MoCap Based New Walk and Climb Synthesis

Shrinath Shanbhag
CDAC Mumbai
shrinath@ncst.ernet.in

Sharat Chandran
IIT Bombay
sharat@cse.iitb.ac.in

Abstract

We describe a method to dynamically synthesize believable, variable stride, and variable foot lift motions for human walks and climbs. Our method is derived from a single *motion captured* walk sequence*, and is guided by a simple kinematic walk model. The method allows control in the form of* stride *and* lift *parameters. It generates a range of variations while maintaining individualistic nuances of the captured performance.*

1. Introduction

Walking is a fundamental motion for us humans. The number of variations of *"the walk"* an individual is capable of performing is potentially infinite. Even so, animated virtual characters in interactive applications such as games, are driven by a severely limited repertoire of walk sequences. These are often only two sequences — a run and a walk. As a result we see walk animation that does not adapt to the environment and is visually unpleasant. For example, climbs are many a times synthesised as walk-sliding-along-an-e incline, resulting in animations that look unnatural.

To better address this some current games employ physically based simulation schemes for animation. For example 'rag-doll' dynamics simulations are often employed to simulate falling and bouncing in real time. Such schemes treat objects as passive and totally governed by Newton's laws of motion. This is not applicable for virtual humanoid characters as they are active and can initiate motion on their own. Attempts at physically synthesizing motion have succeeded in generating only simple motions. It is still difficult to create controllers capable of generating a wide range of human motion. There is, therefore, a strong dependence on pre-created motion.

An alternative is to use motion capture (mocap) driven character animation. Using mocap enables games to easily capture and playback virtually *any* motion that an actor can perform. This is a key property missing from key-framing and physically based simulation systems. However mocap based systems possess one significant drawback inherent to their "bitmap nature" — the lack of dynamic control inherent in procedural methods. Such control may not be essential in certain applications, such as in animation catering to movies. However applications where the playback environment is different from the capture environment suffer from its absence. In addition, interactive applications like games demand dynamic control and adaptation facilities.

Rapid increase in processing power and storage capacities of todays machines translates into the ability to employ *gigantic* mocap databases. Schemes for organizing such databases as graphs and synthesizing motion on demand are available. However, these schemes are more adept at generating high-level motion specifications and not suitable for adapting captured motion to a new environment on the fly. Moreover many such schemes still need human supervision.

1.1 Our contributions

In this paper, we synthesize new walk and climb motions from a single motion captured sequence. Specifically, we describe a new *per frame inverse kinematics* (PFIK) [7] based method that synthesizes variable stride and variable lift walk and climb limb motion from a single motion captured walk sequence using a kinematic walk model. We use *stride* and *lift* as the control parameters. Our use of a single motion clip complements the large database approach. Unlike most mocap based schemes our synthesis can be controlled programatically.

The next section describes related work in this area. The following sections then describe our simple kinematic walk model and our method for synthesizing new motions.

2. Related Work

The history of research in humanoid animation dates back more than 15 years. Motion synthesis methods can be broadly classified as kinematics based, dynamics based and constraint based methods. In addition there are also hybrid methods which mix one or more of the other techniques. More recently, methods based on motion captured

sequences have been developed.

2.1. Gait Synthesis Techniques

Multon et. al [15] provide an excellent survey of computer animation of human walking. Metaxas and Sun [20] describe a low-level gait generator based on sagittal elevation angles, which allows curved locomotion to be created easily. They also describe an inverse motion mapping algorithm, that allows motion to be adapted to uneven terrains. In addition they describe a higher level control frame work that allows motion requirements to be specified at a high level by sketching the desired path. Hodgins et. al [9] describe an algorithm that allows simulation of running, bicycling and vaulting. The simulation is achieved through control algorithms that cause physically realistic models to perform the desired behaviour. Faloustos et. al [5] describe a method to combine various physically based simulation controllers into a unified framework.

