COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds 2011; **22**:3–14
Published online 8 November 2010 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cav.380

RESEARCH ARTICLE

Multiscale motion saliency for keyframe extraction from motion capture sequences

Cihan Halit and Tolga Capin*

Department of Computer Engineering, Bilkent University, Ankara, Turkey

ABSTRACT

Motion capture is an increasingly popular animation technique; however data acquired by motion capture can become substantial. This makes it difficult to use motion capture data in a number of applications, such as motion editing, motion understanding, automatic motion summarization, motion thumbnail generation, or motion database search and retrieval. To overcome this limitation, we propose an automatic approach to extract keyframes from a motion capture sequence. We treat the input sequence as motion curves, and obtain the most salient parts of these curves using a new proposed metric, called 'motion saliency'. We select the curves to be analysed by a dimension reduction technique, Principal Component Analysis (PCA). We then apply frame reduction techniques to extract the most important frames as keyframes of the motion. With this approach, around 8% of the frames are selected to be keyframes for motion capture sequences. Copyright © 2010 John Wiley & Sons, Ltd.

KEYWORDS

motion saliency, motion capture, keyframe extraction, PCA

***Correspondence**

Tolga Capin, Department of Computer Engineering, Bilkent University, Ankara, Turkey.
E-mail: tcapin@cs.bilkent.edu.tr

1. INTRODUCTION

Motion capture is a central technique in the creation of computer games, virtual environments, digital special effects in movies and medical applications. The power of motion capture comes from its ability to produce very realistic results, even for the most complex motions, in real time. Unfortunately, problems still arise when motion capture is used. It is costly to capture an action, as repeated trials are commonly needed. Another problem of motion capture is that applying the captured sequence to a different character model requires a complex retargeting process.

Keyframing delivers a potential solution that overcomes the disadvantages of the use of motion capture alone: massive amounts of motion capture data can be summarized by keyframes, and keyframe editing can be used on an already captured sequence to obtain a new motion without having to go through the costly process of recapturing. This process requires that representative frames be selected from a very large set of frames of a motion capture sequence, which is the focus of this work.

In this paper, we propose a new multiscale approach to extract keyframes from a motion capture sequence. We treat the input motion data as a set of motion curves, and find the most salient parts of these curves that are crucial in the representation of the motion behaviour. We apply the idea of *motion saliency*, a new multiscale metric, to motion curves in the first step of our algorithm. The multiscale property of our approach allows us to measure the degree of difference between the 'centre' frame and the 'surround' frames, in different time scales. Then in the second step, we apply clustering and keyframe reduction techniques to obtain the most important keyframes of the motion.

There are a large number of potential applications of our solution. An obvious application of our technique is as a tool in summarization: it allows the user to acquire an image preview of the motion. This approach is commonly used in automatic thumbnail generation for motion capture databases. Our technique is also applicable for motion editing; it allows the user to manually edit the motion by automatically selecting relevant keyframes. Our method can also be used as a tool in a wide variety of applications, such as motion understanding, motion compression, motion database search and retrieval.

As discussed in the previous work section, there exist prior brute-force approaches to extract better set of keyframes; however, they are computationally expensive with computational complexity $O(n^2)$, where n is the number of frames in the motion capture sequence. Our algorithm solves the same problem with $O(n)$ complexity, and provides a solution that can be used in interactive applications and for processing large motion capture databases.

The remainder of the paper is organized as follows. First, we survey previous work on keyframe extraction and the saliency model of attention. Then, we explain our multiscale keyframe extraction approach with a new metric to estimate the importance of a frame. Lastly, we provide the experimental results and applications of our approach, and present discussions and conclusions of our work.

2. RELATED WORK

We draw upon different areas of research for our keyframe extraction method. The following sections discuss the relevant work in these areas.

2.1. Keyframe Extraction

Various methods have been proposed for keyframe extraction; these methods can be classified into three categories in terms of their approach: (i) curve simplification, (ii) clustering and (iii) matrix factorization. Figure 1 illustrates the difference among these approaches, and the related work for each category is summarized and discussed below.

Curve Simplification Based Algorithms are principally based on simplifying a motion curve as a set of straight lines, which describe the original curve with a certain error margin. An initial work of curve simplification belongs to Lim and Thalmann [1]. Their method uses Lowe's algorithm [2] for curve simplification. Starting with the line which combines the start and end points of the curve, the algorithm divides the line into two line segments if the maximum distance of any point on the curve from the line is larger than a certain error rate. The algorithm performs the same process recursively for each new line segment, until the desired error rate is achieved. Another approach that aims to find the keyframes based on motion curves is the work of Okuda *et al*. [3,4] This algorithm detects the keyframes in motion capture data by using frame decimation: the frames are decimated one by one, according to their importance. When a desired number of keyframes are obtained, the process stops. Another related work is Matsuda and Kondo's approach [5]. First, the solution finds the fixed frames of the motion, which satisfy one of the following: (i) local minimum or maximum value, (ii) one of the end points of a straight line, (iii) a point that has a 'large angle difference' on both sides; that is, the points which are at least 50% of the amplitude far away from the neighbour frames. Having the fixed frames of the motion that cannot be deleted, Matsude *et al*. apply reduction operations to the other characteristic frames and find the keyframes of the motion. However, this method is not optimal; it has been reported that on average, 55% of all frames are selected to be keyframes of a motion.

Clustering Based Methods transform the keyframe extraction process into a clustering algorithm. In this approach, similar frames are grouped into clusters, and a representative frame is selected from each cluster as a keyframe. Similarity is defined here as a function of weighted distance between joints of the character between frames [6]. Park *et al*. [7] represent motion capture data in quaternion form, and apply Principal Component Analysis (PCA) and k-means clustering on quaternions. Then, scattered data interpolation is used to extract keyframes out of these clusters. The order of frames is an important part of keyframe extraction, but these approaches generally do not take the order of frames into account in the clustering step.

Matrix Factorization Based Methods represent frames of motion sequence as matrices, such as feature frame matrices formed by colour histograms of frames. By using techniques such as singular value decomposition [8] or low-order discrete cosine transform (DCT) [9], the summary of the motion is constructed. Key-Probe is such a factorization technique, which constructs a frame matrix that holds vertex positions of a 3D mesh, and processes this matrix to extract the key matrix [10]. Every other frame in the sequence is a linear combination of this key matrix. The key matrix can be calculated by specifying the number of keyframes or an error threshold.

