Workshop on Virtual Reality Interaction and Physical Simulation VRIPHYS (2012)
J. Bender, A. Kuijper, D. W. Fellner, and É. Guérin (Editors)

Real-Time Motion Synthesis for Multiple Goal-Directed Tasks Using Motion Layers

Christos Mousas[1] and Paul Newbury[1]

[1]Department of Informatics, University of Sussex, Brighton, England

Abstract

In this paper, a work in progress approach of layered motion interpolation method for designing realistic animation sequences for multiple goal-directed tasks is presented. The proposed solution is based on the ability to extract and synthesize different motions (in layers), while trying to efficiently reconstruct a natural-looking character's posture in real time. The proposed solution is examined for the case in which running, jumping, and reaching motions are combined. However, in addressing multiple goals fulfilled by a character in complex environments, as well as those involving complex motions, it is necessary to define the best way to handle and reconstruct the information from a motion capture database. Finally, because the character's posture should be as natural looking as possible, a simple centre of mass approach is proposed, to give desirable results at specific time steps.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

1. Introduction

Designing realistic animation sequences, in which multiple goals are fulfilled by a character, is a time-consuming process. The implementation of complex animation sequences that are not pre-computed by designers presents computational challenges. This is especially true when dealing with real-time implementations, such as those required for video gaming and engagement in virtual reality environments; for these applications, techniques that allow high-level controllability of the character in real time should be investigated. Such techniques must be able to reconstruct combinations of different kinds of motions in a natural-looking way. Techniques such as the layered interpolation approach that is used here can give developers the ability to design high-quality animated sequences.

The proposed solution is based on the ability to reconstruct new animation sequences by using different combinations of motions. Basically, the approach is based on the ability to separate and extract the motions for the goals that are defined by the user, and to associate those motions into specific body parts. Additionally, because the extracted motions are either not exactly the targeted ones, or because they influence the naturalness of the animation, the resulting motions must be edited. The editing process tries to synthesize a natural-looking posture of the character at specific time steps, based on a Centre of Mass (CoM) parameter of the articulated figure.

The scenario examined in this paper consists of a reaching motion while the character is simultaneously jumping to avoid an obstacle. Thus, the scenario involves the combined motions of reaching, jumping, and running. The real-time implementation of this approach is based on two analytical steps. The first step addresses the transition between the similar motions of running and jumping; the second step addresses the reconstruction of the character's posture, using the CoM as parameter.

1.1. Multiple Goal-directed Tasks for Animated Characters

In the animation of multiple goal-directed tasks, the virtual character should follow more than one goal specified by the user. A typical example is the reconstruction of an animated sequence in which the character jumps and raises his hand while simultaneously reaching for a virtual object; this motion is reconstructed using two databases, one for various postures involved in reaching and one for the animation of jumping. This is a simple example in which, even if the lo-

DOI: 10.2312/PE/vriphys/vriphys12/079-085

comotion of the character is not updated, there are computational issues related to the transition from one motion to another motion.

Addressing more complex situations in which the character is running and jumping to avoid an obstacle, while simultaneously raising his hand to reach for an object, seems to require more sophisticated animation parameters. In this case, the designer must consider three different motion capture databases: one for reaching, one for jumping, and one for running sequences. The combination of these different tasks appears to be complex because it is necessary to establish multiple criteria for coordinating the positions and orientations of the multiple body parts, which are all necessary components of the final animated scene. The required decisions are quite complex, even if they appear to be simple, because first, it is necessary to retarget some of the goals in real time (e.g., the hand position necessary to exactly reach the target position), and second, because the character posture during the animation process may be out of balance with respect to the CoM, thus resulting in a non-natural looking animation.

2. Background

Motion editing is a technique in which different motions are rearranged or blended to produce a new animation sequence. When performed under certain constraints, motion editing can be a complex process of designing animation sequences based on existing scenes that are stored in a database; the motion data is then edited to meet the conditions and constraints of the particular goals or situation. The solution proposed in this paper attempts to fill a gap in currently existing motion editing techniques; we present a method for synthesizing new animation sequences based on a combination of different motions, and using a postural control optimization framework.

