomputed within **INITIALIZE_STEP**. This higher level part of WMG will be discussed in Section 4.

Lower level details of WMG are computed in **ADVANCE_STEP** at each normalized time step t: first, the locations of the center site, the stance ankle, and the swing ankle are computed from the trajectory that has been computed in the higher level part. The center site location and its orientation determines the locations of the hips. The hip and ankle locations determine the configurations of the legs. The torso is bent to produce the required banking. The torso and the neck are twisted for an appropriate eye gaze direction. Finally the arm swing is added. The lower level details will be explained in Section 5.

Whenever necessary, the LPL system provides the information about the underlying linear step of the current (curved) step.

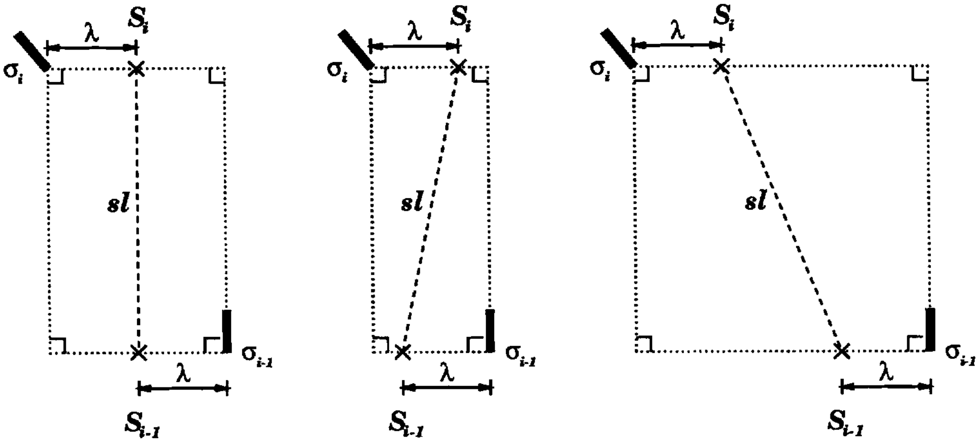

Figure 4: The Step Length of a Curved Step

3 Foot Print Generation

As discussed in the previous section, the direct input to the walking motion generator is a *foot print sequence*

$$\Sigma = \{\sigma_i \mid i = 0, \ldots, n\} \tag{1}$$

where each $\sigma_i = (\vec{footpos}_i, \vec{footdir}_i, left\_or\_right_i, duration_i)$ is called a *foot print.* $\vec{footpos}_i$ is the position of the *heel* of the foot print, and $\vec{footdir}_i$ is a unit vector representing the direction of the foot print from the heel to the tip of the toe. In some contexts, we will also call $(\vec{footpos}_i, \vec{footdir}_i)$ a foot print and denote it as σ_i. A pair of adjacent two foot prints (σ_{i-1}, σ_i) is called the i-th *step.* The algorithm that computes the foot print sequence from a given curve will be given in this section.

The step length sl of a curved step (σ_{i-1}, σ_i) is defined as the distance between the two points S_{i-1} and S_i (Figure 4) that are displaced by λ laterally inward from $\vec{footpos}_{i-1}$ and $\vec{footpos}_i$, respectively, where λ is the (constant) distance from the center site to either of the hip joints. The direction of the rectangle in the Figure is determined by $\vec{footdir}_{i-1}$. (For simplification, we assume that 2λ is the lateral step width in usual straight line walking. Actually it is narrower than 2λ in real walking [9].)

For each curved step (σ_{i-1}, σ_i), we consider its *underlying linear step.* The step duration of this underlying linear step is the same with that of σ_i. The step length of the underlying linear step is given by the step length of the curved step (σ_{i-1}, σ_i).

A curve is represented by a sequence $\Pi = \{\vec{P_i} \mid i = 0, \ldots, N_{curve\_points}\}$ of points. The number of points in the sequence depends on the curve and its density. We will denote it N_{curve_points} for later time analysis. Adjacent points in the sequence are assumed to be close enough.

The final foot print sequence is produced through the two phases. The first phase generates a set of temporary foot prints at step length μ, which is an input constant that controls the step length of the curved path walking in general. In the second phase, the temporary foot prints are slightly modified so that the resulting walking might be more realistic.

3.1 The First Phase

In σ_0 of the following algorithm, $\vec{footpos}_0$ is the position of the first stance foot, and $\vec{footdir}_0$ is the unit directional vector of the foot. A constant μ is given as the *standard* step length, which is for the steps along a linear path. The step length is reduced for the curved steps appropriately according to the curvature of the path (line 5).

