Multi-scale Cartoon Motion Capture and Retargeting without Shape Matching

Mohammad Rastegari
Computer Vision Group,
Institute for Research in Fundamental Science(IPM)
Rastegari.mohammad63@gmail.com

Niloofar Gheissari
Faculty of Engineering and Industrial Science,
Swinburne University
Ngheissari@swin.edu.au

Abstract

This paper presents a novel approach for cartoon motion capture and retargeting. Unlike known approaches this method does not rely on finding corresponding points on shape contours. We assume movements have two components, rigid motions and non-rigid deformations. We capture rigid motions by using a similarity transforms. Non-rigid deformations are encoded by the Fourier transform and a key-shape interpolation. An important aspect of the proposed retargeting method is that it is performed in a multi-scale framework which helps us preserve the target's visual style. We have applied the proposed algorithm to a set of test images and results show very good performance of the proposed method.

1. Introduction

Motion capture is the process of recording movements of a source character and retargeting it to another character (in order to animate it). General surveys on the motion capture problem are [1,2]. Although motion capture based on live performances has attracted considerable attentions in the past few years [3], little work has been done on cartoon motion capture. In fact, to the best of our knowledge only 2 research papers have been published specifically on cartoon motion capture [4, 5]. However, there are a number of papers addressing 2D and 3D shape interpolation [11 ,12 ,13]. There is also some work on re-using traditional animation: for semi-automatic segmentation and inbetweening [14].

Cartoon motion capture extracts the motion style from old cartoons. It facilities the process of reusing old cartoons to generate new animations that have the same motion style as the old ones but in a visually different style. It can reduce the cost of animation studios and computer game producers significantly. It is also an important tool in editing current animations where the visual look of a character needs to be changed without changing the motion style.

This paper was motivated by the work of Bregler et al [4]. However, we compute motion transformations and deformations differently and apply retargeting in a multi-scale framework. In section 2 the necessary background material from [4] is recalled.

The proposed method is described in detail in section 3. To capture the non-rigid deformations, we first interpolate a given source shape by a linear combination of Fourier descriptors of source key-shape contours as in [10]. This eliminates the need for finding corresponding points. Consequently we achieve better control of the visual than is possible with the methods in [4] or [5].

In section 4, experimental results showing the superior performance of the proposed scheme are described.

2. Background and Literature review

In [5] Wang and Li propose a method for cartoon motion capture using global affine transformations and local non-affine deformations. *The strength of their approach lies in the ability to capture and retarget cartoon motion by shape matching [8] without a prior model.* However their method has two drawbacks. Since they compute the motion between every two successive frames and assign it to the target character, there is significant accumulation of error over the entire sequence. Secondly, if the target and source shapes are very different then the local deformation parameters will be very inaccurate.

The method of Bregler et al [4] is more relevant to the present work. Here the animation is represented as an interpolation of key shapes. They represent the animation as an interpolation of key-shapes (similar to facial expression animation applications [6,7]). The motion is parameterized by a combination of affine transformations and key-shape interpolations. Cartoon retargeting applies this information to animate the

target characters. The result is an animation with a new visual look but the same motion style as of the original cartoon. Since the present work is based on [4], here we present a more detailed and technical account of the process.

2.1. Affine Transformation

In order to determine the affine transformations describing a motion, one first extracts contours of each shape and then represents them by key points. Let $\boldsymbol{s}$ be the $\boldsymbol{3\times N}$ array $S=(s_1,\ldots,s_N)$ where

$$s_i^T = \begin{bmatrix} x_i & y_i & 1 \end{bmatrix},$$

and s_i is the i^{th} key-point. The shape $\boldsymbol{S}$ is deformed to shape $\boldsymbol{V(t)}$ at time frame $\boldsymbol{t}$ with affine parameters $\boldsymbol{\theta(t) = [a_1, a_2, a_3, a_4, d_x, d_y]}$:

$$V = warp(\theta, S) = \begin{bmatrix} a_1 & a_2 & d_x \\ a_3 & a_4 & d_y \end{bmatrix} \cdot S \qquad (1)$$

$\boldsymbol{a_1, a_2, a_3}$ and $\boldsymbol{a_4}$ encode rotation and scale parameters, and $\boldsymbol{d_x}$, $\boldsymbol{d_y}$ denote the translation. In order to retarget this motion, it is necessary to determine corresponding points for S and S', and V and V' where superscript ' denotes the target object.

2.2. Key-shape Deformation

Affine transformations are not adequate for describing frequently occurring deformations such as contractions extensions in cartoons. To overcome this difficulty the deformation of a shape is represented by a linear combination of several key-shapes and their affine transformation. These shapes are picked by the user, and should include all possible extreme deformations.

$$V = warp(\theta_1 \ldots \theta_k, S_1 \ldots S_k) = \sum_k (w_k \cdot \theta_k \cdot S_k) \qquad (2)$$

The cartoon shape $\boldsymbol{V}$ is now parameterized by $\boldsymbol{7\times K}$ variables, the $\boldsymbol{6\times K}$ affine parameters (as described previously), and the $\boldsymbol{K}$ key-shape interpolation weights.