While motion editing has been extensively investigated, the research community has not deeply examined the motion editing of multiple goal-directed tasks, despite the face that significant research has been published in recent years regarding real-time motion editing processes. Similar to this work, Safonova and Hodgins [SH07] attempted to reconstruct motion graphs to design realistic transitions between different motions. In the same manner, Feng et al. [FXS12] presented an approach for skilfully animating characters with the abilities to interact with objects in the virtual environment; their approach combined locomotion, path finding, object manipulation, and gaze. Additionally, Oshita [Osh08] proposed a system for designing long motion sequences by combining a number of elementary motion clips, using four different motion editing techniques and attempting to automatically generate the final motion. Furthermore, Li et al. [LWS02] proposed a solution in which the data were divided into motion textons; each texton was modelled as a linear dynamic system, and motions were synthesized by considering the likelihood of switching from one texton to the next. These approaches, even if they can produce desirable results in terms of the transitions between the different motions, are not able to blend contemporaneous or simultaneously occurring motions. Other approaches, such as that of Raibert and Hodgins [RH91], provide a control mechanism to generate a character's locomotion, using a simplified physical model. Yin et al. [YLvdP07] proposed a physics-based controller that can be constructed manually and can be trained with motion capture data. However, these methods, even if they do produce desirable results, are computationally expensive, which prohibits their implementation in real-time applications.

Motion editing processes for the animation of reaching tasks have been a common subject of research during the past several years. Briefly, the most common approach for motion reconstruction involves a combination of extracting postures from the database while applying Inverse Kinematic (IK) solutions to produce the desirable result. Kallmann [Kal08] proposed an IK analytical method for reaching tasks that directly solves for joints parameterized by swing and twist decomposition. Huang et al. [HKT10] proposed a hierarchical control strategy for animating the virtual character by using an analytical IK solution to control the hand, and by providing fast and accurate control of the end effector. Finally, Lv et al. [LZX*11] proposed a framework for reaching tasks using biomechanical strategies to guarantee the naturalness of the reaching motion, while extending the reachable space to enrich the flexibility of the character's behaviour. These approaches, even if they provide desirable results, are based on animated motions involving only single tasks, such as reaching, that already exist in the database.

Many previous research studies have yielded desirable results in the reconstruction of characters' postures. In one interesting approach, Raunhardt and Boulic [RB09] proposed a hybrid posture control system using a combination of goal-directed features and data-driven models. Their method benefits from the natural flow of movements provided by motion capture data, while applying a prioritized IK solution. Similarly, Carvalho et al. [CBT09] proposed an off-line method based on interconnected linear motion models, which allows for interactive motion editing. These dependent linear motion models allow the construction of a framework capable of solving a constraint-based optimization problem within the latent space. This approach, while possessing the ability to edit different motions, is limited to off-line calculations, as well as the ability to reconstruct only similar motions. Similar to the proposed solution that is presented within this paper, O'Brien et al. [ODC11] proposed a method for handling character's posture based on points, formulating a spacetime optimisation problem that solves altered character motion by using a set of constraints. Boulic et al. [BMT94] proposed a general Inverse Kinetics approach, for controlling the CoM, where the constraint related to the position of the CoM is treated as in any other task; their solutions are solved dif-

ferentially, using a special-purpose Jacobian matrix that relates differential changes in the joint coordinates to differential changes in the Cartesian coordinates of the CoM. Additionally, another solution that uses the CoM proposed Shin et al. [SKG03]. This solution uses virtual character Centre of Mass in order to estimate a physically valid motion by using existing motion editing techniques. Thus each edited motion can be described as a physically valid. Although, the disadvantage that this method has, is based on the ability of constructing a physically valid framework based on synthesised partial motion rather than on existing motions. Finally, Wang and Xia [WX11] used layered interpolation as a technique for providing high controllability over the character, and proposed a model for performance animation using interpolation of partial motions in limb subspace, and then animation of the final natural posture by compensation for the motion correlation between limbs. This solution inspired us to work with multiple goaldirected tasks, separating the character's body into different parts, while trying to handle and synthesize motions from different kinds of databases; this is the approach presented in this paper.

