Motion Tweening for Skeletal Animation by Cardinal Spline

Murtaza Ali Khan[1] and Muhammad Sarfraz[2]

[1] Department of Computing Science
Royal University for Women
Bahrain
drkhanmurtaza@gmail.com
[2] Department of Information Science
Kuwait University
Kuwait
muhammad.sarfraz@ku.edu.kw

Abstract. Motion tweening (Inbetweening) is the process of generating intermediate frames between keyframes to create an illusion of motion. Motion tweening is a key process to generate computer based animations. This paper presents a simple and efficient method of tweening of motion data for skeletal animations. The proposed method generates smooth animation using cubic Cardinal spline. Keyframes are taken as control points and spline interpolation is performed to generate the in between frames smoothly. In order to facilitate the input, the keyframes can be manually specified or automatically detected from the motion data.

Keywords: keyframes, animation, motion, tweening, cardinal spline, interpolation, control points.

1 Introduction

Motion data is extensively used in video games, computer animation, and animated videos. Interestingly out of the three nominees for the 2006 Academy Award for Best Animated Feature, two of the nominees (Monster House and Happy Feet) used motion capture data [1]. The importance of using motion data in several fields is growing rapidly. This requires to investigate better methods to manipulate motion data. The aim of this research is to find a simple and efficient method of motion tweening using Cardinal spline. In our work, we focused on tweening of motion data that belongs to skeletal animations. Even though the method can be applied to other types of motion data as well.

Motion tweening uses keyframes with an interpolation scheme to construct new frames. This essentially requires a set of meaningful, high-level control knobs (control points) to the animator or system. Motion tweening must be efficient for use in a run-time environment. This paper describes a technique that is very efficient and works well in real-time for motion tweening using Cardinal spline.

A. Abd Manaf et al. (Eds.): ICIEIS 2011, Part IV, CCIS 254, pp. 179–188, 2011.

Spline and curve are widely used in computer-aided design and computer graphics because of the simplicity of their construction, accuracy of evaluation, and their capability to interpolate and approximate complex shapes [16], [3]. Spline interpolation is used to generate additional points given few sample points (control points/keypoints). Due to continuity of Cardinal spline, it is well suited for generating smooth animation. In addition to that implementation of Cardinal spline is simple and efficient and can be employed for real-time systems.

Organization of the rest of the paper is as follows. Related work is discussed in Section 2. Section 3 describes basic concepts and terminology about animation and motion data. Construction and interpolation of Cardinal spline is explained in Section 4. Section 5 is the most important section and it describes the details of motion tweening strategy of the proposed method. Simulation and results are presented in Section 6. Final concluding remarks are in Section 7.

2 Related Work

A good reference on modeling, capture, and animation of human motion is [15]. This book contains articles contributed by several researchers. General Cardinal spline interpolation schemes are investigated by many authors [13], [9], [4], [10]. A scheme to construct new motions from existing motions is proposed in [14]. This scheme used low order polynomials and B-splines functions to create the interpolation space between example motions. Our method does not generate new motion from existing motions but generates motions from keyframes. The method proposed in [12] is about geostatistics based motion interpolation. This method treats motion interpolations as statistical predictions of missing data in an arbitrarily definable parametric space. However, the method of [12] has limitations, since the number of necessary samples usually increases exponentially in the dimension of a control space. The work in [17] used a local spline keyframe interpolation method for generating animations, but the method is applicable to only 2D animations such as cartoons in comics. Our method is not restricted to 2D animations and can be used for generating both 2D and 3D animations. A method to generate a smooth morphing between the source and the target images by drawing input curves on the features of the objects is presented by [8]. This method uses Bézier curves and shape interpolation techniques . Our method uses Cardinal spline interpolation rather than Bézier curves and does not require to draw any input curve.

3 Basics of Animation and Motion Data

Following is the basic terminology used to describe different aspects of a animation, motion data and motion data file format.