Each of these approaches has its advantages and disadvantages. Curve simplification algorithms are very efficient in terms of speed, but these algorithms have to consider only a subset of the curves which represent the motion in order work efficiently; and there is no single way to select the

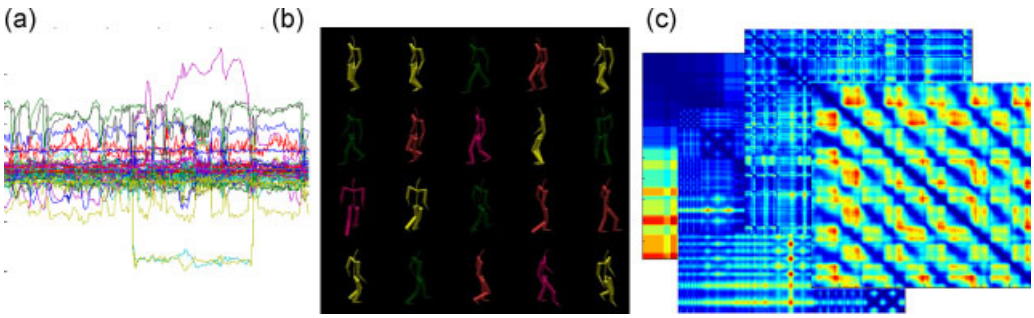

Figure 1. Summary of approaches for keyframe extraction. (a) Curve simplification based methods extract and analyse motion curves of each DOF; (b) clustering based methods compare the motion frame by frame to group similar frames into clusters; (c) matrix factorization based methods form a matrix, where frame-byframe comparison is computed, and keyframes are then extracted by using matrix algebra.

DOI: 10.1002/cav

best subset among all subsets. On the other hand, clustering and matrix factorization methods do not have to select such subsets, and they can operate on the entire set of data. The disadvantage of clustering algorithms is that they do not take the order of the frames into account. However, discarding the time domain in keyframe extraction is a big disadvantage. The best performing methods, in terms of extracting the most favourable keyframes, are the matrix factorization methods; however, these algorithms perform very slowly due to their quadratic running time complexity.

2.2. Human Motion

Several researchers have observed that there is a lot of redundancy in human motion [1], which is caused by the fact that human joints act in a coordinated manner for any kind of motion [11]. Various animation methods, covering a wide range of techniques from inverse kinematics to procedural animation, make use of this property. Coleman *et al*. [12] use coordinated features of human motion, such as maxima of acceleration and directional acceleration, to extract staggered poses out of a capture sequence. Procedural generation of grasping motion is another popular technique; Parent states the motions of all fingers are inter-related in the grasping motion [13]. Other studies have also shown that the spine controls the human motion, and therefore torso, arms and head, are involved even in a simple walking motion [13]. All these and similar findings arise from the fact that the human body aims to minimize the total amount of strain on the body, thus many joints strain in a little amount to account for a great strain on a single joint. Furthermore, many human motions, such as walking, are cyclic in nature and create uniformity in joint positions [14]. Methods which analyse human motion exploit this feature by defining joint groups, where joints in a joint group are functionally dependent on each other [15].

Our method removes such redundancies in human motion using the PCA method. PCA has been used successfully for a number of applications in computer animation. For example, Alexa and Müller [16] have applied this technique to mesh deformations, yielding up to 100 times compression of meshes. Glardon *et al*. [17] have applied the PCA technique to a set of motion captured walking animations, for reconstruction of parameters that provide high-level control over subsequent procedural walking animation. Sattler *et al*. [18] use PCA to compress mesh animations. This method uses Clustered PCA to employ clustering on the mesh, and then applies PCA to compress each cluster by itself. Barbic *et al*. [19] use different models of representation to classify motion capture segments, and they conclude that the best representation is provided by Probabilistic PCA.

2.3. Saliency

Our method builds upon the saliency model that is popular in the perception field. *Saliency*, which characterizes the level of significance of the subject being observed, has been a focus of cognitive sciences for more than 20 years. Saliency is commonly thought as a visual cue, but in effect it is a multiscale metric to measure the significance of the subject as the result of its contrast from its neighbours. Itti *et al*. [20] describe one of the earliest methods to compute the saliency map of 2D raster images. Lee *et al*. [21] have introduced the saliency model of 3D static meshes, using the curvature property of mesh vertices. They have shown how the computed saliency values can be used to drive the simplification of the 3D mesh or best viewpoint selection.

We propose a new *motion saliency* metric for motion frames, based on the multiscale centre–surround operator on Gaussian-weighted mean curvatures in the motion curve. In earlier work, Bulut and Capin [22] use a centre–surround operator to detect important frames from the motion curve. This approach has the shortcoming of selecting a single joint among all the joints to analyse for keyframing, which yields a large error if the wrong joint is selected. Also compared to this earlier work, our proposed method analyses each degree of freedom (DOF) according to its importance by dimensionality reduction, and selects keyframes from a number of motion curves to obtain an optimal amount of keyframes. Finally, our proposed approach uses a multiscale operator with different neighbourhood sizes. The multiscale property of our approach allows us to measure the degree of difference between the 'centre' frame and the 'surround' frames, in different time scales. This type of approach leads to a more robust detection of keyframes. We explain the details of the proposed algorithm in the next section.

3. APPROACH

Our method is composed of four steps: dimension reduction, computing motion saliency, candidate keyframe selection and clustering (Figure 2). The first step involves dimension reduction of the input motion sequence. Our approach considers each joint angle of the character as a separate dimension of the signal, and applies PCA to reduce input dimension space. Next, an initial set of candidate keyframes is selected in the reduced dimensions, using the proposed motion saliency metric. Our method of computing the saliency of each frame is based on the centre–surround operator of Itti *et al*. [20]. In the final step, clusters are formed for neighbouring candidate keyframes and the most significant keyframe is selected within each cluster.