3. Overview

This section describes the procedure for reconstructing a natural-looking animation sequence of multiple goal-directed tasks. Specifically, the registration and extraction of motion capture data for multiple specified goals, as well as the synthesis of the final result, are presented. In this analysis, the motion capture database, provided by Carnegie Mellon University Graphics Lab [CMU], consists of 2500 frames of reaching tasks, 500 frames of running tasks, and 800 frames of jumping tasks. The running and jumping motions were pre-processed to be able to work in a cyclic perspective. Finally, as already mentioned, the examined scenario involved a character raising the hand to reach an object in 3D space, while jumping to avoid an obstacle. Thus, because the procedure consists of different tasks and different motions, it is necessary to divide the problem into the following steps (see Figure 1). The first and the last steps are running procedures, the second and fourth steps are preparation procedures, and the third step is an action procedure.

The division of the goal into these different steps provides the ability to handle and synthesize extracted motions separately. Moreover, the separation of the entire procedure into different steps allows for the implementation of different techniques to obtain the desired final result. Finally, because the basic technique for reconstructing the motions is based on a layered interpolation method, it is necessary that the character's body be divided into three parts (see Figure 2). The first part is the lower body, which is responsible for the locomotion sequences (running and jumping); the second part is the upper body, except for the left hand which is responsible for the reaching motion, as well for upper body

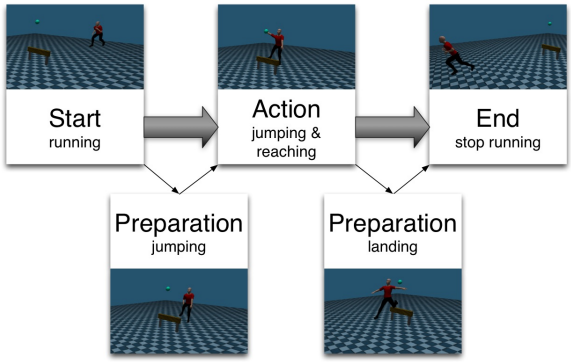

Figure 1: *Separation of the procedure into different steps for motion editing.*

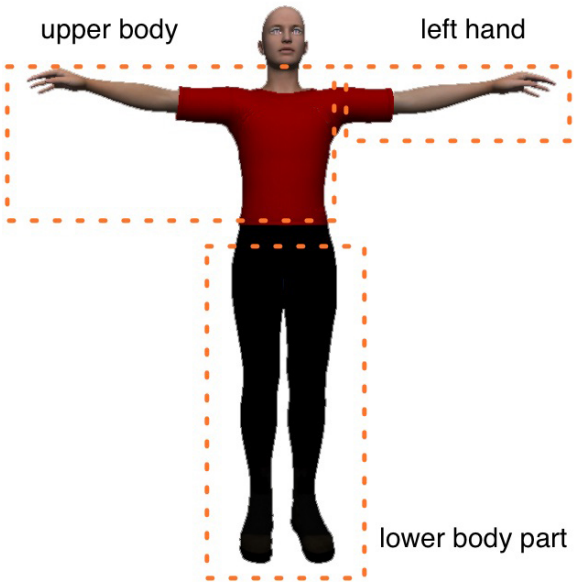

Figure 2: *Different layers of the character's body.*

posture; the third is the left hand, which is responsible for balancing the character's body.

3.1. Handling Motion Capture Data

Addressing different kinds of motions requires that the motions in the database are registered to one another. Because the motions consist of locomotion and non-locomotion data, and to avoid incorrect calculations and transitions, the registration process is divided in two parts. First, the locomotion is updated with the basic motions, which are running and jumping motions; these are stored using only the global root position of the character and local positions for all body

parts. Thus the registered data have the form:

$$\Psi_n^j = \{P^{root}, Q^{root}, R_1^j, ..., R_n^j\} \quad (1)$$

where P^{root} and Q^{root} are the root positions and orientations and R_n^j is the orientation of joint j in the $n-th$ frame. Aditionally, to deal with non-locomotion sequences, the postures are stored, along with the local positions of the parts, giving the global root position and the root orientation for the character's running motion.