- **Skeleton:** It is a whole character on which motion data is applied to create animation. Figure 1 shows a skeleton of human figure [5].
- **Bone:** A skeleton is comprised of a number of bones. Each bone represents the smallest segment within the motion that is subject to individual translation and/or rotation changes during the animation.

– **Channel or Degree of Freedom (DOF):** Each bone within a skeleton can be subject to translation, rotation and scale changes over the course of the animation, where each parameter is referred to as a channel or degree of freedom (DOF). The changes in the channel data over a period of time produce animation. Translation and scale changes are measured in unit of length, while rotational changes are measured in degrees or radians.
– **Frame/Frame-rate:** Every animation is comprised of a number of frames where for each frame the channel data for each bone is defined. Frame-rate refers to number of frames generated in one second. Generally frame rate varies between 24 to 60 frames per second (fps).

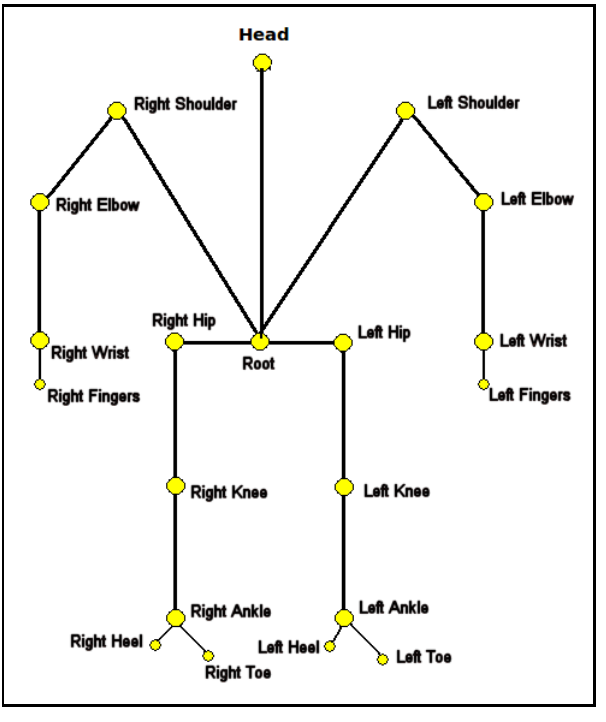

Fig. 1. A skeleton of human figure

We can plot the motion data of a joint for each channel separately such that frame numbers are along horizontal axis and translation or rotation values are along vertical axis. This kind of plot is called *motion curve* of *motion signal*. Figure 2 shows motion curve of *right-shoulder* joint (rotation along z-axis) in 153 frames.

Several file formats exist to store the motion data [11]. We used BVH (Bio Vision Hierarchy) file format. This format is simple and supported by many motion capture devices and animation programs. A BVH file has two sections, a header section which describes the initial hierarchy of the skeleton; and a data section which contains the motion data. BVH can store motion for a hierarchical skeleton. This means that the motion

of the child bone is directly dependent on the motion of the parent bone. Figure 3 shows hierarchy of joints for a segment of a skeleton. For example in Figure 3, *Hips* joint has children *LeftHip*, *RightHip*, and *LowBack*. *LeftHip* has a child joint *LeftKnee*. *LeftKnee* has a child joint *LeftHeel*, etc. Figure 4 shows how BVH file format stores hierarchical skeleton. The motion section of a BVH file contains number of frames, frame rate and the actual motion data. Each line is one sample (one frame) of motion data.