There are clear advantages for decoupling these three tasks as opposed to solving them jointly. First, the human motion is essentially a high-dimensional signal, and dimension reduction helps to remove redundancies in the motion input. Second, rather than focusing on local features of frames that typically have large variations; our solution aims to capture the saliency of frames by searching over varying ranges of frame neighbourhood. Third, the candidate keyframes tend to form groups, because of the high frame rate for motion capture and the smooth nature of human motion; and the clustering step helps to exploit the similarity

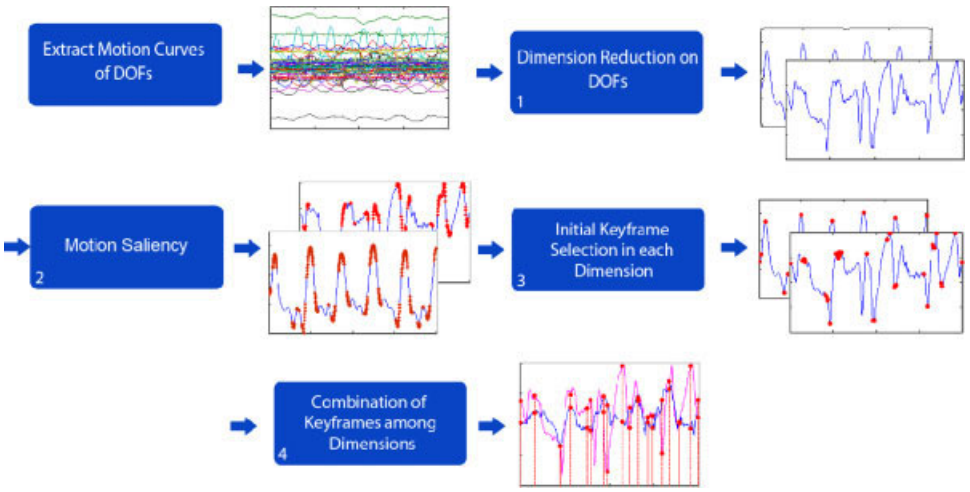

Figure 2. General outline of the method.

in neighbouring frames. From a computational point of view, by solving each of the problems separately in linear time, our entire algorithm runs in linear time.

3.1. Step 1: Dimension Reduction

Human motion is a high-dimensional signal, and it would be infeasible to consider every dimension of the motion in keyframe extraction. However, there are strong correlations between different joint groups, as shown by previous research discussed above. To propose a general solution, we assume that the joints that form these groups are not fixed, and they differ from one motion to another.

We use the PCA dimension reduction technique to take account of this correlation [16]. The PCA technique analyses differences and similarities of data consisting of many trials and a high number of variables, in order to find suitable bases for dimension reduction. We use the PCA method to find the principal components of the input motion. In other words, we reduce the number of degrees of freedom in the motion capture sequence while not losing much of the 'information' content of the motion. There are various methods to perform PCA, such as SVD (Singular Value Decomposition) and covariance matrices [16]. In this work, we use covariance matrices, because extracting eigenvalues from the covariance matrix helps us to find the importance of every component. We use these eigenvalues in the candidate keyframe selection stage.

We build the PCA model so that each joint angle (represented as three Euler angles) is considered as a dimension in the high dimensional space. We construct an $n \times m$ data matrix:

$$M = \begin{bmatrix} f_1 = \{p_{11} & \dots & p_{1m}\} \\ & \vdots & \\ f_n = \{p_{n1} & \dots & p_{nm}\} \end{bmatrix} = [J_1 \dots J_m]$$

where m is the original number of degrees of freedom, and n is the number of frames in the motion, which is standardized in each column. We then construct an $m \times m$ covariance matrix from this data, and find eigenvectors and eigenvalues of this matrix. We choose the most significant k eigenvectors as the new bases by selecting eigenvectors with the highest k eigenvalues. As a result, we form a row feature vector R by

$$R = (\text{eig}_1 \dots \text{eig}_k)$$

where eig_i is the ith significant eigenvector. Thus, by choosing k eigenvectors, we reduce the number of dimensions of the human motion data from m to k. We then construct the reduced motion matrix F, as

$$F^{\text{T}} = R \times D^{\text{T}} = [D_1 \dots D_k]^{\text{T}}$$

where F is a $n \times k$ matrix, and k dimensions are represented as separate k curves, D_i. The percentage variance maintained during the dimension reduction can be found by the ratio of the sum of the selected eigenvalues to the sum of all eigenvalues.

We observe that there is no single number of dimensions k suitable for every motion sequence. This greatly depends on the nature of the motion. Naturally, a higher number of dimensions have to be used for highly dynamic motions, such as jumping, running, etc. in order to maintain visual quality in the constructed motion. A detailed analysis of this parameter is presented in the results section.

3.2. Step 2: Computing Motion Saliency

The second step selects the candidate keyframes in the reduced dimensions. To identify the salient frames in the motion curves, we apply multiscale Gaussian filters. Our method of computing the saliency of each frame is based on the centre–surround operator of Itti *et al.* [20], which

Comp. Anim. Virtual Worlds 2011; **22**:3–14 © 2010 John Wiley & Sons, Ltd.
DOI: 10.1002/cav

measures the degree of difference between the 'centre' and 'surround' of an image element. Lee *et al*. [21] have used this metric for 3D mesh geometry to calculate the regions of the mesh that attract attention. Our method for computing the significance of motion frames uses a similar curvature-based centre–surround operator, adapted to motion curves.

The first step in our saliency computation is calculating standard curvatures of the motion curves in the neighbourhood of each frame i [23]:

$$C(D_{ij}) = \frac{|D''_{ij}|}{\left(1 + D_{ij}^2\right)^{3/2}}$$

where D_{ij} is the value of principal curve j at frame i in the lower dimension space, calculated by PCA.