Additionally, while propagating though the virtual character's body hierarchy as well as through the motion hierarchy, it is possible to construct new motions. Those motion results from the division of the human body hierarchy into lower motion layers. Hence, in the proposed approach, where motions are divided in three different body parts, as mentioned earlier, it is possible to construct a general model while propagating through the motion and character's body hierarchy. Thus, the motion model that is constructed for the first division, between the upper- and lower-body part, has the form of:

$$M = \{P^{root}, Q^{root}, M^{upperBody}, M^{lowerBody}\} \quad (2)$$

where $M^{upperBody}$ and $M^{lowerBody}$ are the first hierarchical division of the motion respectively. Additionally, while propagating again through the hierarchy of upper-body motion, the new motion model that is generated has the form of:

$$M^{upperBody} = \{M^{mainUpperBody}, M^{leftHand}\} \quad (3)$$

Finally, for the generation process of the fional motion, the synthesised motion could have the form as it is presented by the following function:

$$M^{synthesised} = M^{locomotion}(M^{upperBody}(M^{leftHand})) \quad (4)$$

where each higher-level layer motion is an internal parameter of the lower-layer motion sequence.

3.2. Motion Capture Data Extraction

To generate the animation sequence, it is necessary to extract the most suitable motions stored in the database. The extraction of reference motions occurs in two steps. In the first, the problem is approached by representing the desired motion as an interpolation of two motions (running and jumping). In practice, each motion has a successor state (e.g., the left foot on the ground), as demonstrated by Safonova and Hodgins [SH07]. To extract the jumping motion while the character is running, it is necessary to match the successor state (see Figure 3) of running (S_{run}^n) with the successor state of jumping (S_{jump}^n), to ensure continuity between the states of the two motions; this eliminates discontinuities in the final result. Thus, in the first step, the goal to be fulfilled by the character is extracted by unrolling and augmenting the procedure in the environment, using the global root position (P^{root}). Additionally, it is necessary that the motion graph constructed for the procedure is represented by a straight line between the starting and ending positions, and that the character and the obstacle are on the line. In the second step, the extracted new motion is interpolated again to register the reaching postures. For the second interpolation, it is not necessary to account for successor states, as the only parameter that influences this task is the distance between the hand and the object. Finally, it is important to note that in this approach, the obstacle is always the goal with the higher priority, with lower priority given to the reaching task.

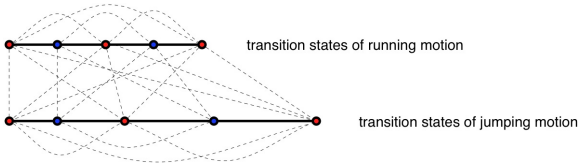

Figure 3: *Motion transition states based on contact of the foot with the ground. Red dots represent the position of the right foot, and blue dots the position of the left foot. The distance between two dots is the horizontal distance between the foot and the root position.*

3.3. Synthesizing the Animation

For animating different motions, it is necessary to define the transition process between two or more motions, especially for those that are related, as in the transition between running and jumping and vice versa. After having registered the suitable motions, the transition can be calculated by applying blendweight techniques directly to the graphs. In the proposed solution, the blend weight technique was implemented by extracting eleven intermediary postures ranging from 0 to 1. The blend weight step can be described as:

$$P_i = \sum_{i=-5}^{5} P_i^{run} * w_1 + P_i^{jump} * (1 - w_1) \quad (5)$$

where and are, respectively, the running and jumping postures for the extracted animation sequences at the frame. To avoid discontinuities, five frames were chosen before and after the transition, thus ensuring that the transition would be normalized (see Figure 4).

Having designed the transition between the two motions, the next step involves adding the reaching task into the sequence. Thus, having attached the database with the reaching postures, the most suitable of those can be extracted for use in the overall procedure. Because the motion capture database is limited to movements, finding the most suitable motion does not involve fulfilling the desired criteria for reaching the object in 3D space. The upper body motion and the right hand motion can be calculated using barycentric coordinates, determined according to Cramer's rule and the triangle generated by the three most suitable positions