Algorithm 1 (**INPUT** : *A Curve* Π, σ_0, *and* μ, **OUTPUT** : *Temporary Foot Print Sequence*)

```
• $j = 0$.

• for $(i = 1; i \leq N_{steps}; i = i + 1)$ {

        do {
    1.          $j = j + 1$.

    2.          Find the point $\vec{Q}_j$ displaced by $\lambda$ laterally outward from the point $\vec{P_j}$ in $\Pi$.
                        (Figure 5)

    3.          Compute (numerically) the normalized derivative $\vec{Q}'_j$ at the curve point $P_j$.
                        I.e. $\vec{Q}'_j$ is a unit vector tangent to the curve.

    4.          Compute the curved step length $sl_j$ between $\sigma_{i-1}$ and $Q_j$.

    5.          $\mu' = \frac{1}{2}\mu(1 + \vec{\text{footdir}}_{i-1} \cdot \vec{Q}'_j)$.

        } until $(sl_j \geq \mu')$

    6. Output $(\vec{Q}_j, \vec{Q}'_j)$ as the next foot print.
  }
```

□

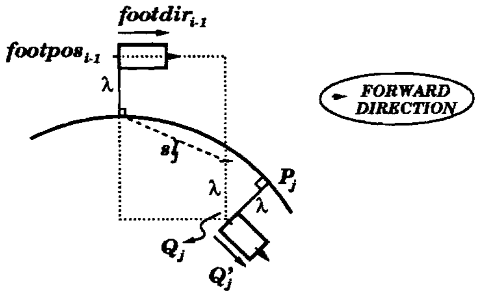

Figure 5: The Diagram for Algorithm 1

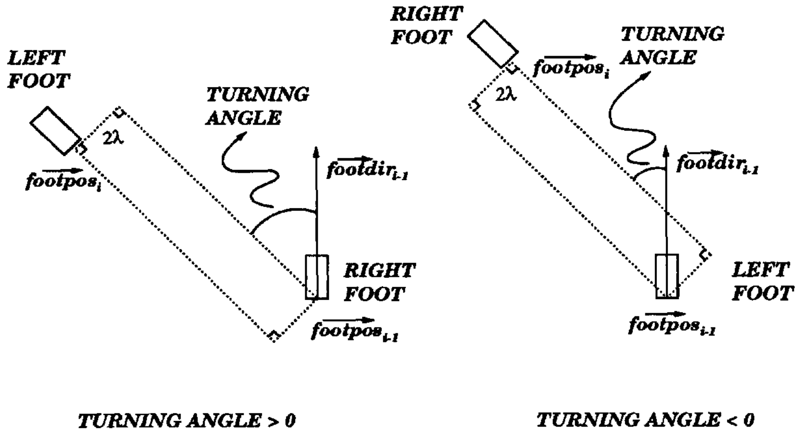

Figure 6: The Definition of the Turning Angle of a Curved Step

3.2 The Second Phase

One purpose of this phase is to modify the foot print sequence obtained in the first phase so that the foot may be aimed in the anticipated direction. Another purpose is to make bigger turns in the steps of the *favored* direction: for example, when the left foot is the stance foot, turning to the right is easier than to the left, if the angle is the same.

To describe the tightness of a curved step quantitatively, we need a more precise definition of the *turning angle* of a step (σ_{i-1}, σ_i): We can *uniquely* determine the rectangle which has $\overrightarrow{footpos}_{i-1}$ and $\overrightarrow{footpos}_i$ as its two non adjacent vertices, and a lateral side with length 2λ the direction of which depends on whether the stance foot is left or right (Figure 6). The turning angle is defined as the *signed* angle between $\overrightarrow{footdir}_{i-1}$ and the longitudinal direction of the rectangle. It is positive in the favored direction, and negative in the unfavored direction. According to the sign of its turning angle, a step is said to be positive or negative.

In real walking, as a matter of fact, a positive step tends to turn the greater angle. This phenomenon, however, is more or less exaggerated in our animation. Because of our two assumptions, after the heel strike moment the swing leg proceeds in the direction of the old swing foot print direction until the toe off moment. Therefore in a positive step the swing knee gets away from the stance knee during that interval, and in a negative step it gets close. The second case is visually less tolerable. To compensate for this, in the second phase of the FPG the turning angles are basically shifted to the positive direction.

In the following algorithm the shift of the turning angle is done through line 7 by slightly displacing the foot print position laterally outward in negative steps. δ_1 is used to control the amount of shift. If it is zero there is no shift; if it is one the turning angle becomes zero for the negative steps. Lines 4 and 6 modify the foot print direction for anticipation. In a negative step (line 6), instead of anticipating, it actually reduces the foot direction change from the previous foot. δ_2 can be adjusted to control the amount of the anticipation: $\delta_2 = 0$ for no anticipation; $\delta_2 = 1$ for excessive anticipation. Most of the time, and in producing the accompanying animation, δ_1 was set to 0.25 and δ_2 was set to 0.5.

Algorithm 2 (**INPUT** : *δ_1, δ_2, and Temporary Foot Print Sequence Obtained in the First Phase,* **OUTPUT** : *Final Foot Print Sequence*)

```
• for $(i = 1; i \leq N_{steps}; i = i + 1)$ {
    1. Compute the turning angle $\theta_i$ of $(\sigma_{i-1}, \sigma_i)$.
    2. Compute the lateral distance $ld$ of $\sigma_{i-1}$ from $\sigma_i$ (Figure 7).
    3. Let $\overrightarrow{\mathrm{footdir}}^{\perp}_{i-1}$ be the unit vector inward and perpendicular to $\overrightarrow{\mathrm{footdir}}_{i-1}$.
    4. if $\theta_i > 0$   $\overrightarrow{\mathrm{footdir}}_i = \mathbf{unitize}(\delta_2 \overrightarrow{\mathrm{footdir}}_i + (1 - \delta_2)\overrightarrow{\mathrm{footdir}}_{i+1})$.
    5. else {
    6.     $\overrightarrow{\mathrm{footdir}}_i = \mathbf{unitize}(\delta_2 \overrightarrow{\mathrm{footdir}}_i + (1 - \delta_2)\overrightarrow{\mathrm{footdir}}_{i-1})$.
    7.     $\overrightarrow{\mathrm{footpos}}_i = \overrightarrow{\mathrm{footpos}}_i + \delta_1(2\lambda - ld)$.
       }
  }
```