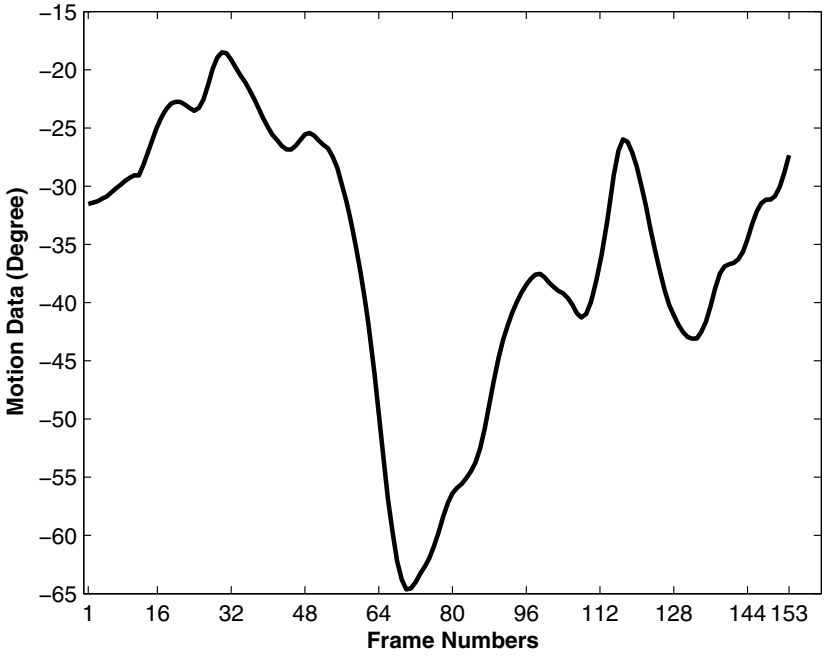

Fig. 2. Motion curve of *right shoulder* joint (rotation along z-axis) in 153 frames

4 Cardinal Spline

This section describes the construction of parametric cubic Cardinal spline. Cardinal spline is a C^1 continuous curve. Cardinal spline interpolates piecewise cubics with specified endpoint tangents for each segment. A Cardinal spline segment, as shown in Figure 5, is defined by four control points, i.e., P_{j-1}, P_j, P_{j+1} and P_{j+2}. The j^{th} segment of Cardinal spline interpolates between two *middle control points*, i.e., P_j and P_{j+1}. The *end control points*, i.e., P_{j-1} and P_{j+2} are used to calculate the tangents at P_j and P_{j+1}. Equations for boundary conditions of j^{th} segment are written as:

$$P'_j = \frac{1}{2}(1-T)\left(P_{j+1} - P_{j-1}\right), \tag{1}$$

$$P'_{j+1} = \frac{1}{2}(1-T)\left(P_{j+2} - P_j\right), \tag{2}$$

where parameter T is *Tension* and it controls looseness/tightness of spline.

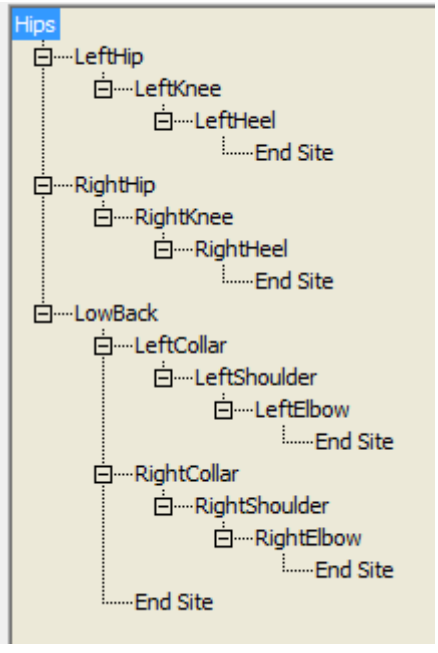

Fig. 3. Hierarchical skeleton

```
HIERARCHY
ROOT Hips
{
    OFFSET 0 0 0
    CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
    JOINT LeftHip
    {
        OFFSET 3.5 0 0
        CHANNELS 3 Zrotation Xrotation Yrotation
        JOINT LeftKnee
        {
            OFFSET 0 -19.0555 0
            CHANNELS 3 Zrotation Xrotation Yrotation
            JOINT LeftHeel
            {
                OFFSET 0 -21.1464 0
                CHANNELS 3 Zrotation Xrotation Yrotation
                End Site
                {
                    OFFSET 0 0 9.64661
                }
            }
        }
    }
    ...
```