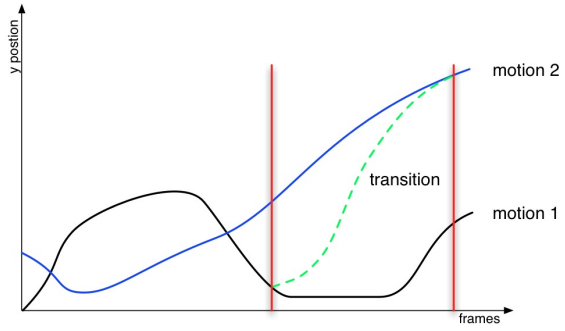

Figure 4: *Blendweight transition.*

of the right hand. The triangle, consisting of vertices v_α, v_β and v_γ and associated with the most related existing positions (p_1, p_2, p_3), is generated. Then, the values for each position are blended to produce the final correct position of the right hand. The barycentric coordinates can be calculated as follows:

$$v_\alpha = \begin{bmatrix} p & p_2 & p_3 \\ 1 & 1 & 1 \end{bmatrix} \setminus \begin{bmatrix} p_1 & p_2 & p_3 \\ 1 & 1 & 1 \end{bmatrix} \quad (6)$$

$$v_\beta = \begin{bmatrix} p_1 & p & p_3 \\ 1 & 1 & 1 \end{bmatrix} \setminus \begin{bmatrix} p_1 & p_2 & p_3 \\ 1 & 1 & 1 \end{bmatrix} \quad (7)$$

$$v_\gamma = 1 - v_\alpha - v_\beta \quad (8)$$

Having calculated the barycentric coordinates by the vertices of the triangle, the final right hand motion can be calculated by:

$$P'_{hand} = v_\alpha * P^i_{hand} + v_\beta * P^i_{hand} + v_\gamma * P^i_{hand} \quad (9)$$

where P^i_{hand} is the reference posture of the $i-th$ example $(i = [1, ..., n])$ of the hand extracted from the database. Finally, it should be noted that the position of the object must be inside the reaching area of the character's hand motions.

3.4. Posture Reconstruction

Having analysed the process for synthesizing the animation sequence, the next step is to reconstruct the character's posture. This is an important aspect of the proposed procedure because the correct posture determines how realistic the final result is. The character's CoM is an important parameter for the determination of the correct final posture. Especially in the case of a rigid body, as in a 3D character, the CoM is fixed in relation to the body and does not necessarily coincide with the geometric centre. Although, various other methodologies have been proposed during the past years using the CoM for the postural control of the virtual character like the methodology proposed by Shin et al. [SKG03] where the mass distribution consistent is estimated with reference motions, known to be physically correct. Another similar solution proposed by O'Brien et al. [ODC11] where the CoM of the virtual character is used to represent the poses of the character at sampled frames of the animation. Those, techniques even if can provide desirable results, seems to be limited especially in the case where the generated motion is not resulting from a single motion sequence but from a combination of various integrated motions.

In the proposed solution, the CoM, which is extracted from the motion capture data, must be calculated to reconstruct the character's posture. In the first step for the calculation of the CoM for the existing motion sequence is calculated according to:

$$P^t_{CoM} = (\sum_{i=-1}^{n} P^t_i * m_i)/M \quad (10)$$

where P^t_{CoM} is the Vector3 position of the CoM in the $t-th$ animation frame, $i = 1, ..., n$ are the total number of joints, m_i is the mass of the $i-th$ joint, M is the total mass of character's body, and P^t_i is the Vector3 position of the $i-th$ joint.

Assuming that the motion capture data can provide a natural looking posture, the calculated position of the CoM at each frame of the animation sequence should be at P^t_{CoM}. As mentioned, each body layer is responsible for different tasks. Thus, the character's left hand is responsible for keeping the balance of the character as well as for positioning the CoM as correctly as possible. Knowing the P^t_{CoM}, and solving the CoM equation for the unknown P'_{hand} parameter, the new position of the left hand can be retrieved (see Figure 5).