□

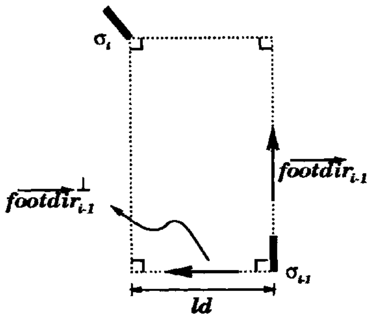
σ_i
$\overrightarrow{footdir}_{i-1}$
$\overrightarrow{footdir}^{\perp}_{i-1}$
σ_{i-1}
ld

Figure 7: ld and $\overrightarrow{footdir}^{\perp}_{i-1}$ of a Curved Step

where **unitize** is an operator that unitizes a vector.

3.3 Summary

The complexity of the first phase is obviously $O(N_{curve\_points})$. The complexity of the second phase is $O(N_{steps})$. Because N_{curve_points} is far larger than N_{steps}, the overall complexity of FPG is $O(N_{curve\_points})$. N_{curve_points} depends on the point density on the curve. In generating 30 steps with a dense curve, FPG does not take more than 0.1 second. Because this FPG is done more or less in an off-line fashion, the complexity of this part will be treated separately from that of the WMG part later.

4 The Higher Level Part of the Walking Motion Generation: Trajectories of the Center Site and the Ankles

The center site location C_{HSM} at the heel strike moment is approximated as the inner division point of $\overrightarrow{footpos}'_{i-1}$ and $\overrightarrow{footpos}'_{i}$ with ratio κ_1:κ_2, where $\overrightarrow{footpos}'_{i-1}$ and $\overrightarrow{footpos}'_{i}$ are the points displaced from $\overrightarrow{footpos}_{i-1}$ and $\overrightarrow{footpos}_{i}$ by λ laterally inwards, as shown in Figure 8. κ_1:κ_2 is the ratio of the center site in the underlying linear step. Note that the displacement of $\overrightarrow{footpos}'_{i}$ from σ_i is based on the direction $\overrightarrow{footdir}_{i}$ (not on $\overrightarrow{footdir}_{i-1}$).

In our algorithm, the center site (top view) moves along a second degree de Casteljau curve [5], whose three control points are shown as P_1, P_2, and P_3 in Figure 9 (A). The orientation of the center site (and thus the pelvis) is assumed to face the derivative direction of the curve and to be horizontal. C_{HSM} obtained above is used for P_3. The current (at the beginning of the current

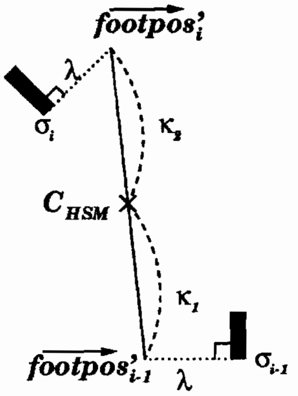

Figure 8: The Position of the Center Site at the Heel Strike Moment (Top View)

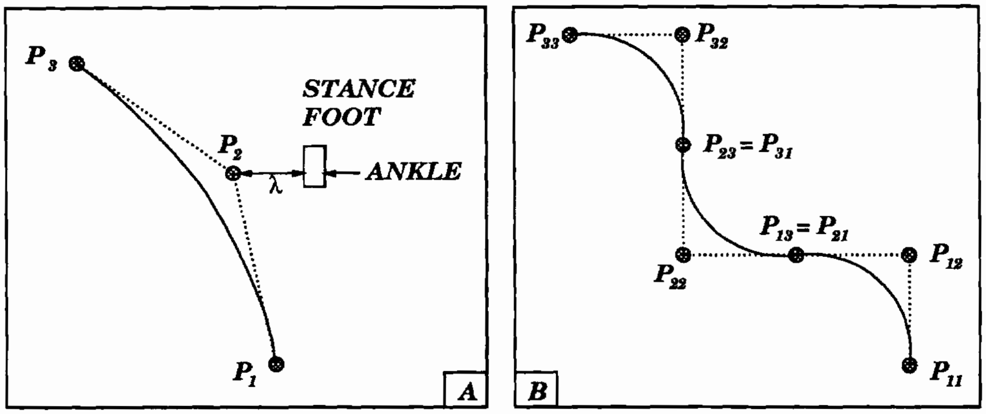

Figure 9: The Trajectory of the Center Site (Top View)

step) center site is used for P_1. P_2 is obtained by laterally displacing the stance ankle by λ inward. The planar center site trajectory of three steps are shown in Figure 9 (B). C_{HSM} is defined in such a way that the latter line segments of the previous steps ($P_{12}P_{13}$ and $P_{22}P_{23}$ in the Figure) are always collinear with the first line segments of the next steps ($P_{21}P_{22}$ and $P_{31}P_{32}$ in the Figure), respectively. Thus the resulting trajectory of the center site is C^1 continuous.