2.2. Mocap based Motion Editing Techniques

Mocap based techniques, unlike synthesis techniques described above, start with an existing motion and adapt it to different requirements. Researchers working in this field have proposed a number of innovative techniques adapting signal processing methods or employing constraint based solvers. Bruderlin et. al [4] have successfully applied techniques from image and signal processing domains to designing, modifying and adapting animated motions. Unuma et. al [21] describe a method for modeling human figure locomotions with emotions. Herein Fourier expansions of experimental data of actual human behaviors serve as a basis from which to interpolate or extrapolate the human locomotions. Witkin et. al [22] describe a simple technique for editing captured animation based on warping of the motion parameter curves. Gleicher et. al [7][6]use space-time constraint formulations to modify captured motion and retargeting motion to new characters with different segment lengths. Gleicher [7] provides a comparison of constraint based motion editing methods. Lee et. al [14] describe a hierarchical framework for adapting existing motion of humanoids to externally specified constraints. Gleicher et. al [19] describe a method that takes into consideration physical principles to touch up synthetically generated motion, so as to make it more plausible.

2.3. Mocap Based Synthesis Techniques

Motion synthesis techniques create new motion from existing motion data. Gleicher et. al [11], Lee et. al [13], Forsyth et. al [1] describe techniques to create new motion sequences from a corpus of motion data. Each technique essentially clusters similar motion into nodes. The next phase builds a graph of nodes, where each edge represents a transition between nodes. A walk through the cluster node graph results in synthesis of new motion sequences. The techniques differ in metrics used for clustering, pruning schemes and control criteria for node walk. Bregler et. al [17],[18] describe a scheme for synthesizing missing degrees of freedom and adding details to specified degrees of freedom, for a roughly specified motion sequence. Their method uses the various correlation between the various degrees of freedom (DOF's) within each motion sequences. Forsyth et. al [2] describe a technique using a novel search method based around dynamic programming to interactively synthesize motion from annotations. Here the system synthesizes motion corresponding to an annotated timeline painted by the user. Gleicher et. al [8] present a technique that preprocesses a corpus of motion capture examples into a set of short clips that can be concatenated to make continuous streams of motion. The resulting simple graph structure can be used in virtual environments where control and responsiveness are more important than accuracy. Brand et. al [3] use machine learning techniques to model motion sequences as stylistic HMM parameterized by a style vector. Motion can be synthesized in a number of ways - a random walk over the HMM states, by trying to match given input sequence to an optimum set of HMM states etc. *However, as the method is statistical there is no direct control over the desired motion.*

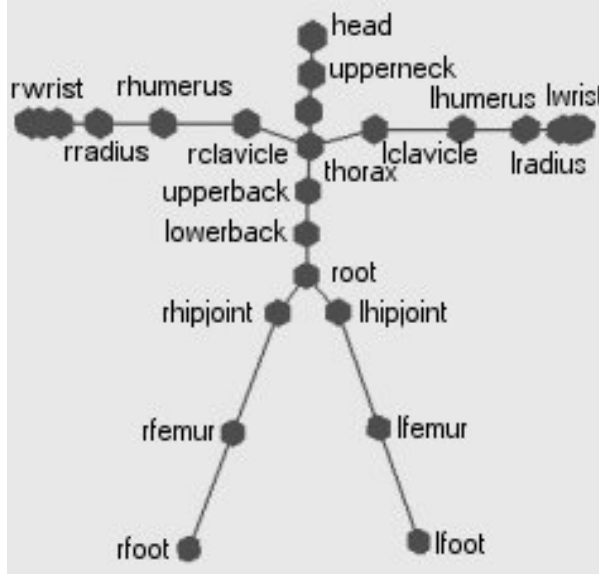

Figure 1. Skeletal hierarchy

3. Our method

Our work falls into the mocap based synthesis category. It differs from the others in this group in that it uses a sin-