The second step computes the Gaussian-weighted average of the curvature in the neighbourhood of a point for each curve, assuming a Gaussian distribution with mean 0 and standard deviation σ and centred at that point. We calculate values for fine-to-coarse scales, with standard deviation $(\sigma_1, \sigma_2, \sigma_3, \sigma_4)$ [21]:

$$G(f, \sigma) = \frac{\sum_{x \in N(f,2\sigma)} C(f) e^{[-||x-f||^2/(2\sigma^2)]}}{\sum_{x \in N(f,2\sigma)} e^{[-||x-f||^2/(2\sigma^2)]}}$$

where $G(f, \sigma)$ is the Gaussian-weighted average of curvature at frame f. To compute the motion saliency of a frame f, we use the saliency definition of frame f at a scale level i as $S_i(f)$ following the approach proposed by Lee *et al*. [21]. Extracting poses out of important features of animation is better done in multiple scales, because the features of animation can be categorized on multiple scales, such as the individual poses of a walking animation on a small scale and the transition from a long walking motion to a jumping motion on a bigger scale. To be able to capture all features on different scales we apply our multiscale model:

$$S_i(f) = |G(f, \sigma_i) - G(f, 2\sigma_i)|$$

where σ_i is the standard deviation of the Gaussian filter at scale i. Although any ratio can be choosen for σ_i, our motivation in selecting small ratios between scales is to avoid disregarding the highly temporal coherency in human animation and to eliminate problems arising when choosing a large neighbourhood, such as foot skating. A detailed explanation for σ parameter is presented in the results section.

$$(\sigma_1 = \sigma,\ \sigma_2 = \frac{3}{2}\sigma;\ \sigma_3 = 2\sigma;\ \sigma_4 = \frac{5}{2}\sigma)$$

The final *motion saliency S(f)* is computed by adding the saliency maps at all scales after applying nonlinear normalization operator [20]. Since we have a number of saliency maps, each saliency map has to be normalized within the same values to ensure equal contribution of each saliency map. Therefore, we first normalize each saliency map between [0, H], where H is the greatest saliency value among all the saliency maps. Then we find the global maximum M_i of the saliency map and calculate mean m_i of the local maxima excluding M_i and then multiply the saliency map by $(M_i - m_i)^2$. This multiplication gives the nonlinear property to the normalization; and it eliminates excessive number of salient points which would arise in a linear normalization.

If a point is significant for the motion, it is due to its location on the curve. That is, if its value shows a remarkable change according to the values of neighbouring points, it is a significant frame. If a remarkable change occurs in the value of the point between the results of the two Gaussians, then its motion saliency has a higher value. Figure 3 shows the saliency values on a sample motion curve.

3.3. Step 3: Candidate Keyframes

After the calculation of the saliency value for each frame on the motion curve, we define the frames having a saliency value greater than average saliency for the motion as *candidate keyframes*. Figure 4 shows the result of this process on the curve given in Figure 3.

The value σ is a significant parameter in keyframe selection process and it is not uniform for every motion sequence. Our approach is to reduce σ as the motion becomes more

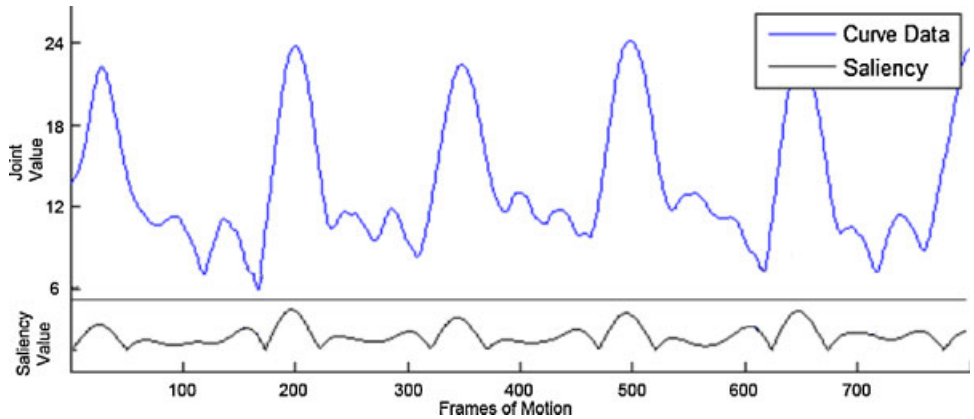

Figure 3. Saliency values of frames in a sample joint in walking action.

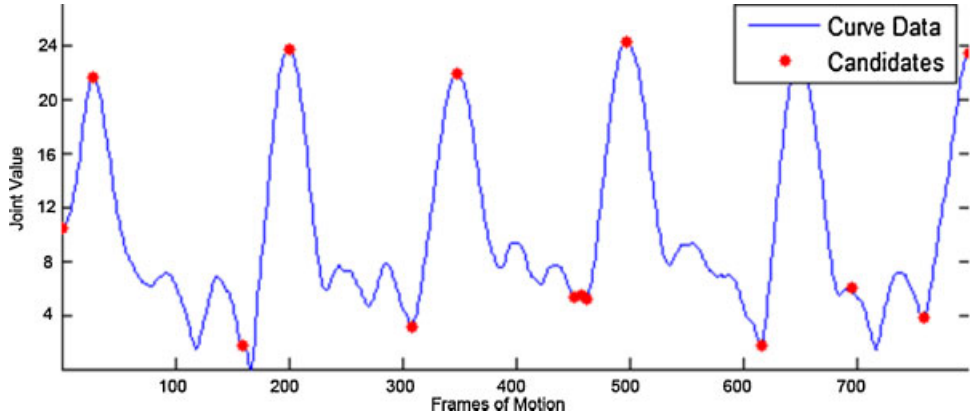

Figure 4. Set of candidate keyframes are indicated on sample curve. Each of the points indicated in red are more salient than the average saliency value.

dynamic, in order to maintain fast and frequent changes in these types of motions. This parameter is examined in detail in the results section.

3.4. Step 4: Clustering

In the third step, we form clusters for neighbouring candidate keyframes, and select the most significant keyframe within each cluster. The selected keyframes from different curves tend to form clusters, due to the redundancy of human motion as discussed above. Extracting each frame in every group as a keyframe results in an excessive number of candidate keyframes. Since each group actually represents similar keyframes in the motion, it is only sufficient to select the most significant frame among each group to represent that group. For every cluster, i.e. for each of group of sequential frames, the selection of the most significant frame is done by calculating a weighted average among each cluster. The weight of each frame is assigned as the eigenvalue of the curve where this frame was found salient.

However, the human motion is continuous and there may be times where clusters overlap each other while the actor is finishing a pose and starting another. Therefore we limit the maximum number of elements in a group to 2σ, because every frame takes into account a neighbourhood of 2σ while calculating the saliency of that frame.

4. COMPLEXITY

The computational complexity of the system can be assessed by evaluating the complexity of each step in Figure 5. The combination of these complexities will yield the overall complexity of the algorithm.

1. Construct frame matrix $M = \begin{bmatrix} f_1 = \{p_{11} & \cdots & p_{1m}\} \\ \vdots & & \\ f_n = \{p_{n1} & \cdots & p_{nm}\} \end{bmatrix}$ from sequence, where $p_{i,j}$ is the rotation of the j^{th} joint in i^{th} frame of motion.

