Enriching Animation Databases

Amr Ahmed[†], Farzin Mokhtarian, and Adrian Hilton

Department of Electronic Engineering, University of Surrey, UK.

Abstract

The paper presents a framework for enriching an existing basic animation databse by modifying existing motions to generate a variety of new motions. Intuitively controlled motion varities are generated on demand which reduces the storage requirements, avoids recapturing missing variations, and provides greater flexibility and utilisation of existing clips. An improved root-trajectory blending is introduced that overcomes the limitations of ordinary blending (including flipping of the root trajectory when blending motions of opposite path-curvature or opposite directions). A novel synchronised mirror technique is also introduced.

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

1. Introduction

Motion capture systems produce realistic animation. However, the capturing process is expensive, time-consuming, and the facilities may not be available or affordable for many users. The idea of the proposed framewaork is to utilise existing basic motion clips to enrich the animation database and avoid the need for re-capturing variations of existing clips.

The paper is focused on the proposed work and key related work is only referenced wherever appropriate, due to limited space. Section 2 presents an overview of the proposed framework and how existing clips can be utilised. Our *improved root – trajectory blending* and *synchronised mirror* techniques are presented in sections 3 and 4 respectively. Finally the paper is concluded in section 5

2. Framework for Enriching Animation Database

The proposed framewaork consists of constructing a layered encapsulated animation database as depicted in figure 1. At the bottom layer L_r, only real animation clips exists. The middle layer L_m contains modifiers that generate new motion clips based on one or more real animation clip from L_r. The top layer L_v represents the virtual database that is available to user, which includes real and modified animation clips.

Modifiers should be selected, designed, and implemented to be as simple and computationally efficient as possible.

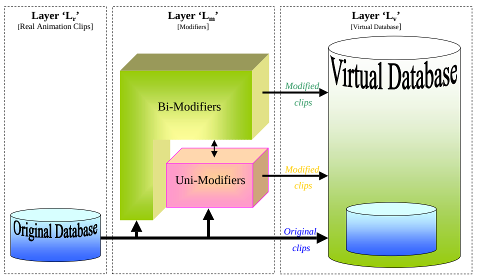


Figure 1: *The Layered Animation Database diagram*

However, the exact action of the modifier can be controlled by some parameters or attributes.

Depending on the number of animation clips the modifier is accepting as inputs, there are uni-modifiers, bi-modifiers, and multi-modifiers. Uni-modifiers are mainly basic animation processing techniques, such as "*Mirror*" and "*RotatePath*" (figure 2), that are applied on individual animation clips. Bi-modifiers use two input motions to generate a new one. Examples include speed, slope and path curvature control as depicted in figure 2. Multi-modifiers (or modules) are employing more than two input motions to produce new motion and can control more than one parameter simulteneously. This analogous, to some extent, to the motion models introduced in [Gra00].

The above modifiers use the parametric approach introduced in [AMH01]. Controlled by intuitive high-level motion paramters, from two or more given motions, the modifiers can produce varieties of motions which can be im-

[†] A.Ahmed@eim.surrey.ac.uk

practical to capture and store. Moreover, they are generated on demand (transparent to the user) and need not be saved permanently, which reduces storage requirements. The proposed layered database framework is expandable by defining new modifiers, cascading existing modifiers, or inserting real clips.

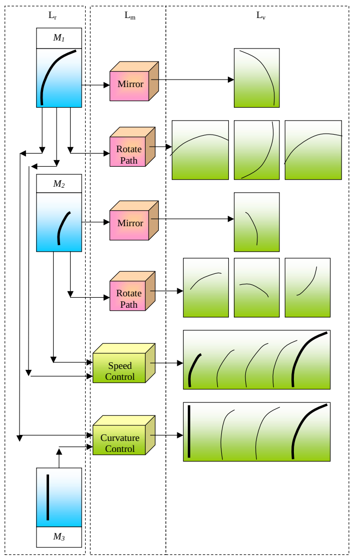


Figure 2: *Examples of enriching animation database contents*

In the light of the above framework, two novel contributions are introduced in the rest of this paper that allow for better utilisation of existing clips and greater reduction in the data requirements.

3. Improved Root-Trajectory Blending

It is important to handle the root joint carefully because it influences the whole skeleton. Figure 3 explains two principal limitations of existing root blending which we successfully overcomed in our proposed technique.