Let $L[0,1]$ represent the trajectories (center site, stance ankle, and swing ankle altogether) of the underlying linear step from normalized time 0 to 1. $L(t)$ is the snapshot of $L[0,1]$ at t. $L^C[0,1]$, $L^{STA}[0,1]$, and $L^{SWA}[0,1]$ are the linear trajectories of the center site, the stance ankle, and the swing ankle, respectively. Similarly, $L^C(t)$, $L^{STA}(t)$, $L^{SWA}(t)$ are the snapshots at t. The corresponding trajectories and snapshots $C[0,1]$, $C(t)$, $C^C[0,1]$, $C^{STA}[0,1]$, $C^{SWA}[0,1]$, $C^C(t)$, $C^{STA}(t)$, and $C^{SWA}(t)$ are defined for the planar trajectory of the curved step.

Suppose that we already have computed $L[0,1]$ and $C[0,1]$. We want to build a correspondence between $L[0,1]$ and $C[0,1]$, so we form an arc length ratio preserving matching function (ALRPMF) ϕ between the center site trajectories of the underlying linear step and the curved path step (planar):

$$C(t) = \phi(L(t)) \tag{2}$$

Such a function can be approximated by computing $C[0,1]$ at much smaller steps than $L[0,1]$ (about 100 times as many), and then approximating the arc length as the sum of small line segments. (Of course, in the implementation, we should have three subfunctions, ϕ^C, ϕ^{STA}, ϕ^{SWA}.)

Any point on the linear trajectory can then be matched to the corresponding point on the planar curved trajectory. This function is essential to **ADVANCE_STEP**. If a normalized time t is given for **ADVANCE_STEP**, the center site location in the linear step is looked up, and the center site of the curved step is computed through ϕ.

In getting the height value of the center site later in **ADVANCE_STEP** at the normalized time t, we use the underlying linear step information. Let the *triangular leg* be the line segment between the hip and the ankle of the stance leg. We assume that the length of the triangular leg of the underlying linear step and that of the curved step is the same, at any moment during the step. Also we assume that the stance *foot* configuration in the curved step is the same with the one in the linear step. (Thus ϕ^{STA} is the identity function.) As shown in the Figure 10, the height h_C of the center site of the curved step is

$$h_C = \sqrt{(h_L - h_A)^2 - x^2} + h_A \tag{3}$$

where h_L is the height of the center site in the underlying linear step, h_A is the height of the stance ankle at the normalized time t, and x is the offset of the stance hip from the stance ankle in the planar lateral direction. The planar position of the stance hip is determined by the position and orientation of the planar center site.

The planar trajectory followed by the swing ankle is approximated by a second degree de Casteljau curve [5]. The three control points are given by the position D_1 of the swing ankle at toe off, which is given from the underlying linear step due to our two assumptions: i.e, the swing foot of the curved step moves in the same way as in the underlying linear step until toe off. Let γ be the line in the direction of $\overrightarrow{footdir}_{i-1}$, displaced from the stance ankle laterally inward by λ. D_2 is the symmetric point of the stance ankle with respect to the line γ (Figure 11). The ankle position

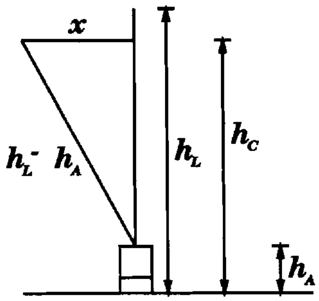

Figure 10: The Height of the Stance Hip in a Curved Step (Rear View)

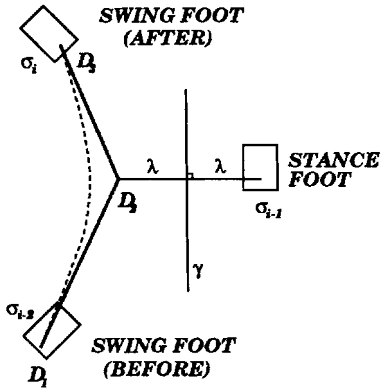

Figure 11: The Trajectory of the Swing Ankle

D_3 at the next heel strike is computed from σ_i. The foot sole angle and the height component of the swing ankle is inherited from the underlying linear step.

As a summary, we construct the three basic trajectories and the ALRPMF of a curved step. Note that if the path is actually a straight line, these trajectories will be the same as those of the underlying linear steps, according to our construction.

Consider the time complexity excluding the LPL algorithm computation. Let C_{center_site} and C_{swing_ankle} be the integer *constants* that represent the number of de Casteljau points for storing the trajectory of the center site and the swing ankle of *one* curved step, respectively. Let L_{center_site} and L_{swing_ankle} be the integer *constants* that represent the number of points describing the trajectory of the center site and the swing ankle of an underlying linear step, respectively.

C_{HSM} can be computed in $O(1)$ time. The curved center site trajectory can be computed in $O(C_{center\_site})$. The curved swing ankle trajectory can be computed in $O(C_{swing\_ankle})$. ϕ can be constructed by first scanning the $C[0,1]$ to compute the total length, and then establishing a crude correspondence by scanning $L[0,1]$ and $C[0,1]$ in parallel, taking $O(C_{center\_site}+C_{swing\_ankle})$. Therefore the overall complexity of this section is $O(C_{center\_site}+C_{swing\_ankle})$.