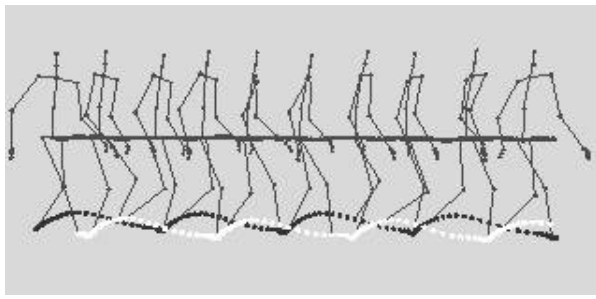

Figure 2. Motion captured base walk sequence

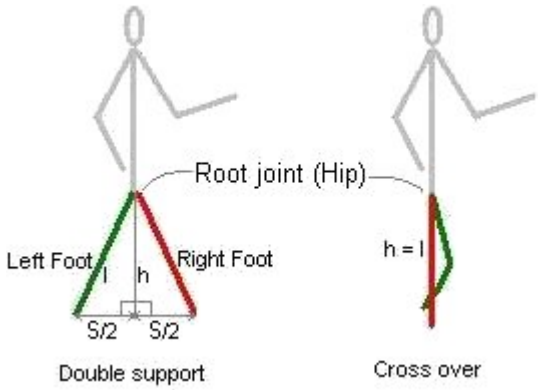

Figure 3. Simple Kinematic Walk Model

gle motion clip, is guided by a kinematic model, and is accessible programatically when new walk sequences are presented.

We start with a skeletal hierarchy, such as in Figure 1, to acquire motion capture data. Motion data consists of a bundle of motion signals. Each signal represents a sequence of sampled values for each degree of freedom (DOF). Motion signals are sampled at discrete instances of time with a uniform sampling interval to yield a motion clip. In each frame, the sampled values of the different DOFs at each joint determine the configuration of an articulated figure for that frame. Often the root of the skeletal hierarchy contains six DOFs (three for translation and three for rotation about x, y and z axis), whereas rest of the nodes contain three DOFs (only rotation). Scaling is mostly unused.

Our base motion sequence is a motion captured straight line walk, shown in Figure 2. The aim is to programatically synthesize variations using stride and lift as control parameters. Our emphasis is on creating *believable motion* as against *simulating physically accurate behaviour*.

3.1. Kinematic Model for Walk

Human walking is a process of locomotion in which the erect, moving body is supported by first one leg and then the other. As the moving body passes over the supporting leg, the other leg is swinging forward in preparation for its next support phase. One foot or the other is always on the ground, and during that period when the support of the body is transferred from the trailing to the leading leg there is a brief period when both feet are on the ground. [10] describes human walking in more detail.

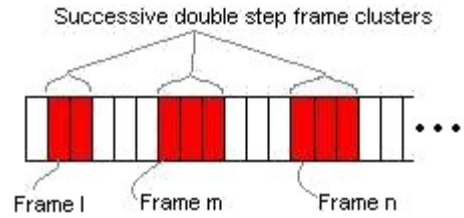

Figure 4. Frame m and Frame n

Here we define a simple kinematic model for human walking that allows us to estimate root-ground clearance h, during the gait cycle. Figure 3 shows ground clearance values for the root joint. The minimum occurs during double support stance. The maximum occurs during the crossover stance. From figure 3 we have

$$h = \begin{cases} \sqrt{l^2 - (\frac{S}{2})^2} & \text{for double step stance} \\ l & \text{for crossover stance} \end{cases} \quad (1)$$

where S is the stride and l is the length of the legs measured from hip to heel.