Figure 5: *Initial reaching posture (left), initial jumping posture (middle), and reconstructed final posture (right) based on centre of mass (CoM) and inverse kinematic (IK) calculations.*

Because the desirable position of the hand ($P'_h and$) is observed to be outside the goal hull of the reference posture, an Inverse Kinematics (IK) solution based on a Jacobian pseudo-inverse is applied to the character limb to deform the blended motion for more accurate grasping. Given the desirable end effector ($P'_h and$), and reference joint angles $\theta = [\theta^{arm}_i, \theta^{elbow}_i]$, the IK method solves for the joint angles

in $\Delta\vartheta$ increments. The resulting left hand pose that has as its end effector the constraint of the desirable $P'_h and$ position should satisfy the pose constraint ϑ as closely as possible. For any given joint configuration defined by ϑ, the Jacobian matrix J is a linear mapping that transforms the differential changes $\Delta\vartheta$ to the end effector coordinates of the hand ($\Delta P'_{hand}$), where the i, j elements of the Jacobian matrix can be calculated from the partial derivative of the $j-th$ component of the joint angles and the $i-th$ component of the end effector coordinates, as $dY = (\vartheta P^i_{hand}/\vartheta\theta_j) * d\theta$. Thus, the relationship between $\Delta\theta$ and $\Delta P'_{hand}$ is approached using $\Delta P'_{hand} = J(\vartheta_i)\Delta\vartheta$. Finally, at each frame step, the IK joint angle increment $\Delta\vartheta$ is calculated as:

$$\Delta\vartheta = J^+\Delta P^i_{hand} + (I - J^+J)\Delta x \tag{11}$$

where $J^+ = J^T(JJ^T)^{-1}$ is the Jacobian pseudoinverse, $\Delta P'_{hand}$ is the offset of the current end effector's coordinates from the P'_{hand} target coordinates, and Δx is the desired joint angle increment toward the target pose for $x = d\vartheta$.

3.5. Time Alignment

Having synthesized and reconstructed the sequence, it is necessary to apply a time alignment function that has the ability to construct the right timings, especially for the reaching motion and the motion back to the initial position. Assuming that the extracted reaching task has a duration of x frames, and the character reaches the object in frame t_x of the sequence $T_{sequence} = [t_1, t_2, ..., t_n]$, where t_n is the number of frames, the procedure for the reaching task would start at the $t_x - x$ frame (see Figure 6). In this way, it is ensured that the character can reach the object with his hand at the correct time. Using the same procedure, the motion back to the initial position starts at frame x and has the duration t^{back}_x. On the other hand, the timings for the transition between the running and jumping sequences are constrained only by the time frame step of the running sequence; this is chosen because it is necessary to normalize both motions to keep a static time transition.

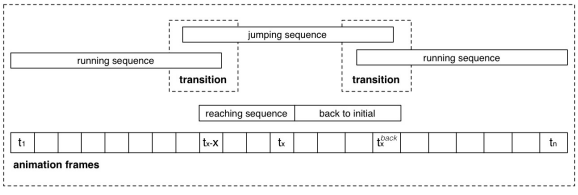

Figure 6: *Time alignment example for different motions.*

4. Results

To evaluate the proposed approach to animation, an application was developed in the Unity3D game engine. In this application, the character tries to reach a location in 3D space specified by the user area using the right hand, while jumping and avoiding an obstacle. The evaluation tests were performed on a system with a 2.2 GHz Intel i7 processor with 8 GB of RAM. The maximum performance was chosen in the Unity3D game engine, which gives a maximum frame rate of 250 fps. With a 3D mesh of 2000 polygons, the implementation requires 100 ms for the registration process; the real-time implementation of the algorithms for the animation process doesn't exceed the 120 fps. To ensure that the approach works properly in various situations, obstacles with different heights were used, and the reaching target was placed in various areas in the 3D space. Finally, an interesting result was observed when solving the IK calculation for the left hand; because the transition between the two positions of the hand evolves gradually, the animation results show no discontinuities, thus resulting in a natural-looking transition.

4.1. Conclusion and Future Work

The high level controllability of virtual characters, and the capacity for real-time motion editing and synthesis involving the blending of multiple motions to design new animation sequences, are promising areas of research in computer animation. Synthesis of novel animation sequences, where different motions and motion capture databases can be handled and blended to provide complex animation sequences, is important for designing realistic animation sequences for real-time applications. This paper presents a method for addressing different parts of a character's body while reconstructing an animation sequence where multiple goals are fulfilled by the character. Furthermore, a solution for reconstructing the character's posture while maintaining the character's CoM in the correct position is presented. Finally, experiments were performed that demonstrate that the proposed solution can characterize the naturalness of movements.