Later call of ϕ will need two binary searches (one in $L[0,1]$ and the other in a segment of $C[0,1]$) and an interpolation. It takes $O(\log(L_{center\_site}+L_{swing\_ankle}))$ + $O(\log(\frac{C_{center\_site}}{L_{center\_site}}+\frac{C_{swing\_ankle}}{L_{swing\_ankle}}))$ + $O(1)$ time, which turns out to be $O(\log(C_{center\_site}+C_{swing\_ankle}))$.

5 The Lower Level Details of the Walking Motion Generation

5.1 Computation of the Center Site, Stance Ankle, and Swing Ankle

Once the trajectories of the center site and the swing ankle are formed, computing their locations at a specific normalized time t is straightforward. All those locations are sought first in the underlying linear step. Note that the stance ankle location of the curved step is identical to that of the linear step. Through ϕ, we can find the locations of the center site and the swing ankle of the curved step on the curved trajectories.

The location of the hips can be easily determined based on the position and the orientation of the center site. If the pelvis rotation is considered, which can be given by a simple sinusoidal function, we can perturb the center site position and orientation accordingly, and the subsequent computation can be similarly performed to determine the hips.

The time complexity of this subsection is basically the two computations of the function ϕ. As discussed above, it can be done in $O(\log(C_{center\_site}+C_{swing\_ankle}))$.

The next two subsections will show how the stance and swing leg configurations are determined. In each case we will consider (1) how to connect those computed hip and ankle with the thigh and the calf, (2) how to decide the configuration of the foot.

5.2 Stance Leg

The configuration of the stance foot is inherited from the underlying LPL system. Therefore in the stance leg, the configurations of only the thigh and the calf remain to be decided. Because the stance hip and stance ankle are determined, there is only one degree of freedom to connect these two points with the thigh and the calf: the rotation of the knee around the axis defined by the stance hip and the stance ankle.

In the anatomical aspect, the ankle is regarded as a joint with *two* degrees of freedom: flexion-extension and inversion-eversion [4]. That is, the ankle can hardly be twisted to produce the rotational degree of freedom of the previous paragraph with the foot fixed. This is more severely imposed by our two assumptions.

Therefore all the twist between the pelvis and the stance foot is achieved through the stance hip joint. The joint angle around each axis at the hip can be computed by an Euler angle computation in $O(1)$ time.

5.3 Swing Leg

The swing leg (including the foot) configuration during the double stance phase is determined in the same way as in the stance leg, since the swing foot behaves exactly the same way as in the underlying linear step until the toe off.

From the toe off moment, however, the planar direction of the swing foot changes. The planar swing foot direction is assumed to be the derivative direction of the swing ankle trajectory. The swing foot (sole) angle around the flexion-extension axis is inherited from the underlying linear step. The swing foot angle around the inversion-eversion axis is considered to be zero during the swing. (It is maintained at zero until the toe off moment by our two assumptions.) The swing foot angle together with the swing ankle location computed above determines the swing foot configuration. Of course, the angle at the ball of foot is zero. (In real walking on bare feet, this angle is not zero [9], but it is neglected in our work.)

As in the stance leg, once the swing foot configuration is fixed, there is no degree of freedom in deciding the configuration of the swing thigh and calf. Thus swing leg configuration can be decided similarly as in the stance leg, taking $O(1)$ time.

5.4 Torso Motion for Banking

The displacement of the center site of the curved path from that of the underlying linear path is shown in Figure 12. It mostly results from the stance ankle joint. However, banking should be considered in terms of the center of mass of the whole body. Therefore the upper body has to be

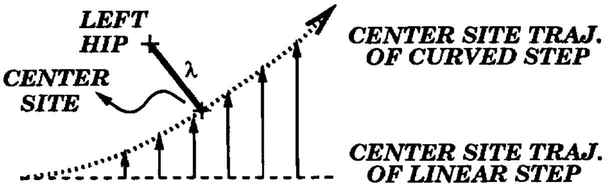
LEFT HIP
CENTER SITE
λ
CENTER SITE TRAJ. OF CURVED STEP
CENTER SITE TRAJ. OF LINEAR STEP

Figure 12: Displacement of the Center Site in a Curved Step

bent further to produce the correct banking. The overall banking is given by

$$\omega = \arctan(\frac{\kappa v^2}{g}) \tag{4}$$

where v is the velocity, g is the gravity, and κ is the curvature of the curve [6]. Here we use the center site trajectory as an approximation to get the curvature. (The center site is not far away from the center of mass, especially when it is seen from the top.) The upper body should be bent so that the center of mass may make the angle ω around the stance ankle with respect to the ground.

The torso is modeled by 17 segments [13] in our implementation. An iterative method is used to compute the current center of mass and reduce the difference from the current one and the desired one [17]. Five iterations were enough in most of the cases. The computation of the center of mass is $O(C_{bsegs})$, where C_{bsegs} is the number of segments in the body, a constant. Therefore this subsection can be done in $O(C_{bsegs})$.

5.5 The Head and Torso Motion for Eye Gaze Direction – Anticipation

For realistic motion, eye contact is maintained two steps ahead as shown in Figure 13: σ_i is used as the stance foot, and the foot located at σ_{i-1} is about to be located at σ_{i+1} by the current step. In this case, the first eye contact (at the moment the step starts) is at the mid point of σ_{i+1} and σ_{i+2}, and the last contact (at the moment the step ends) is at the mid point of σ_{i+2} and σ_{i+3}. The eye gaze direction during the step is obtained by interpolating the first and last eye contact points. For anticipation, a few more foot prints should actually be passed to **INITIALIZE_STEP** than the parameters shown in Section 2.