The root joint, trajectory between two double step for the saggital plane (Y-Z plane in our case) is a sinusoidal wave [10]. If R_i is the position of the root joint for the i^{th} frame, m is the first frame of a double step cluster and n is the first frame of the immediately succeeding double step cluster as in Figure 4, then the y position of the root joint is given by:

$$R_i.y = h_{min} + \delta h * |\sin(\theta)| \quad (2)$$

$$\text{where} \begin{cases} \theta = \frac{\pi*(i-m)}{m-n}, m \leq i \leq n \\ h_{min} = \sqrt{l^2 - (\frac{S}{2})^2} \\ \delta h = l - h_{min} \end{cases}$$

Given a skeletal model with leg length L and a walk sequence with desired stride S, we recompute the root trajectory in the saggital plane using the above equation. The trajectory for the root joint in the transverse plane (Z-X plane in our case) is retained from the original motion clip. Figure 5 shows the trajectories for the root joint and the left foot joint, for a half walk cycle. In the second cycle, the root joint trajectory repeats itself and the right foot follows a trajectory similar to the one shown for left foot.

3.2. Preprocessing

We preprocess the base walk sequence to identify and annotate foot plant and double support frames. We com-

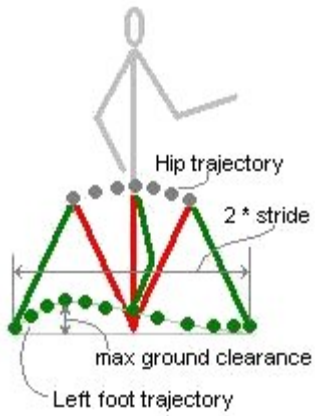

Figure 5. Root and Foot trajectories in the saggital plane

pute per frame relative displacement vectors of root and feet joints. We compute per frame foot lift vectors and mean foot lift vector.

3.2.1 Identifying Foot Plant Frames

We identify foot plant constraints as follows. We identify frames with zero crossings for vertical displacement (the y axis in our case). We select all frames which are close to the ground, within a given threshold. This forms the seed set of foot plant frames. For most normal walk sequences, we observe that the foot is placed on the ground for more than one frame. However, in a motion captured sequence, the foot positions may not coincide exactly due to foot skate. [12] describe a technique to identify and correct foot skate. We use a simpler method. From the initial set of foot plant frames obtained above, we sequentially search in either direction and cluster frames, near the seed foot plant frame, where the magnitude of the displacement vector is below a given threshold value. We stop the search at the first frame that fails the test. We then cluster together, like foot plant frames based on their sequence in the clip.

3.2.2 Computing Relative Displacement Vector and Stride

We compute per frame relative displacement vectors for the *root node*, *left foot node* and *right foot node*. The displacement vector is computed as follows:

Let P_{j_i} and $P_{j_{(i+1)}}$ be the world positions of joint j at frame i and $i+1$ respectively. Then the relative displacement vector D_{j_i} for joint j at frame i is given by:

$$D_{j_i} \leftarrow P_{j_{(i+1)}} - P_{j_i} \tag{3}$$

The the foot stride vector S is computed by adding individual frame displacement vectors of frames for either left or right foot joint from frame l to frame n, where l and n are as explained in Figure 4. The magnitude of S, so computed, is twice the stride. The stride for left and right feet in a rhythmic walk motion are equal.

$$S = \frac{\sum_{i=l}^{n} D_{j_i}}{2} \tag{4}$$

3.2.3 Computing Lift Vector

Computing the lift vector for the base sequence, involves projecting the feet position on the ground plane, and computing the vector difference of the two positions.Let P_{j_i} and P'_{j_i} be the world positions of joint j, representing a foot, at frame i and its projection on the ground plane respectively. Then the lift vector $L_{j_i}$for joint j at frame i is given by:

$$L_{j_i} \leftarrow P_{j_i} - P'_{j_i} \tag{5}$$

Traversing through the frames sequentially, we identify frames with local maximas of the foot lift vector. We find the mean magnitude, L_m, of magnitudes of local maxima of foot lift vectors. This is used to determine the foot lift scaling factor as explained in Section 3.3.2.