Fig. 4. Hierarchical skeleton in BVH file format

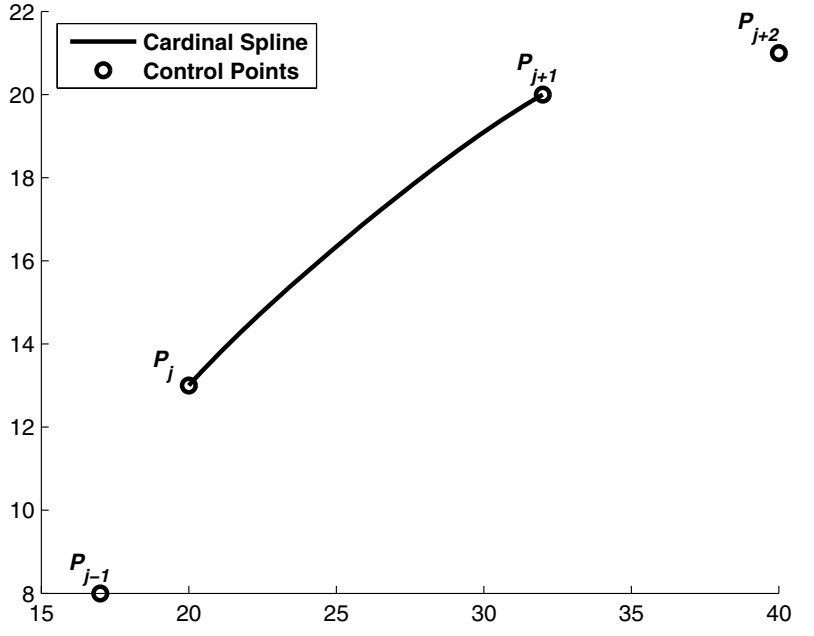

Fig. 5. The j^{th} Cardinal spline segment, $T = 0$

For l joined segments, there are $2l$ conditions for continuity of functions and $2l$ conditions for continuity of slopes. Finally the equation of Cardinal spline for j^{th} segment is written as follows:

$$\begin{aligned} Q(t_i) =& (-st_i^3 + 2st_i^2 - st_i)P_{j-1} \\ &+ [(2-s)t_i^3 + (s-3)t_i^2 + 1]P_j \\ &+ [(s-2)t_i^3 + (3-2s)t_i^2 + st_i]P_{j+1} \\ &+ (st_i^3 - st_i^2)P_{j+2}, \end{aligned} \tag{3}$$

where t_i is parameter of interpolation , $0 \leq t_i \leq 1$, and s is related to *Tension* by $s = \frac{(1-T)}{2}$. In order to generate n points between P_j and P_{j+1} inclusive, the parameter t_i is divided into $(n-1)$ intervals between 0 and 1 inclusive, and $Q(t_i)$ is evaluated (interpolated) at n values of t_i.

5 Tweening Strategy

This section elaborates the strategy of motion tweening. Our objective is to generate smooth animation using few keyframes. We will describe the strategy of motion tweening of a single channel of a joint. Same strategy is applied for all the channels of all the joints. Initially the user has to specify the keyframes. These keyframes are taken as control points of Cardinal spline. There are three possibilities to specify keyframes which are as follows:

1. Keyframes can be specified from the existing motion data at regular interval, no human intervention is required.
2. User manually input/select the keyframes at non-regular interval, human intervention is required.
3. Feature point extraction algorithm is use to specify keyframes at non-regular interval, no human intervention is required.

Several feature point extraction algorithms based on curvature analysis are proposed in the literature, such as in [2], [6], [7], etc. We modified the algorithm of [6] to extract the keyframes from motion data. Curvature based algorithms are sensitive to some threshold value, specified as an input. We took default values of all the input parameters specified in [6]. Once the keyframes are specified/extracted then Cardinal spline interpolation is used to generate in between frames between keyframes. A single Cardinal spline segment is constructed between each adjacent pair of keyframes by taking adjacent keyframes as middle control points, i.e., P_j and P_{j+1} of Cardinal spline. Therefore, for n keyframes the number of Cardinal spline segments are $n-1$.