Even if the proposed approach works well in the specific situation that was described, limitations may affect the ability of the approach to achieve more complex animations, e.g., in which the character's locomotion path is cuvilinear, or the character is presented with an obstacle that cannot be cleared by jumping. Based on these limitations, a possible extension of the approach would be the ability to design more sophisticated animations involving irregular paths. This can give designers and novices the ability to produce long realistic sequences in which the character may take complex paths through the environment, as opposed to moving in a straight line between two points. Another possible extension would be the ability to add obstacles which the character cannot avoid or clear by jumping. In such situations, the character could have the ability to pass an obstacle by changing directions. Finally, it should be noted that the generation of movements in this approach is based on the ability to extract reference data, which is required to design new animation sequences. Issues such as these, which limit

or are limited by the motion capture database, should be investigated and extended by adding the ability to calculate new motions based on thorough IK calculations, as well as the temporal dynamics of the character.

Acknowledgement

The authors would like to thank the anonymous reviewers for their helpful and valid comments.

References

[BMT94] BOULIC R., MAS R., THALMANN D.: Inverse kinetics for center of mass position control and posture optimization. In *Proc. of European Workshop on Combined Real and Synthetic Image Processing for Broadcast and Video Production* (1994). 2

[CBT09] CARVALHO S. R., BOULIC R., THALMANN D.: Motion pattern encapsulation for data-driven constraint-based motion editing. In *Proc. of Motion in Games* (2009). 2

[CMU] Carnegie mellon university, motion capture database. 3

[FXS12] FENG A. W., XU Y., SHAPIRO A.: An example-based motion synthesis technique for locomotion and object manipulation. In *Proc. of ACM SIGRAPH Symposium of Interactive 3D Graphics and Games* (2012), pp. 95–102. 2

[HKT10] HUANG W., KAPADIA M., TERZOPOULOS D.: Full body hybrid motor control for reaching. In *Proc. of Motion in Games* (2010). 2

[Kal08] KALLMANN M.: Analytical inverse kinematics with body posture control. *Computer Animation and Virtual Worlds 19*, 2 (2008), 79–91. 2

[LWS02] LI Y., WANG T., SHUM H.-Y.: Motion texture: A two-level statistical model for character motion synthesis. *ACM Transactions on Graphics 21*, 3 (2002), 465–472. 2

[LZX*11] LV P., ZHANG M., XU M., LI H., ZHU P., PAN Z.: Biomechanics-based reaching optimization. *The Visual Computer 27*, 6-8 (2011). 2

[ODC11] O'BRIEN C., DINGLIANA J., COLLINS S.: Spacetime vertex constraints for dynamically-based adaptation of motion-captured animation. In *Proc. of ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (2011), pp. 277–286. 2, 5

[Osh08] OSHITA M.: Smart motion synthesis. *Computer Graphics Forum 27*, 7 (2008), 1909–1918. 2

[RB09] RAUNHARDT D., BOULIC R.: Motion constraint. *The Visual Computer 25* (2009). 2

[RH91] RAIBERT M. H., HODGINS J. K.: Animation of dynamic legged locomotion. In *In Proc. of the 18th annual conference on computer graphics and interactive techniques* (1991), pp. 349–358. 2

[SH07] SAFONOVA A., HODGINS J. K.: Construction and optimal search of interpolated motion graphs. *ACM Transaction on Graphics 26*, 3 (2007). 2, 4

[SKG03] SHIN H. J., KOVAR L., GLEICHER M.: Physical touch-up of human motions. In *Proc. of 11th Pacific Conference on Computer Graphics and Applications* (2003), pp. 194–203. 3, 5

[WX11] WANG J., XIA S.: Layered interpolation for interactive avatar control. In *Proc. of the 10th International Conference on Virtual Reality Continuum and Its Applications in Industry* (2011), pp. 49–58. 3

[YLvdP07] YIN K. K., LOKEN K., VAN DE PANN M.: Simbicon: Simple biped locomotion control. *ACM Transactions on Graphics 26*, 3 (2007). 2