2. Apply dimension reduction on columns of frame matrix M, i.e. PCA(M). At least 7 dimensions and 90% of variance are acquired.

3. Find salient frames of each of the motion curves in reduced dimension space.

3.1 Calculate multiscale saliency maps for each curve, i.e. $S(f) = \oplus_i S_i(f)$.

3.2 Extract salient frames from the saliency map $S(f)$.

4. Collect every extracted frame into one set and assign sequential frames into same clusters. Extract a single keyframe from each cluster.

4.1 Calculate weighted average of each cluster, where weights of the points are assigned from the eigenvalues of the dimension which this point was extracted as salient.

4.2 Point closest to the weighted average in each cluster is in the set of final keyframes.

Figure 5. Algorithm.

Comp. Anim. Virtual Worlds 2011; **22**:3–14 © 2010 John Wiley & Sons, Ltd.
DOI: 10.1002/cav

The first step of the algorithm places the values in matrix M, which can be trivially done in linear time during processing of the motion capture sequence. In the second step, the PCA method is used to perform dimension reduction. This part of the algorithm governs the complexity of the algorithm, because of the matrix operations involved in the algorithm. The calculation of the covariance matrix involves the multiplication of one $m \times n$ and one $n \times m$ matrix which can be done in $O(m^2n)$ time. The calculation of eigenvectors and eigenvalues are performed in constant time, independent of the number of frames, because inputs and outputs of this function are always $m \times m$ matrices. Lastly, the matrix multiplication between a $n \times m$ and $m \times k$ matrix takes $O(kmn)$ time. Since the system operates on motion capture data, the number of sensors is fixed for every motion and does not increase asymptotically. Therefore, the number of captured degrees of freedom, value m, can be considered as constant. The system sets a bound on the number of principal components, which also renders k as constant. Therefore, the PCA operation effectively becomes an $O(n)$ process for this system.

The next step of the algorithm creates the saliency maps by filtering the curvature values of the principal components, all of which can be done in linear time by sequentially computing these values. Lastly, clustering sequential candidate frames together is performed in linear time.

Therefore, on the overall, each step contributes to the algorithm with $O(n)$ complexity, therefore the entire system runs in $O(n)$.

5. RESULTS

We have tested our method using motions from the motion capture database of Carnegie Mellon University [24]. From this database, we have selected motions with different dynamic properties to measure the performance of our algorithm under different conditions. We have selected walking and stretching as *low-dynamic motions*, and playing basketball and boxing as *high-dynamic motions*. All the motion capture sequences were recorded with 120 frames/sec. In order to assess the quality of our work, we have evaluated the reconstruction of the original motion back from the frames selected as keyframes. We have used a metric, *mean squared error*, to quantitatively describe the quality of the reconstructed motion.

The first observation is that more keyframes must be selected in a motion as it becomes more dynamic, otherwise the visual quality of the reconstructed motion degrades significantly. Another important observation is that there is a direct relationship between the number of selected keyframes and the error rate of the constructed motion. Different values of σ and k yield different sets of keyframes. These parameters are studied in detail in the next section.

We use Euler joint angles as the main representation method in the system. We have used quaternion spherical interpolation for reconstruction of the motions from the keyframes We also provide results with motion curves in the Euclidean body coordinate space that is local to the body, with body root as the origin, for comparison.

5.1. Keyframe Extraction

We have compared our results with two state-of-the-art keyframe extraction techniques: Frame Decimation [4] and Curve Simplification [25], both for low- and high-dynamic types of motions. We have slightly modified these two algorithms to be able to compare them with our approach: all methods are assumed to take the same desired number of keyframes as input. In the case of frame decimation, no changes had to be made because the initial algorithm already depends on the required number of keyframes as the stopping condition. Curve Simplification, however, depends on another metric, called 'tolerance', which defines the maximum distance between the original curve and constructed curve. There is no direct conversion between tolerance and the number of selected keyframes; therefore we have modified this approach to continue until the desired number of keyframes are acquired. The output of each algorithm is an interpolated motion sequence, with a rate of 120 frames/second.

Naturally, due to interpolation for in-between frames, the method might not be able to reconstruct a motion exactly as its original on a per frame basis, even if optimum keyframes were selected. Therefore, we compute the mean squared error rate of the techniques using the formula:

$$E = \left[\frac{\sum_i (\sum_k |F_i^o(k) - F_i^r(k)|^2)}{n \times j} \right]$$

where $F_i^o(k)$ and $F_i^r(k)$ are the (x, y, z) body coordinate values of kth joint of the ith frame in the original and reconstructed motions respectively, n is the number of frames, and j is the number of dimensions in the skeleton (62 for our motions). Table 1 shows the comparison of this error for Frame Decimation, Curve Simplification, and our approach. The table illustrates that our algorithm creates an error rate higher than of Frame Decimation [4], as expected, as Frame Decimation applies a brute-force approach that creates near-optimal keyframes. However, our approach is asymptotically more efficient than Frame Decimation with increasing number of frames, since the complexity of our algorithm is linear compared to $O(n^2)$ complexity of Frame Decimation. Our algorithm performs better in terms of error rate compared to the Curve Simplification Algorithm [25], which has $\Omega(n \log n)$ complexity. Table 1 also shows that body coordinate representation yields better performance than Euler angle representation because less keyframes are selected with less error.

To be able to compare the error rates among different motions, we have also calculated the PSNR (peak signal-to-noise ratio) using the formula below for Euler angle representation:

$$\text{PSNR} = 20 \times \log_{10}\left(\frac{\text{MAX}}{\sqrt{E}}\right)$$

Table 1. Comparison of keyframe extraction algorithms in the Euler angle space and body coordinate space.