3.3. Synthesis

In this section we describe our synthesis of variable stride, variable foot lift and climb. We synthesize motion only for the lower body and playback the upperbody animation as originally recorded. The method in [19] can be applied as a post process to our synthesis for this purpose. Sample results of our work are best seen in video; a glimpse is shown in Figure 3.3.

3.3.1 Varying stride

The stride of the captured walk sequence is varied by directly scaling the relative displacement vectors D_{j_i}, with a scaling factor s. The displacement vectors of the root joint are scaled by the same scaling factor.

$$s = \frac{\textit{new stride}}{|S|} \tag{6}$$

The trajectory of the root joint in the saggital plane is obtained from equation 2. A new trajectory is obtained for the root joint and foot joints as follows:

for $i = l$ to $(n-1)$
$\quad P_{j_{(i+1)}} \leftarrow P_{j_i} + s * D_{j_i}$

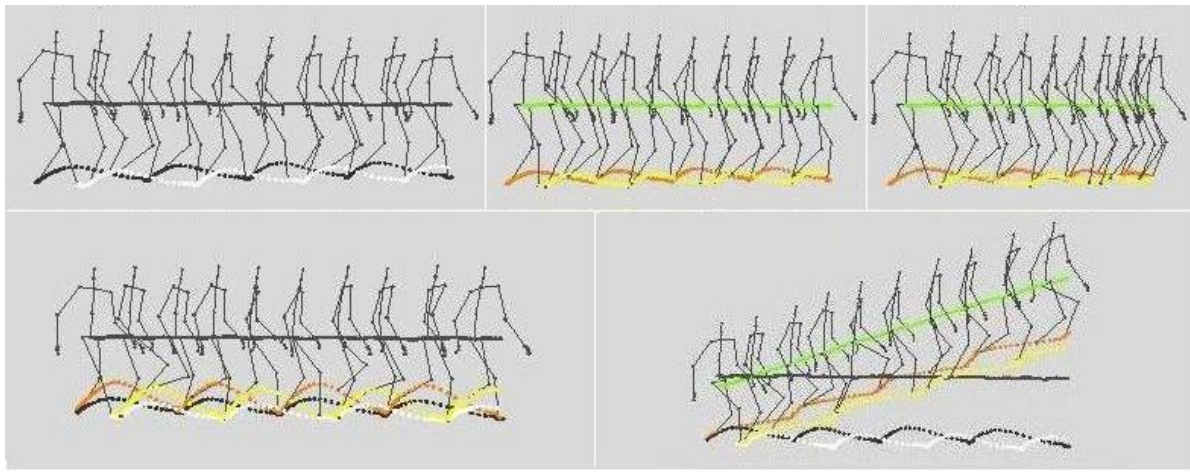

Figure 6. Synthesized walk: Clockwise from top-left – base sequence, uniformly scaled stride, continuously varying stride, climb with base sequence trajectory superimposed, scaled foot lift with base sequence trajectory superimposed

Smoothly varying the scale factor s creates a smoothly varying stride. This may be used to synthesize a accelerating or retarding walk sequence. The DOF angles for each joint in the foot kinematic chain are recomputed, for each frame, by a two link analytical *inverse kinematics solver*[16].

3.3.2 Varying Foot Lift

We vary foot lift by computing new positions for the feet joints by scaling the frame foot lift vectors by a scale factor f. The scale factor is obtained as:

$$f = \frac{\textit{new lift}}{L_m} \tag{7}$$

where L_m is the mean magnitude of local maxima of foot lift vectors as described in Section 3.2.3.

For a motion sequence containing n frames, the new foot trajectory is computed as

for $i = 0$ to $(n-1)$
$\quad P_{j_i} \leftarrow P'_{j_i} + f * L_{j_i}$

where P'_{j_i} is the projection of joint position P_{j_i} on the ground plane. The computation for foot lift and stride are combined as follows

for $i = l$ to $(n-1)$
$\quad P_{j_{(i+1)}} \leftarrow P_{j_i} + s * D_{j_i}$
$\quad P_{j_{(i+1)}} \leftarrow P'_{j_{(i+1)}} + f * L_{j_{(i+1)}}$

The DOF angles for each joint in the foot kinematic chain are recomputed, for each frame, using the IK solver.