Since, in addition to control points of the current segment, Cardinal spline needs control points of previous and next segments (see Eq. 3). Therefore, we used keyframes of previous, current and next segments as control points of a current Cardinal spline segment and obtained the interpolated data of current segment using Eq. (3). An interesting question is how to obtain the keyframes of first and last segments? Because they do not have previous and next segments respectively. We opted to take $P_{-1} = P_0$, for the first segment and $P_{l+2} = P_{l+1}$, for the last segment.

All the keyframes and their in-between frames constitute a complete motion curve of the channel. Figure 6 shows the keyframes and motion curve of *right-shoulder* (rotation along z-axis) obtained by Cardinal spline interpolation between keyframes. Note that this motion curve is similar but smoother than Figure 2, where no motion tweening was used. It is also worth to mention that in the proposed method, only keyframes are needed to be saved, while in-between frames are generated on the fly using keyframes. Where as in the conventional systems, all the frames need to be saved. Algorithm 1 summarizes the motion tween strategy.

Algorithm 1. Motion Tweening

1: Extract or specify the keyframes. First and last frames are always taken as keyframe.
2: Specify number of in between frames between every pair of keyframes.
3: Take each pair of adjacent keyframes as middle control points, i.e., P_j and P_{j+1} of Cardinal spline. Take previous and next keyframes as first and last control points (P_{j-1} and P_{j+2}) respectively.
4: Obtain the motion tweening data by Cardinal spline interpolation between every pair of keyframes P_j and P_{j+1}.
5: If the result is not satisfactory then go to step 1, add more keyframes and repeat the process.
6: Save the keyframes.

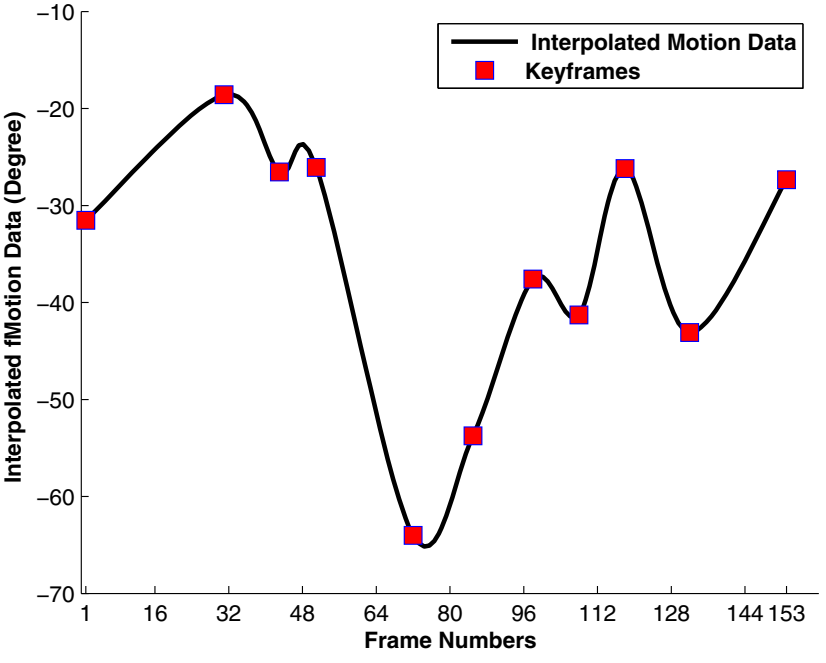

Fig. 6. Interpolated data (Motion tweening) with keyframes for *right shoulder* joint