Low dynamic motions			High dynamic motions		
	Stretching	Walking		Basketball	Boxing
No. of frames	4592	800	No. of frames	4905	3000
		Euler representation			
($\sigma = 10$, $k = 7$) No. of keyframes	131	38	($\sigma = 3$, $k = 10$) No. of keyframes	616	311
Saliency error	0.06	0.0005	Saliency error	0.024	0.0088
Curve simplification error	0.15	0.0069	Curve simplification error	0.045	0.0105
Frame decimation error	0.02	0.0001	Frame decimation error	0.020	0.0027
		Body coordinate space			
($\sigma = 10$, $k = 7$) No. of keyframes	117	23	($\sigma = 3$, $k = 10$) No. of keyframes	559	305
Saliency error	0.0527	0.0004	Saliency error	0.0033	0.0057
Curve simplification error	0.0847	0.0140	Curve simplification error	0.0067	0.0087
Frame decimation error	0.0269	0.00008	Frame decimation error	0.0017	0.0024

where MAX is the greatest value which a joint angle can take (i.e. 360 degrees). In the body coordinate system, each joint has different bounds for its allowed positions, thus a PSNR value cannot be determined for the constructed motions with body coordinate representation.Table 2 shows the PSNR values of the three approaches. As illustrated in Table 2, for low-dynamic motions, all the three keyframe extraction methods provide higher signal-to-noise ratio—thus, a better reconstruction of the motion—than high-dynamic motions.

5.2. Extraction Parameters

Unlike prior approaches for keyframe extraction, our method has two parameters that can be used to control the extraction process. As discussed, prior solutions assign fixed weights to bones or joints of the skeleton, with limited control of the extraction results. In our approach, we have investigated the effect of the two parameters on the resulting choice of keyframes.

The first parameter k, which is used in the Dimension Reduction step, describes the number of dimensions desired to express the high-dimensional motion capture data. Increasing k increases the number of clusters; as a result, the number of total keyframes will increase in the corresponding motion. The value of k is important to capture sufficient significant information from the original data—high dynamic motions need a greater number of dimensions, not to miss important changes in various joints of the skeleton. As shown in Figure 6(a) and (b), there is a significant decline in error during the reconstruction of motions when $k > 7$. Even for a low dynamic and slowly changing type of motion as walking, there is a great quality loss when k is low and σ is high. The result of our empirical tests shows that $k \in [7,10]$ ensures a satisfactory visual quality.

The second parameter σ is used to describe the number of neighbouring frames to be taken into consideration during filtering and clustering steps. Increasing σ smoothens the Gaussian filter and thus creates larger clusters, resulting in a smaller number of keyframes for the motion. For better accuracy in the constructed motion, a smaller σ value must be used. We have tested a different σ range for high- and low-dynamic motions. Low-dynamic motions can have a greater value of σ, since there is no rapid change in the motion. Our tests show that $\sigma \in [7,10]$ give satisfactory signal-to-noise ratios, between 35 and 55 dB, for low-dynamic motions. As shown in Figure 6(c) and (d), for values of σ greater than 3, the error in reconstructed motion increases significantly, resulting in the loss of fast changes in the motion. Thus, high-dynamic motions become very unnatural and foot-skating problems start to appear. We recommend using $\sigma \leq 3$ for high-dynamic motions to ensure satisfactory visual quality.

5.3. Motion Curve Representation

We have also compared our algorithms with the two prior approaches, when the body coordinate space is used, instead of Euler angles. Table 1 shows the results when the three

Table 2. Peak signal-to-noise ratio values for reconstructed motions in Euler angle space.

	Stretching	Walking		Basketball	Boxing
No. of frames	4592	800	No. of frames	4905	3000
No. of keyframes ($\sigma = 10$, $k = 7$)	131	38	No. of keyframes ($\sigma = 3$, $k = 10$)	616	311
Saliency PSNR	42.90	64.27	Saliency PSNR	39.28	46.94
Curve simplification PSNR	40.20	49.42	Curve simplification PSNR	38.50	45.92
Frame decimation PSNR	46.42	68.26	Frame decimation PSNR	43.98	50.04

DOI: 10.1002/cav

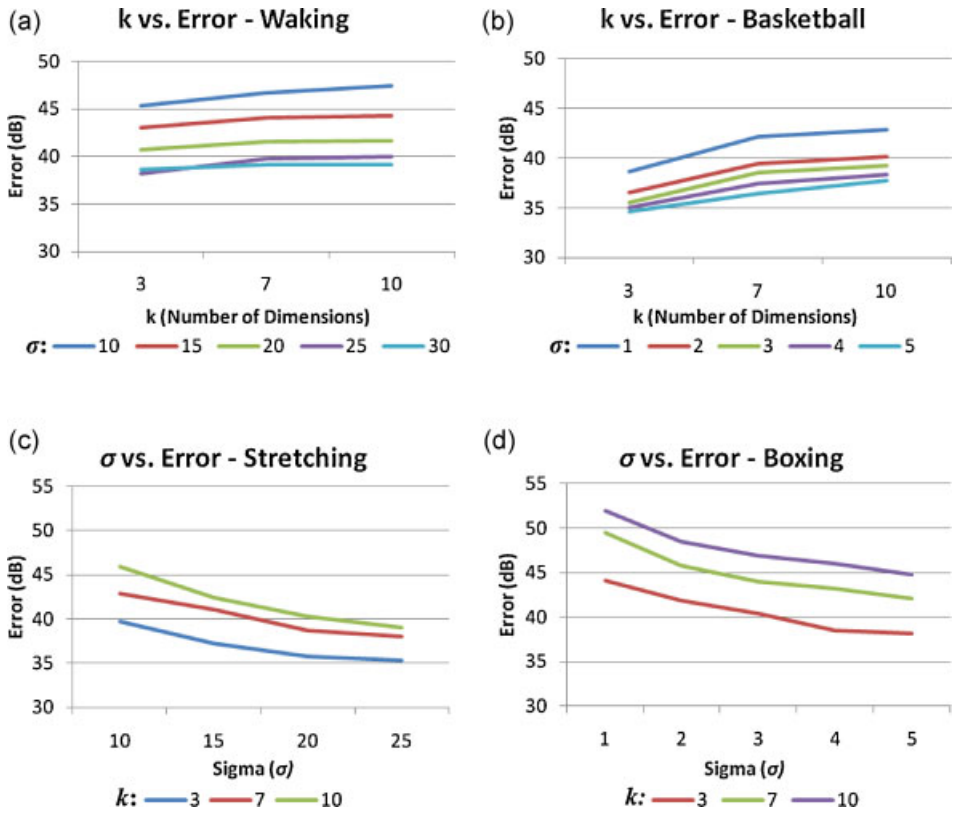

Figure 6. (a and b) Effect of dimension reduction on the overall error, for high-dynamic and low-dynamic motions. (c and d) Effect of parameter σ on the overall error, for high-dynamic and low-dynamic motions.

approaches are used. In the body coordinate space, each joint is represented by three DOFs, yielding a total of 90 DOFs. In the Euler angle space, a DOF is present only if the human body is capable of rotating that joint in that dimension, with a total of 62 DOFs. Because of the greater DOFs used in body coordinate space, better error rate is observed with this representation.