3.3.3 Synthesizing climb

Here we describe the synthesis of climb motion, along a plane which makes an angle θ with the horizontal ground plane, as illustrated in Figure 7. The climb is synthesized as a combination of stride and lift as follows. Let S be the desired stride along the inclined plane. The distance parallel to the ground plane is given by $S^{\parallel}$. The corresponding rise in height in ground level is given by $S^{\perp}$.

$$\begin{aligned} |S^{\parallel}| &= |S| * \cos(\theta) \\ |S^{\perp}| &= |S| * \sin(\theta) \end{aligned} \tag{8}$$

For synthesizing the climb motion, we need to estimate the path of the root joint and the feet joints of our articulated body. For this we use the simple kinematic model described in section 3.1. We first synthesize a sequence with stride scaled to $S^{\parallel}$. We use this modified sequence for further modifications as described later.

Consider successive double step frames as in frames l, m, n shown in Figure 4. The geometry of these frames is as visualized in Figure 7. To compute the trajectory of the root joint, note that the root undergoes an additional vertical displacement of magnitude $S^{\perp}$ between each successive double step frame. The trajectory of the root from frame l to m is computed by first displacing the root position at m in the perpendicular direction by $S^{\perp}$. The per frame displacement for in between frames is linearly interpolated as follows:

for $i = l$ to m
$\quad P_{j_i}.y \leftarrow P_{j_i}.y + \frac{(i-l)*S^{\perp}}{(m-l)}$

where P_{j_i} is the position of the root joint in the i^{th} frame of the modified sequence.

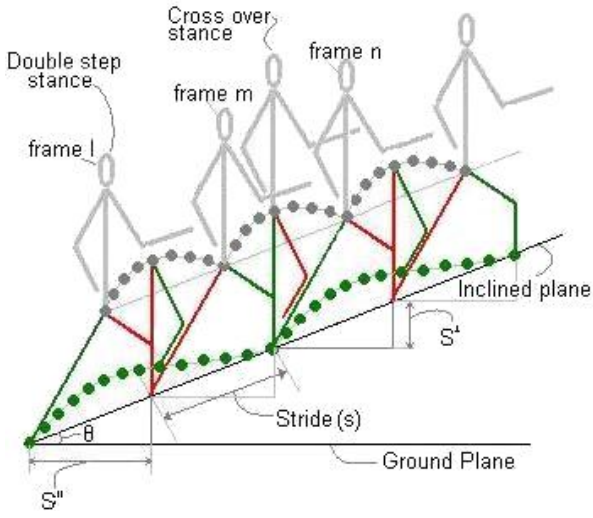

Figure 7. Modelling climb

Trajectory for the left and right foot joints can be calculated in a similar manner. Note that the left foot, shown in green color in Figure 7, accumulates an additional vertical displacement of magnitude $2 * S^{\perp}$ between frames l and m. For frames m to n, the right leg, shown in red, accumulates this vertical displacement. The legs climb alternately, with the planted leg, maintaining its height. The trajectory of the left leg joint from frames l to m can be computed as follows:

for $i = l$ to m

$$P_{j_i}.y \leftarrow P_{j_i}.y + \frac{(i-l)*2*S^{\perp}}{(m-l)}$$

where P_{j_i} is the position of the left leg joint in the i^{th} frame of the modified sequence. The DOF angles for each joint in the foot kinematic chain are recomputed, for each frame, using our IK solver.