In figure 3.a, the magnitude of the blended root translation $\overrightarrow{C}$ is affected by the angle θ between the input motion directions $\overrightarrow{A}$ and $\overrightarrow{B}$. As a result, the overall motion parameters, such as speed, will be affected which reduces the accuracy of synthesised animation in terms of the desired parameter. Moreover, artifacts, such as foot slipping, will increase with the angle θ. The other problem, explained in figure 3.b, is the sudden flipping of the root trajectory when the angle θ changes around 180^o (i.e from $\theta < 180^o$ to $\theta > 180^o$). An example of this problem is depicted in figure 4 where two curved-path motions are blended together. The blended motion suddenly turns back.

The sudden flip in root position trajectory is inherited from vector analysis as depicted in figure 3.b. The resulting

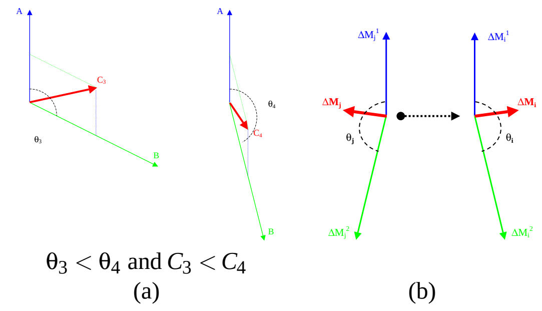


Figure 3: *Explanation of the Root Blending problems. a) Root translation magnitude is affected by the difference in direction between input motions. b) Root trajectory flips when moving from frame j (left) to frame i (right) with $i > j$ and $\theta_j < 180^o < \theta_i$.*

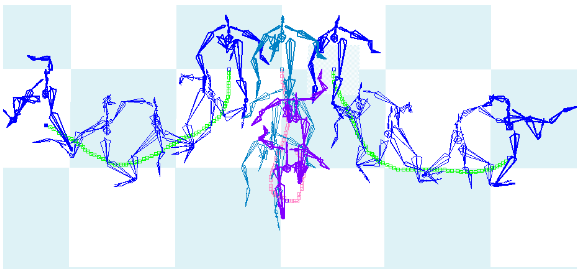


Figure 4: *The Synthesised root trajectory (middle) is suddenly flip to the opposite direction after the difference in direction of input motions exceeds 180^o.*

vector is always on the side of the smallest angle which suddenly changes around 180^o. Hence the sudden flip of root-position trajectory.

In addition to the above, the sudden flip in root orientation is also attributed to the fact that a single orientation can be achieved by two different rotations (e.g., a rotation of 100^o will lead to the same orientation of a rotation by -260^o).

In our improved blending technique, the resulting root trajectories are built by blending the incremental changes in input root trajectories. This incremental approach allows us to preserve the magnitude of root translations and prevent the sudden flipping regardless of the global root position and orientation of the input motions.

The incremental blending of the root orientation , with blending factor α, is achieved on the following steps (using quaternion interpolation):

- The initial root orientation, at frame 1, is calculated by direct blending of root orientation of input motions.
- For each frame i, calculate the accumulated root orientation R^i such as: $R^i = R^{i-1} + \Delta R^i$

where $\quad \Delta R^i = \alpha \Delta R^i_{M_1} + (1-\alpha)\Delta R^i_{M_2}$

and $\quad \Delta R^i_{M_j} = R^{i-1}_{M_j} - R^i_{M_j}; \qquad$ for $j = \{1,2\}$

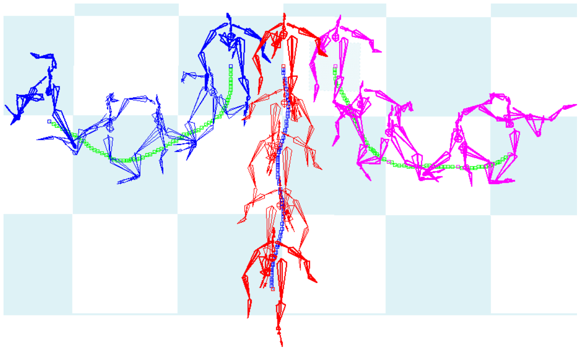
The incremental blending of the root translation, with blending factor α, is achieved on the following steps:

- The initial root positions, at frames 1 and 2, are calculated by direct blending of root positions of input motions.
- For each frame $i > 2$,
  - For each input motion, calculate the *magnitude* of incremental root translation.
    $\Delta T^i = \alpha \Delta T^i_1 + (1-\alpha)\Delta T^i_2$
    where $\Delta T^i_j = \sqrt{||X^i_j - X^{i-1}_j||^2} \qquad$ for $j = \{1,2\}$.
  - Calculate the change in direction of root position trajectory for each input motion:
    $\Delta\theta^i_j = \theta^{i-1}_j - \theta^i_j \qquad$ for $j = \{1,2\}$;
    where $\theta^i_j = \tan^{-1}(\frac{Z^i_j - Z^{i-1}_j}{X^i_j - X^{i-1}_j})$.
  - Check for sudden change in direction due to different representations of the same orientation (e.g. $\pm 180^o$) and compensate it:
    if $(\Delta\theta^i_j > 180^o) \qquad \Delta\theta^i_j = \Delta\theta^i_j - 360^o$
    if $(\Delta\theta^i_j < 180^o) \qquad \Delta\theta^i_j = \Delta\theta^i_j + 360^o$
  - Calculate the blended incremental change in direction of the root trajectory: $\qquad \Delta\theta^i = \alpha\Delta\theta^i_1 + (1-\alpha)\Delta\theta^i_2$
  - Calculate the global *direction* of the incremental root translation, relative to the global direction at last blended frame $i-1$: $\qquad \theta^i = \theta^{i-1} + \Delta\theta^i$
  - Calculate the global root position vector $\overrightarrow{P^i}$ by accumulating the incremental translation to the previous global position: $\qquad \overrightarrow{P^i} = \overrightarrow{P}^{i-1} + \Delta\overrightarrow{P^i}(\Delta T^i, \theta^i)$

Figure 5 presents the result of applying the *improved root trajectory blending* on the example of figure 4. We can see that the improved blending technique has successfully avoided the problem of sudden flip of the root trajectory.

From the example of figure 5, by allowing successful blend between motions of opposite path-curvature (and opposite directions such as straight and turn-back motions), the proposed improvement has allowed for syntheis of inbetween motions which were not achievable using the ordinary blend. Moreover, motion parameters, that depends on root translation such as speed, are preserved by maintaining the correct root translation within the proposed improvement as depicted in figure 6.a. The data requirements can be reduced further using the technique proposed in section 4.

To the best of our knowledge, the only related work to this section is Kovar et. al [KC03]. Their proposed solution is to align corresponding input motion frames, by applying 2D transformation in the floor-plane. The aligned frames are blended to produce the resulting frame posture and the 2D transformations are blended to produce the 2D transformation required to return the resulting frame to its global orientation. However, the rotation part of the 2D transformation can be done in both directions, clockwise and counterclockwise and there is no criteria for selecting the appropriate one for correct results in different situations. In our solution, blending the changes of input orientation avoids the large difference in direction between inputs. The history of one frame provides a clue for the change of direction of each input motion and allows for preventing sudden flip in root trajectory that may results from different representation of orientation. Also, instead of storing 2D transformation between every pair of corresponding frames for every pair of motions [KC03], we can save only the changes of direction for each individual motion. This reduces the storage requirements from $O(\frac{m}{2}(m-1)n)$ to $O(mn)$, as shown in figure 6.b, where m is the number of motion clips and n is the number of corresponding frame pairs.



Original clip (left) and *Synchronised Mirror* clip (right)

Figure 5: *The synthesised root trajectory (middle) has no sudden flipping any more after using the improved blending technique.*

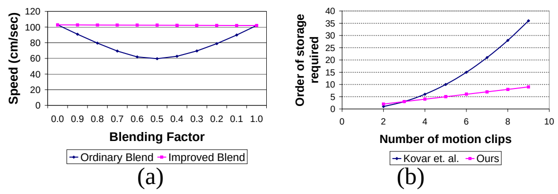


Figure 6: *Benefits of improved blending. a) Motion parameter (such as speed) is preserved. b) Storage requirements reduced.*

4. Synchronised Mirror

Traditionally, mirroring of human animation is performed on the sagittal plane defined at the initial frame; the plane dividing the human body vertically into two symmetrical halves, right and left. Traditional mirroring can double the number of clips in an animation database. However, the limitation is that the mirrored clips are out of phase with their original clips. For example, if the original clip is a 'walk' in a clockwise curved path starting with left foot ahead, its mirror will be a 'walk' in a counter-clockwise curved path with the right foot ahead (instead of the left foot). This means that the key events of the motion are out of phase and blending between them will produce invalid motion (see accompanied animation examples).