2.3. Contour capture

Assume the input is a sequence of cartoon contours: $\boldsymbol{V(1),\ldots,V(t)}$ and the hand labeled key-shapes $\boldsymbol{S_1,\ldots,S_k}$. To estimate the motion vector $\boldsymbol{\theta}$ that approximates the contour input, the following optimization problem defined by:

$$Err = \| V - warp(\theta_1 \ldots \theta_k, S_1 \ldots S_k) \|^2 \qquad (3)$$

Affine Capture: Affine motion of contours is estimated by can be done with a closed-form solution. We estimate affine motion by minimizing the following error function:

$$Err_{aff} = \| V - \begin{bmatrix} a_1 & a_2 & d_x \\ a_3 & a_4 & d_y \end{bmatrix} \cdot S_K \|^2 \qquad (4)$$

The standard least-squares solution is

$$\theta_K = \begin{bmatrix} a_1 & a_2 & d_x \\ a_3 & a_4 & d_y \end{bmatrix} = V \cdot S_K{}^T (S_K \cdot S_K{}^T)^{-1} \qquad (5)$$

Key-weight capture: The set of key-weight $\boldsymbol{w_i}$ are estimated by minimizing the full approximation error defined as:

$$Err = \| V - \sum_k (w_k \cdot \theta_k \cdot S_k) \|^2 \qquad (6)$$

The result might be improved if one imposes additional constrains on the key weights $\boldsymbol{w_k}$ as in [4]. For this purpose, the additional constraints we imposed are

- **Constraint 1:** Only $\boldsymbol{J}$ interpolation weights are non-zero. This ensures that each possible shape lies in a smaller (local-linear) $\boldsymbol{J}$ dimensional subspace.
- **Constraint 2:** All key-weights add to 1.

The minimization in (6), which involves both linear and nonlinear terms, is implemented by quadratic programming [9].

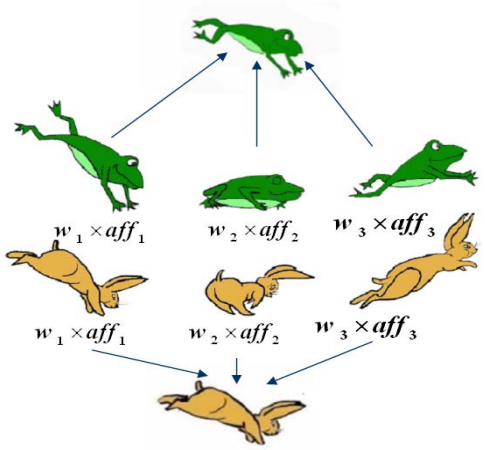

Figure 1- This figure illustrates how the motion from source shape is captured and retargeted to the target shape

Figure 1 shows a global view of the algorithm proposed by Bregler et al [4].

2.4. Retargeting

Bregler et al [4] use corresponding points to directly apply the affine and non-affine transformations to the target image. However, for deformations given by key-shape models, they estimate the interpolation function describing the target shape in terms of target key-shapes.

3. The Proposed Method

Although the algorithm of Berglar et al. has shown to have satisfactory performance in a number of cases, its applicability to real situations is limited due to certain underlying assumptions. One assumption is that we have nearly perfect contour matching methods. However, in real world applications and for complex scenes with rich textures, we do not have satisfactory contour matching algorithms. In addition, the exaggerated motions in cartoons, which often include large deformations and displacements between consecutive frames, are not amenable to conventional matching algorithms. In general, contour matching is still an open problem in computer vision. If the contours are matched incorrectly, then the algorithm of Bregler et al. fails.

Another drawback of the Bregler et al. method is that it cannot be directly applied to highly articulated objects. This is because local deformation models cannot describe deformations of highly articulated objects sufficiently accurately. As a result they manually sub-part articulated objects to simpler parts, and then apply their method.

Following the idea of Berglar et al., we assume that each motion has two components, a rigid motion and a non-rigid deformation. However, in the initial stage we limit ourselves to similarity transformation rather than the affine transformation as in [4]. The similarity transformation parameters are estimated by assuming a rectangular bounding box around each shape and estimating the similarity transformation for this rectangle.

To compute the non-rigid deformation parameters for an arbitrary shape, we first interpolate an input source shape by a linear combination of Fourier descriptors [10] of given source key-shapes. The weights for this interpolation are used to retarget the deformation to a new cartoon character. This is performed by applying the above weights to the parameters of Fourier descriptor of given target key-shapes. However, in cases where the texture is rich or there are large deformations, we will not have a satisfactory contour matching result. In fact, the exaggerated movements might result in a large and deformed displacement between two consecutive frames, which will cause any shape matching algorithm to fail.