4. Conclusion

Motion capture based animation is here to stay given the variety that is otherwise difficult to accomplish. However, the bitmap nature of this data makes it difficult for adaptation in new environments, e.g., in gaming. In this paper, we have described a method to synthesize variable stride and variable foot lift walk and climb using a single motion clip. The basis for this comes from a simple kinematic model adapted from [10]. Using a base motion clip consisting of a motion captured straight line walk, we invent stride and lift as the control parameters to program – as it were – the motion capture data. As an extension of our work, a logical next step is to model real life where people walk along curvilinear paths, sometimes with continously varying gradients.

References

[1] O. Arikan and D. A. Forsyth. Interactive Motion Generation from Examples. In *Proc. of Siggraph '02*, pages 483 – 490, 2002.

[2] O. Arikan, D. A. Forsyth, and J. F. O'Brien. Motion Synthesis from Annotations. In *Proc. of Siggraph '03*, pages 483 – 490, 2003.

[3] M. Brand and A. Hertzmann. Style machines. In *Proceedings of Siggraph '00*, 2000.

[4] A. Bruderlin and L. Williams. Motion Signal Processing. In *Proc. of Siggraph '95*, 1995.

[5] P. Faloutsos, M. van de Panne, and D. Terzopoulos. Composable controllers for physics-based character animation. In *Proceedings of Siggraph '01*, August 2001.

[6] M. Gleicher. Retargetting Motion to New Characters. In *Proc. of Siggraph '98*, 1998.

[7] M. Gleicher. Comparing Constraint-based Motion Editing Methods. *Graphical Model*, 63(2):107 – 134, 2001.

[8] M. Gleicher, H. J. Shin, L. Kovar, and A. Jespen. Snap together motion: Assembling run-time animation. In *2003 Symposium on Interactive 3D Graphics*, April 2003.

[9] J. K. Hodgins, W. L. Wooten, D. C. Brogan, and J. F. O'Brien. Animating human athletics. In *Proceedings of Siggraph '95*, 1995.

[10] V. T. Inman, H. Ralston, and F. Todd. *Human Walking*. Lippincott Williams & Wilkins, 1989.

[11] L. Kovar, M. Gleicher, and F. Pighin. Motion Grahs. In *Proc. of Siggraph '02*, 2002.

[12] L. Kovar, J. Schreiner, and M. Gleicher. Footskate cleanup for motion capture editing. In *Proceedings of the 2002 ACM Symposium on Computer Animation (SCA)*, July 2002.

[13] J. Lee, J. Chai, P. S. A. Reitsma, J. K. Hodgins, and N. S. Pollard. Interactive Control of Avatars Animated with Human Motion Data. In *Proc. of Siggraph '02*, 2002.

[14] J. Lee and S. Y. Shin. A Hierarchial Approach to Interactive Motion Editing for Human like Figures. In *Proc. of Siggraph '99*, 1999.

[15] F. Multon, L. France, M.-P. Cani-Gascuel, and G. Debunne. Computer animaton of human walking: A survey. Technical Report 3441, INRIA, June 1988.

[16] K. Perlin. Two link inverse kinematics. GDC 2002 course notes `http://mrl.nyu.edu/p̃erlin/gdc/ik/talk.html`, March 2002.

[17] K. Pullen and C. Bregler. Animating by Multi-level Sampling. In *Proc. of IEEE Computer Animation 2000*, 2000.

[18] K. Pullen and C. Bregler. Motion Capture Assisted Animation: Texturing and Synthesis. In *Proc. of Siggraph '02*, 2002.

[19] H. J. Shin, L. Kovar, and M. Gleicher. Physical touch-up of human motion. In *Pacific Graphics 2003*, October 2003.

[20] H. C. Sun and D. N. Metaxas. Automating gait generation. In *Proceedings of Siggraph '01*, August 2001.

[21] M. Unuma, K. Anjyo, and R. Takeuchi. Fourier principles for emotion based human figure animation. In *Proceedings of Siggraph '95*, 1995.

[22] A. Witkin and Z. Popovic. Motion Warping. In *Proc. of Siggraph '95*, 1995.