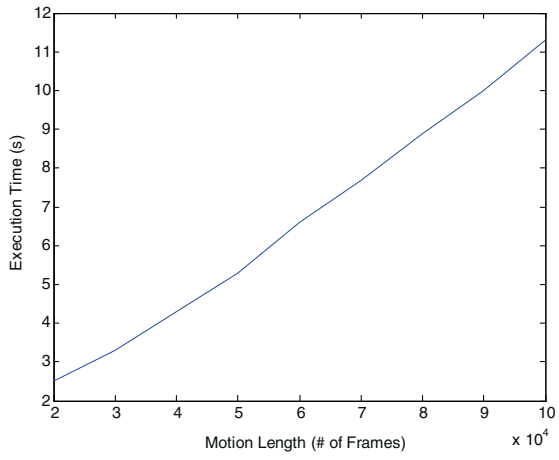

Figure 7. Keyframe extraction time of our method for different size of motions.

5.4. Performance

Figure 7 includes the running times for different size of motions. As shown in Figure 7, for a motion of 100 000 frame motions, with 120 captured frames/second, the process takes close to 11 seconds. Figure 7 shows that the running time of the method increases linearly with respect to the number of frames, and the method can be used effectively with a window of less than 20 000 frames. Thus, we can conclude that the method can also be used effectively in real-time solutions.

6. APPLICATIONS

Our proposed solution can be used in the scope of a variety of applications. One of the most suitable applications is as a tool for motion editing: our solution will allow the user to manually edit the motion by automatically selecting relevant keyframes. Kwon and Lee [26] use keyframes of motion capture data to incorporate rubber-like exaggeration to motion capture sequences. Witkin and Popovic [27] use keyframes of motion sequences data together with given constraints, to warp the original animation. Boulic *et al.* [28] use lowpass and bandpass pyramids to blend motion capture sequences using their keyframes. Our solution can

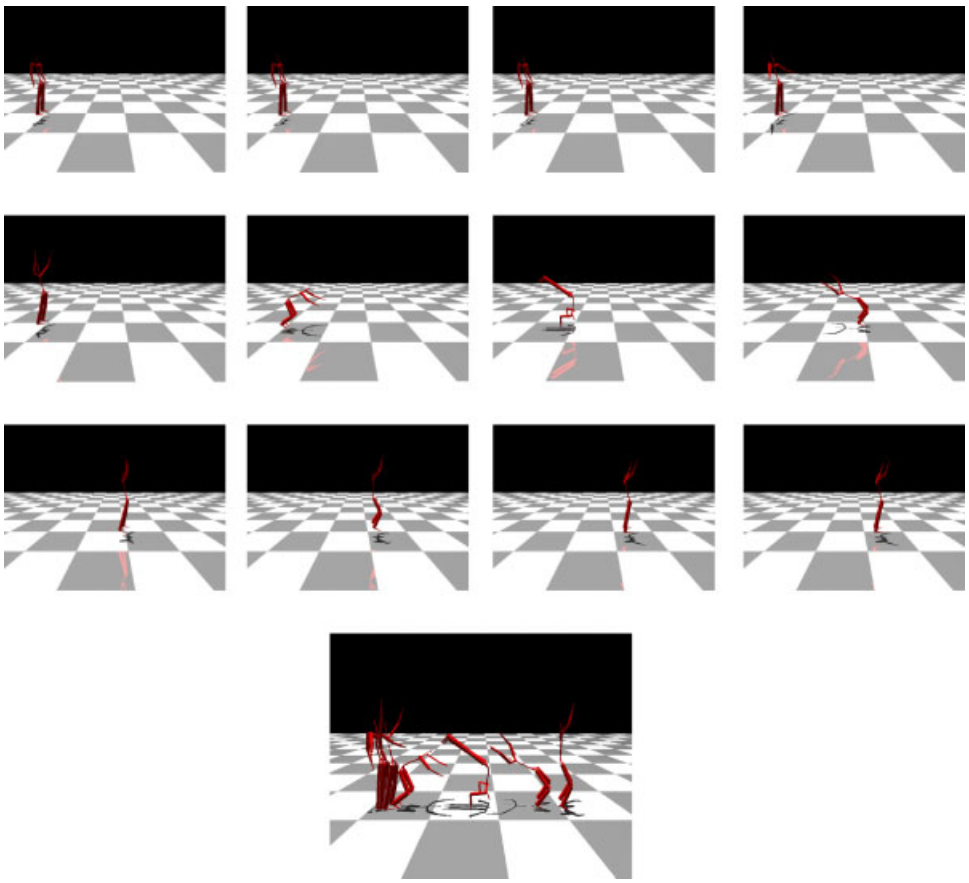

Figure 8. Keyframes of a flip motion and the thumbnail generated from the set of keyframes.

thus be used as a tool to improve these solutions by detecting keyframes in a motion automatically. Although extraction of important keyframes is important for motion editing, issues regarding foot skating and collision detections should be handled as a separate step while constructing the motion from keyframes. This aspect of motion editing is not in the scope of this article; Arikan's work handles this problem while compressing the motion capture databases. [29]

The second application of our approach is as a tool in motion capture data compression. Motion capture compression methods require a means of detecting the importance of frames in a motion capture sequence, to correctly capture the significant parts of the motion and compress the rest of the information with a lossy compression scheme [29,30]. Such a metric to capture the significance in motion can be constructed using our method.

Another possible application of our method is a tool for thumbnail generation of motion capture sequences. In a motion capture database with a large number of motions, a desired task would be to have previews of motions available as thumbnails. These thumbnails can be used by database queries, or for browsing with a quick preview of motion capture sequences. In Figure 8, we provide keyframes and the generated thumbnail for a flip motion, based on extraction of 12 keyframes from the original motion and concatenation of these into a single thumbnail image.

7. CONCLUSION

In this paper, we have proposed a new approach for keyframe extraction from motion capture sequences. Our method finds the candidate keyframes of the input motion via the new *motion saliency* metric. Motion saliency is calculated by taking the absolute difference between the Gaussian weighted averages of each point computed at different scales. Obtaining the candidate keyframes with this approach, we eliminate redundant keyframes in further steps. Based on the experimental results, we conclude that the method provides a fast solution to the keyframe extraction problem.