Once the eye gaze point is determined, the angle of the eye direction relative to the pelvis (facing) direction can be computed. In our work, $\frac{2}{3}$ of the anticipation is done by the twist at the neck. The remaining $\frac{1}{3}$ is evenly distributed through the whole torso. Thus each vertebra is twisted by $\frac{1}{3} \times \frac{1}{17}$ of the total anticipation. Obviously this subsection takes $O(1)$.

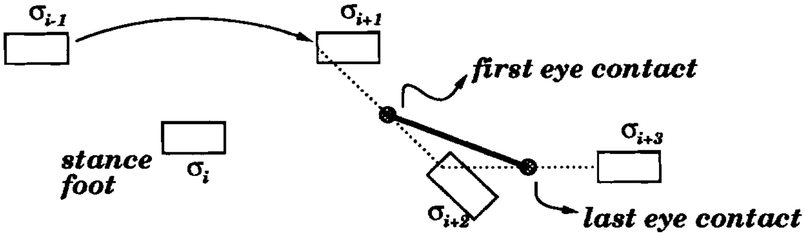

Figure 13: The Eye Contact Trajectory

5.6 Arm Swing

The swing of the arm mostly depends on the leg movements. The swing is bigger for the longer steps. The elbow angle decelerates in the forward swing, and accelerates in the backward swing. A minimum elbow angle (positive) is set so that it should not be passed at the end of the backward swing. Time complexity of this part is also $O(1)$.

5.7 Summary

If the curved steps are actually along a linear path, the resulting curved path walk will be the same as the underlying LPL. The complexity of this section is $O(\log(C_{center\_site} + C_{swing\_ankle}) + C_{bsegs})$.

6 The Initial and Final Steps

They are obtained through a slight modification of the normal step generation procedure. The previous two sections can be followed except for the two things.

First, the ALRPMFs should be defined during the appropriate intervals. They were defined from the entire linear trajectories to the curved trajectories: i.e. in the normal step, $L[0,1]$ was mapped one-to-one and onto $C[0,1]$. However, in the initial step, $L[\alpha,1]$ $(0 \leq \alpha < 1)$ is mapped to $C[0,1]$. The value of α depends on the relative foot print locations just before the initial step. It is decided so that the stance foot configuration of the underlying linear step at α coincides with the initially given configuration just before the walking. If there are multiple choices (usually, the initial foot is horizontal, and in walking there is an *interval* during which the foot is maintained flat on the ground), choose the time of the closest stance leg configuration in the underlying linear step with the initial configuration. Similarly, in the final step, $L[0,\beta]$ $(0 < \beta \leq 1)$ is mapped to

$C[0,1]$. The value of β is determined so that the final foot configuration may end up in the desired one.

Second, the swing foot angle around the flexion-extension axis should be modified. For example, at the final step, the foot sole should be monitored so that it is placed flat (or in a desired way) at the end of the step. This can be easily done by locking the foot flat if β is after the flat point, or accelerating the foot sole angle if β is before the point.

7 Maintaining the Continuity of Velocity between the Steps

Because we allow different step sizes and durations, there can be a discontinuity of velocity at the boundaries of the steps. It is avoided by distributing this instantaneous velocity difference into an interval: at the beginning it takes the old velocity. During the next $C[0, \frac{1}{3}]$ it is accelerated or decelerated and achieves the normal speed. This velocity adjustment is performed in **INITIALISE_STEP**. Because the velocity difference is usually not large, this method works pretty well.

8 Results and Conclusion

The complexity of the whole algorithm excluding the LPL and FPG computation is the sum of the complexities in Section 4 and 5, which is $O(C_{center\_site} + C_{swing\_ankle}) + O(\log(C_{center\_site} + C_{swing\_ankle}) + C_{bsegs})$. All of C_{center_site}, C_{swing_ankle}, and C_{bsegs} are constants. For example, in our implementation, $L_{center\_site} = 70$, $L_{swing\_ankle} = 70$, $C_{center\_site} = 7000$, $C_{swing\_ankle} = 7000$, and $C_{bsegs} = 71$. Moreover, the complexity of **ADVANCE_STEP** alone is just $O(\log(C_{center\_site} + C_{swing\_ankle}) + C_{bsegs})$, which is called much more often than **INITIALISE_STEP**. Therefore our curved path walking motion generation is a *constant time algorithm* per each step, and it can be used as a *filter* on top of an LPL system.

Figure 14 shows the foot print sequence generated by the interactive *foot print generator.* A curve can be edited by inserting, deleting, moving the control points. If μ values are given by the user, the sequence is generated according to the algorithms in Section 3.