6 Simulation and Results

Simulation is done on motion capture data of various animation sequences. Table 5 provides details about environment of simulation. Even though the proposed method needs only keyframes. But in order to see how effectively the proposed method performs tweening that resembles original motion, we took complete sequences and extracted the keyframes from these sequences then performed the motion tweening on the keyframes. Table 2 gives the details of two animations namely *Ladder-Climb* and *Walk-Turn* used

Table 1. Environment of simulation

Hardware	Inter Core Duo 2.4 GHz
Operating System	Linux (Ubuntu 10.04 LTS)
Programming Language	MATLAB R2010

Table 2. Animations used in the simulation

Animation Name	Frame Rate (fps)	Number of Frames
Ladder-Climb	30	153
Walk-Turn	24	119

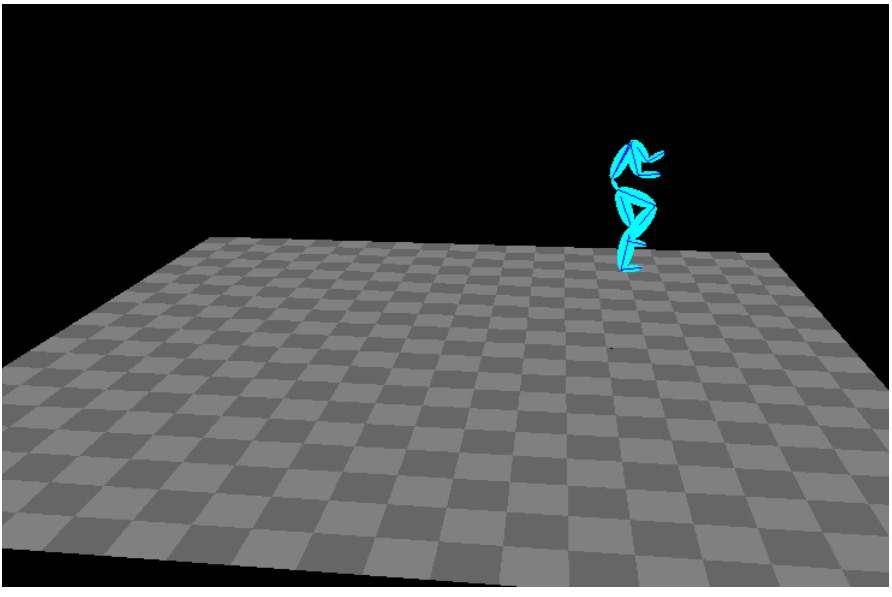

Fig. 7. Frame 133 obtained by motion tweening of keyframes of *Ladder-Climb* animation

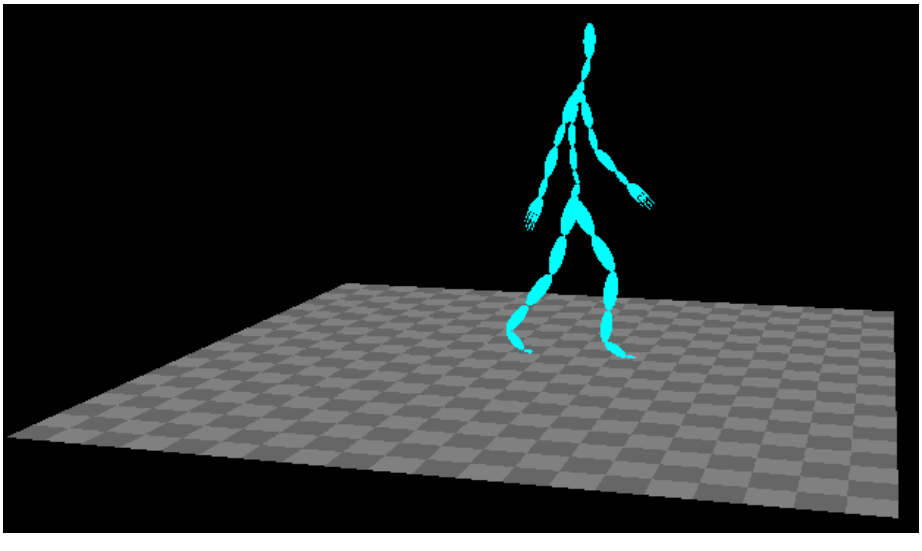

Fig. 8. Frame 7 obtained by motion tweening of keyframes of *Walk-Turn* animation