Because of the slight differences in keyframes for each joint, keyframe extraction using all joints in the body, instead of a single reference joint, results in an excessive number of keyframes. Instead of analysing every joint, we apply dimension reduction by PCA. This gives us local maxima and minima of many of the joints, therefore eliminates a lot of excess keyframes. The performance of our technique

Comp. Anim. Virtual Worlds 2011; **22**:3–14 © 2010 John Wiley & Sons, Ltd.
DOI: 10.1002/cav

also depends on the choice of σ (number of neighbourhood frames) and k (number of dimensions to analyse) value in computing saliency, which is done manually based on the input motion. There is no fixed formula for the selection of these parameters, but recommendations in the results section may be used as a reliable heuristic.

ACKNOWLEDGEMENTS

The data used in this project was obtained from mocap.cs.cmu.edu. The database was created with funding from NSF EIA-0196217. The authors thank Eyuphan Bulut for his help in initial implementation of curve saliency.

REFERENCES

1. Lim S, Thalmann D. Construction of animation models out of captured data. *Proceedings of IEEE Conference Multimedia and Expo*, 2002.
2. Lowe DG. Three-dimensional object recognition from single two dimensional images. *Artificial Intelligence* 1987; **31**: 355–395.
3. Li S, Okuda M, Takahashi S. Embedded keyframe extraction for CG animation by frame decimation. *Proceedings of IEEE Int. Conference on Multimedia & Expo*, 2005.
4. Togawa H, Okuda M. Position-based keyframe selection for human motion animation. *Proceedings of the 11th International Conference on Parallel and Distributed Systems–Workshops*, Vol. 2, 2005; 182–185.
5. Matsuda K, Kondo K. Keyframes extraction method for motion capture data. *Journal for Geometry and Graphics* 2004; **8**: 81–90.
6. Liu F, Zhuang Y, Wu F, Pan Y. 3D motion retrieval with motion index tree. *Computer Vision and Image Understand* 2003; **92**(2–3): 265–284.
7. Park M, Shin S. Example-based motion cloning. *Computer Animation and Virtual Worlds* 2004; **15**(3–4): 245–257.
8. Gong Y, Liu X. Video summarization using singular value decomposition. *Computer Vision and Pattern Recognition* 2000; **2**: 174–180.
9. Cooper M, Foote J. Summarizing video using non-negative similarity matrix factorization. *IEEE Workshop on Multimedia Signal Processing*, 2002; 25–28.
10. Haung K, Chang C. Key probe: a technique for animation keyframe extraction. *The Visual Computer* 2005; **21**: 532–541.
11. Pullen K, Bregler C. Motion capture assisted animation: texturing and synthesis. *SIGGRAPH'02*, 2002.
12. Coleman P, Bibliowicz J, Singh K, Gleicher M. A character motion representation for detail-preserving editing of pose and coordinated timing. *Symposium on Computer Animation*, July 2008.
13. Parent R. Computer Animation: Algorithms and Techniques, Morgan Kaufmann: San Francisco, 2001.
14. Inman V, Ralston H, Todd F. Human Walking. Williams & Wilkins: Baltimore, 1981.
15. Zhao J, Badler NI. Real Time Inverse Kinematics with Joint Kimits and Spatial Constraints. Technical Report MS-CIS-89-09, University of Pennsylvania: 1989.
16. Alexa M, Müller W. Representing animations by principal components. *Computer Graphics Forum* 2000; **19**: 411–418.
17. Glardon P, Boulic R, Thalmann D. PCA-based walking engine using motion capture data. *Computer Graphics International* 2004; 292–298.
18. Sattler M, Sarlette R, Reinhard K. Simple and efficient compression of animation sequences. *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, 2005.
19. Barbic J, Safanova A, Pan J-Y, Faloutsos C, Hodgins JK, Pollard NS. Segmenting Motion Capture Data into Distinct Behaviors. Graphics Interface: Ontario. 2004.
20. Itti L, Koch C, Niebur E. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1998; **20**(11): 1254–1259.
21. Lee C, Varshney A, Jacob D. Mesh Saliency. *ACM Transactions on Graphics* 2005; 659–666.
22. Bulut E, Capin T. Keyframe extraction from motion capture data by curve saliency. *CASA* 2007, Hassel University Press, 63–67.
23. Weisstein E. Curvature. 2009. From *MathWorld*–A Wolfram Web Resource: mathworld.wolfram.com/Curvature.html
24. Carnegie Mellon University. 2009. Retrieved 2009, from Carnegie Mellon University–CMU Graphics Lab–Motion Capture Library: mocap.cs.cmu.edu
25. Lim S, Thalmann D. Key-posture extraction out of human motion data by curve simplification. *Proceedings of 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society* 2001; **2**: 1167–1169.
26. Kwon J-y, Lee I-K. Rubber-like exaggeration for character animation. *15th Pacific Conference on Computer Graphics and Applications*, 2007.
27. Witkin A, Popovic Z. Motion Warping. *Proceedings of SIGGRAPH*, 1995; 105–108.
28. Boulic R, Thalmann D. Motion editing using multiresolution filtering. *International Conference on Multimedia Modeling*, 1999.
29. Arikan O. Compression of motion capture databases. *Computer Graphics Proceedings, Annual Conference Series*, 2006.

30. Gu Q, Peng J, Deng Z. Compression of human motion capture data using motion pattern indexing. *Computer Graphics forum* 2009; **28**: 1–12.

AUTHORS' BIOGRAPHIES

Cihan Halit is a M.S. Student in Bilkent University, Department of Computer Engineering. His research interests include computer animation and mobile software development. He is currently working on 3DPHONE, a EU funded Project by EC 7th Framework Programme. Contact him at halit@cs.bilkent.edu.tr.

Tolga Capin is an assistant professor at the Department of Computer Engineering at Bilkent University. He has received his PhD at EPFL (Ecole Polytechnique Federale de Lausanne), Switzerland in 1998. He has more than 25 journal papers and book chapters, 40 conference papers, and a book. He has 3 patents and 10 pending patent applications. His research interests include networked virtual environments, mobile graphics, computer animation, and human-computer interaction.

DOI: 10.1002/cav