The curved path walking algorithm is implemented in *Jack*$^{\text{TM}}$ [17]. Ko and Badler's straight line walking algorithm [11] is used for the underlying LPL system. This algorithm can handle the

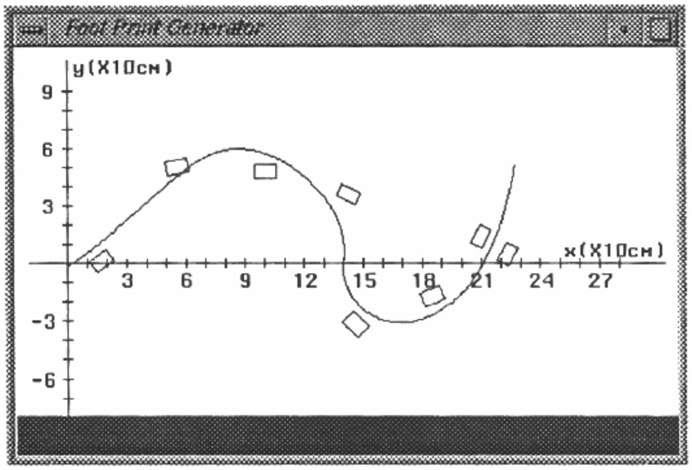

Figure 14: Steps Generated by the Foot Print Generator

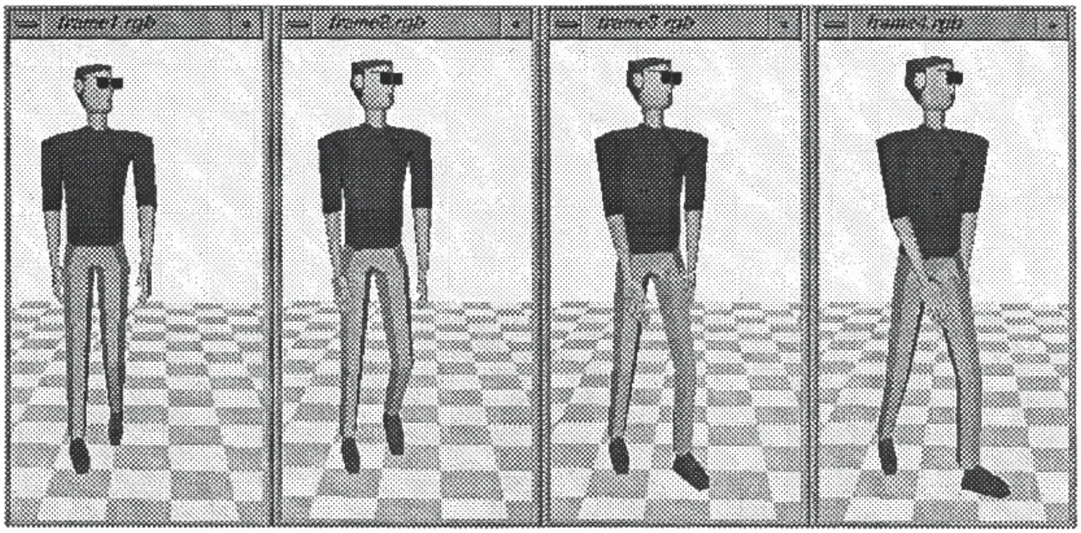

Figure 15: Four Snapshots during a Turning Step

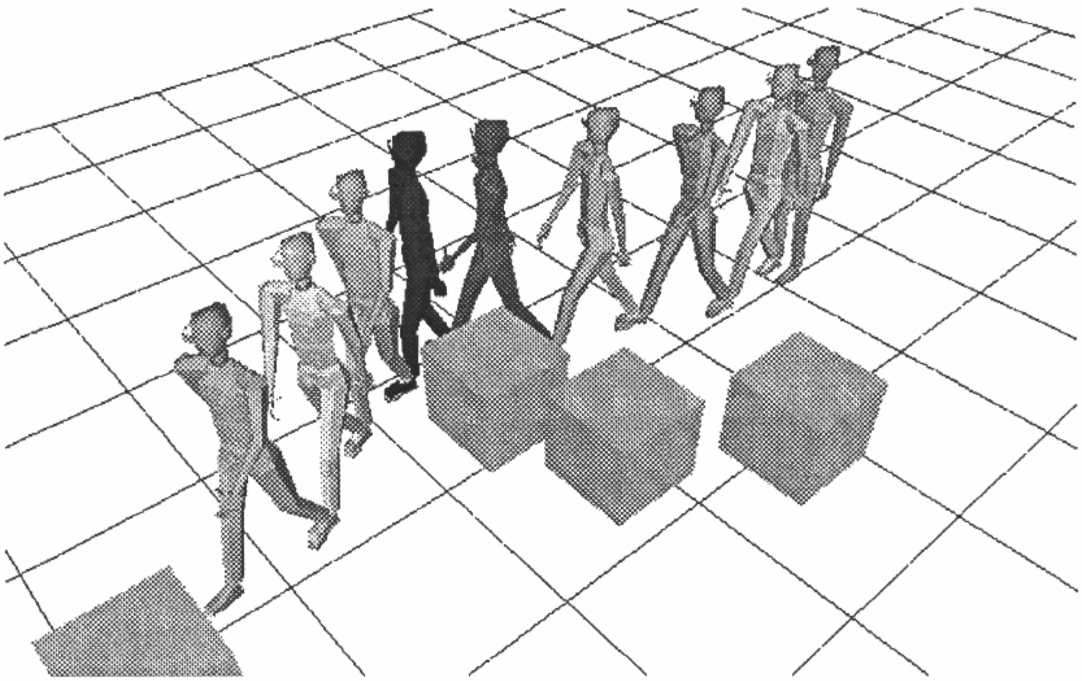

Figure 16: A Curved Path Walk Avoiding Obstacles

anthropometrical variety as well as different step lengths. Figure 15 shows four snapshots during a curved step. Figure 16 shows the figure walking along a curved path to avoid obstacles.