in the simulations. In the *Ladder-Climb* animation, a character climbs on a ladder. In the *Walk-Turn* animation, a character first walks straight then turn back 180 degree and walks again. Figures 7 and 8 show the frames obtained by motion tweening of *Ladder-Climb* and *Walk-Turn* keyframes respectively.

7 Conclusion

Motion data is widely used in many applications such as video games, computer animation, and animated videos. Motion tweening is an essential process to generate computer based animations using keyframes. We presented a simple and efficient method of tweening of motion data to generate smooth animation using cubic Cardinal spline. For each channel of a joint, keyframes are specified manually or extracted automatically as an input to the method. The keyframes serve as control points of Cardinal spline to generate in between frames by Cardinal spline interpolation. We presented the algorithm of motion tweening using Cardinal spline. The proposed method is well suited to generate skeletal animations. Simulation results show that the proposed method yields very good results for variety of animations.

References

1. http://www.videocapturehardware.org/video-capture-artifacts/
2. Awrangjeb, M., Lu, G.: Robust image corner detection based on the chord-to-point distance accumulation technique. IEEE Transactions on Multimedia 10(6), 1059–1072 (2008)
3. Bartels, R.H., Beatty, J.C., Barsky, B.A.: An Introduction to Splines for Use in Computer Graphics and Geometric Modeling. Morgan Kaufmann (1995)
4. Bejancu, A.: A new approach to semi-cardinal spline interpolation. East Journal on Approximations 6(4), 447–463 (2000)
5. http://www.cs.wisc.edu
6. He, X.C., Yung, N.H.C.: Curvature scale space corner detector with adaptive threshold and dynamic region of support. In: Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004), vol. 2, pp. 791–794. IEEE Computer Society, Washington, DC, USA (2004)
7. He, X.C., Yung, N.H.C.: Corner detector based on global and local curvature properties. Optical Engineering 47 (2008)
8. Johan, H., Koiso, Y., Nishita, T.: Morphing using curves and shape interpolation techniques. In: Proc. Of The Pacific Graphics 2000 (PG 2000), 1 Rao, And Shah / View Interpolation, pp. 348–358 (2000)
9. Kochanek, D.H.U.: Interpolating splines with local tension, continuity, and bias control. Computer Graphics 18(3), 33–41 (1984)
10. Silbermann, M.J., Tagare, H.D.: Local cardinal spline interpolation and its application to image processing. In: Proceedings of SPIE, pp. 272–283 (February 1992)
11. Menache, A.: Understanding Motion Capture for Computer Animation. Morgan Kaufmann (2010)
12. Mukai, T., Kuriyama, S.: Geostatistical motion interpolation. In: ACM SIGGRAPH 2005 Papers, pp. 1062–1070. ACM, New York (2005)
13. Renka, R.J.: Algorithm 716: TSPACK: Tension spline curve-fitting package. ACM Transactions on Graphics (TOG) 19(1), 81–94 (1993)
14. Rose, C., Bodenheimer, B., Cohen, M.F.: Verbs and adverbs: Multidimensional motion interpolation using radial basis functions. IEEE Computer Graphics and Applications 18, 32–40 (1998)
15. Rosenhahn, B., Klette, R., Metaxas, D.: Human Motion: Understanding, Modelling, Capture, and Animation. Springer, Heidelberg (2007)
16. Sarfraz, M.: Interactive curve modeling with applications to computer graphics, vision and image processing. Springer, Heidelberg (2008)
17. Sun, N., Ayabe, T., Okumura, K.: An animation engine with the cubic spline interpolation. In: International Conference on Intelligent Information Hiding and Multimedia Signal Processing, pp. 109–112 (2008)