Our novel '*SyncMirror*' technique achieves the goal of mirroring while producing motion that is in-phase with original motion. Given a curved-path motion cycle, say clockwise curvature M_{cw}, its synchronised mirror cycle M_{ccw} can be obtained by applying the algorithm in figure 7 which is also explained by the diagram in figure 8. The algorithm mirrors a sequence of two cycles, constructed from the given cycle, which allows selecting the synchronised part of the mirrored cycle which is in phase with the given cycle.

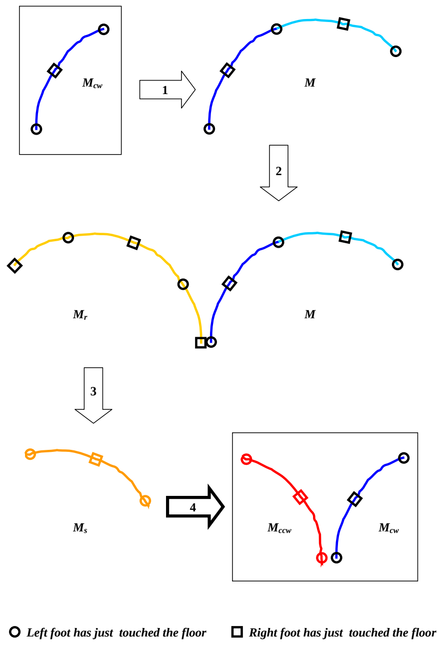
```
Input:
    1. One motion cycle '$M_{cw}$' on curved-path. (Cycle
       is assumed to have matched boundaries [AMH03].)
    2. List of key-event times [$T^{M_{cw}}$].
Output:
    1. Mirrored and Synchronised motion cycle '$M_{ccw}$'
Procedure:
    1. Concatenate $M_{cw}$ to its self
           $M = Append(M_{cw}, M_{cw})$;
    2. Mirror the 2 cycles clip
           $M_r = Mirror(M)$;
    3. Get the synchronised 1 cycle from '$M_r$'
           $M_s = M_r[K_i : K_{i+n}]$;
    where $K_i$ is the Key-event that in-phase with $T^{M_{cw}}(0)$
    and $n$ is one less the number of key-events per cycle.
    4. Translate and rotate $M_s$ to align with $M_{cw}$
           $M_{ccw} = TR * M_s$;
    where $TR$ is the appropriate transformation matrix.
```

Figure 7: *Synchronised Mirror algorithm, 'SyncMirror'.*

The proposed *SyncMirror* algorithm can reduce data requirements to half. In figure 5, the only given clip is the clockwise curved-path motion (left). The counter-clockwise curved-path (right) motion is the synchronised mirror of the given motion. Inbetween motions, such as the straight-path motion (middle), are generated by blending both curved motions as discussed in section 3.

5. Conclusion

The proposed enriching framework can reduce the data requirements by slightly modifying existing motions to generate variety of motions. Intuitively controlled motion varities are generated on demand which reduces the storage requirements, avoid recapturing missing variations, and provide more flexibility. The *improved root − trajectory blending* and the novel *synchronised mirror* techniques allow for greater utilisation of existing clips as depicted in figure 5. Using the proposed techniques, all clips in the L_v layer of figure 2 can be generated from only one clip such as M_1. Animation examples are enclosed in the accompanied archive.



M_{cw}
1
M
2
M_r
M
3
M_s
4
M_{ccw}
M_{cw}
○ Left foot has just touched the floor
□ Right foot has just touched the floor

Figure 8: *This diagram explains the synchronised mirror algorithm, 'SyncMirror' of figure 7*

References

[AMH01] AHMED A., MOKHTARIAN F., HILTON A.: Parametric motion blending through wavelet analysis. In *Eurographics'01. Proceedings of Short Presentations* (2001), pp. 347–353. 1

[AMH03] AHMED A., MOKHTARIAN F., HILTON A.: Cyclification of human motion for animation synthesis. In *Eurographics'03. Proceedings of Short Presentations* (2003), pp. 193–200. 4

[Gra00] GRASSIA F. S.: *Believable Automatically Synthesized Motion by Knowledge-Enhanced Motion Transformation*. PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, 2000. 1

[KC03] KOVAR L., CLEICHER M.: Flexible automatic motion blending with registration curves. In *Proceedings of the SIGGRAPH Symposium on Computer Animation* (2003). 3