Through a *reasonable generalization*, this algorithm generates curved path locomotion from an underlying linear path walk. Experiments prove that this method is very *robust.* As the accompanying animations demonstrate, the walking motion is quite *realistic.* Also this method enables people to study the straight line walk *independently* from the curved path walk. Therefore we conclude this method is an *effective* and *practical* way of producing curved path human locomotion.

A limitation of this method – more angle is turned in positive steps than in negative steps compared with the real walking – is imposed by our two assumptions. This is noticeable in tight turns such as walking around a 100cm radius circle (the animation in the accompanying video tape). With a variable axis at the ball of the foot or a deformable foot (even without sliding), this problem can be greatly reduced. On the other hand, the rigid segments and fixed axis are very popular in animating linked structures, because of their computational efficiency. Our solution can be considered a viable trade-off between realism and efficiency.

9 Acknowledgments

This research is partially supported by ARO Grant DAAL03-89-C-0031 including participation by the U.S. Army Human Engineering Laboratory, Natick Laboratory, and NASA Ames Research Center; U.S. Air Force DEPTH contract through Hughes Missile Systems F33615-91-C-0001; MOCO Inc.; and NSF CISE Grant CDA88-22719. Partial support for this work was provided by the National Science Foundation's Instrumentation and Laboratory Improvement Program through Grant number USE-9152503.

References

[1] Ronan Boulic, Nadia Magnenat-Thalmann, and Daniel Thalmann. A global human walking model with real-time kinematic personification. *The Visual Computer*, 6:344–358, 1990.

[2] Armin Bruderlin. Goal-directed, dynamic animation of bipedal locomotion. Master's thesis, Simon Fraser University, 1988.

[3] Armin Bruderlin and Thomas W. Calvert. Goal-directed, dynamic animation of human walking. *Computer Graphics*, 23(3):233–242, July 1989.

[4] M. Cartmill, L. W. Hylander, and J. Shafland. *Human Structure.* Harvard University Press, Cambridge, Massachusetts, and London, England, 1987.

[5] Gerald E. Farin. *Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide.* Academic Press, Boston, second edition, 1990.

[6] Michael Girard. Interactive design of 3D computer-animated legged animal motion. *IEEE Computer Graphics and Applications*, 7(6):39–51, June 1987.

[7] Michael Girard and A. A. Maciejewski. Computational modeling for the computer animation of legged figures. *Computer Graphics*, 19(3):263–270, July 1985.

[8] Marc Grosso, Richard Quach, and Norman I. Badler. Anthropometry for computer animated human figures. In N. Magnenat-Thalmann and D. Thalmann, editors, *State-of-the Art in Computer Animation*, pages 83–96. Springer-Verlag, New York, NY, 1989.

[9] Verne T. Inman, Henry J. Ralston, and Frank Todd. *Human Walking.* Williams and Wilkins, Baltimore/London, 1981.

[10] Shuuji Kajita, Kazuo Tani, and Akira Kobayashi. Dynamic walk control of a biped robot along the potential energy conserving orbit. *IEEE International Workshop on Intelligent Robotics and Systems, IROS '90*, pages 789–794, 1990.

[11] Hyeongseok Ko and Norman I. Badler. A kinematic generalization of rotoscoped human walking. Technical Report MS-CIS-92-79, University of Pennsylvania, Dept. of Computer and Information Science, Philadelphia, PA 19104-6389, October 1992.

[12] Michael McKenna and David Zeltzer. Dynamic simulation of autonomous legged locomotion. *Computer Graphics*, 24(4):29–38, August 1990.

[13] Gary Monheit and Norman I. Badler. A kinematic model of the human spine and torso. *IEEE Computer Graphics and Applications*, 11(2), 1991.

[14] M. Pat Murray. Gait as a total pattern of movement. *American Journal of Physical Medicine*, 46(1):290–333, 1967.

[15] M. Pat Murray, A. Bernard Drought, and Ross C. Kory. Walking patterns of normal men. *The Journal of Bone and Joint Surgery*, 46-A(2):335–360, March 1964.

[16] S. Onyshko and D. A. Winter. A mathematical model for the dynamics of human locomotion. *Journal of Biomechanics*, 13:361–368, 1980.

[17] Cary B. Phillips and Norman I. Badler. Interactive behaviors for bipedal articulated figures. *Computer Graphics*, 25(4):359–362, July 1991.

[18] Marc H. Raibert and Jessica K. Hodgins. Animation of dynamic legged locomotion. *Computer Graphics*, 25(4):349–358, July 1991.

[19] G. Ridsdale and T. W. Calvert. Animating microworlds from scripts and relational constraints. In *Proceedings of Computer Animation '90*, Geneva, 1990.

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

[21] David A. Winter. *Biomechanics and Motor Control of Human Movement.* Wiley, New York, second edition, 1990.

[22] David A. Winter, Arthur O. Quanbury, Douglas A. Hobson, H. Grant Sidwall, Gary Reimer, Brian G. Trenholm, Thomas Steinke, and Henry Shlosser. Kinematics of normal locomotion - a statistical study based on T.V. data. *Journal of Biomechanics*, 7:479–486, 1974.