To circumvent this problem we use a global feature (bounding box and Fourier coefficients) to explain the contour rather than a local feature. In other words, we look at the contour as a set of unordered points. In addition, we compute the deformation transform in a coarse to fine (multi-scale) framework. This enables the algorithm to preserve the visual style of a target after the retargeting process. This multi-scale framework is an important contribution of our proposed method. The details of the algorithm are as follows:

3.1. Similarity Transformation

To compute the similarity transformation, we construct the bounding box that encloses the shape's contour as seen in Figure 2.

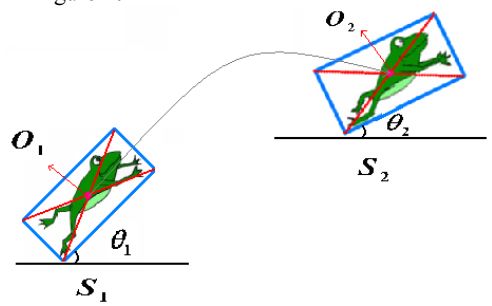

Figure 2- Bounding box used for computing multi-scale similarity deformation.

To compute the similarity matrix, we introduce the following five transformation matrices T_{O_1}, T_{O_2}, M_1, M_2, R:

$$T_{O_1} = \begin{pmatrix} 1 & 0 & -O_{1x} \\ 0 & 1 & -O_{1y} \\ 0 & 0 & 1 \end{pmatrix}, \quad T_{O_2} = \begin{pmatrix} 1 & 0 & -O_{2x} \\ 0 & 1 & -O_{2y} \\ 0 & 0 & 1 \end{pmatrix}$$

$$M_1 = \begin{pmatrix} \max 1 & 0 & 0 \\ 0 & \max 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

$$M_2 = \begin{pmatrix} 1/\max 2 & 0 & 0 \\ 0 & 1/\max 2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\max 1 = \max(|[x_i]| or |[y_i]|) \ in \ T_{O_1} \times S_1.$$
$$\max 2 = \max(|[x_i]| or |[y_i]|) \ in \ T_{O_2} \times S_2.$$

$$R = \begin{pmatrix} \cos(t) & -\sin(t) & 0 \\ \sin(t) & \cos(t) & 0 \\ 0 & 0 & 1 \end{pmatrix}, t = \theta_1 - \theta_2. \qquad (7)$$

Now one can estimate the similarity coefficients using:

$$Sim = -T_{O_1} \times M_1 \times R \times M_2 \times T_{O_2} \quad (8)$$

$$S_1 = Sim \times S_2 . \quad (9)$$

Next we define five transformations such that their combination is a similarity transformation. In fact, T_{O1} and T_{O2}, respectively translate S_1 and S_2 into the origin **(0,0)**. Transformations M_1 and M_2 scale S_1 and S_2 respectively by ***max1*** and ***1/max2*** times the original shape. Then R is a rotation matrix.

To transform S_2 into S_1, we first translate S_2 to the origin by T_{O2} and then scale it by M_2. Next we rotate the resulting shape to be aligned with S_1. Then transformation M_1 rescale it to the size of S_1. Finally, the transformed shape is shifted to the centre of S_1 using $-T_{O1}$. This procedure eliminates the need of finding point correspondences between two shapes.

3.2 Multi-Scale Similarity Deformation

To estimate the key-shape deformation, we use Fourier descriptor for describing the contours. A contour is defined as a set of pixels *S*.

$$S = \{(x_0, y_0), (x_1, y_1), \ \dots \ , (x_{N-1}, y_{N-1})\} \quad (10)$$

To each pixel, we assign a complex number as:

$$S(k) = x(k) + jy(k), \ k = 0, 1, ..., N-1. \quad (11)$$

Applying the discrete Fourier transform to S, we obtain:

$$a(\omega) = \frac{1}{N} \sum_{k=0}^{N-1} S(k) e^{\frac{-j 2\pi \omega k}{N}}, \quad (12)$$
$$\omega = 0, 1, ..., N-1.$$

Considering the contour as a set of frequencies, we can write:

$$S = \{a(0), a(1), \ \dots \ , a(N-1)\}. \quad (13)$$

In this representation, smaller values of ω represent more details of the boundary. In contrast, larger values of ω contain coarser and more general information about the boundary. This fact is illustrated in Figure 3(a) is the contour of a frog and we calculate the Fourier transform or coefficients $a(0), a(1), \ \dots \ , a(N-1)$ of this contour. We keep low frequencies, e.g., $a(0)$ and $a(N-1)$ and set the remaining coefficients equal to zero. We iterate this process to go from coarse to fine description of the shape. Figures 3-c to 3-p illustrate the process. The Fourier transform allows us to dispense with the corresponding points requirement and instead matches frequencies from low to high.

After the similarity transformations, Fourier transform is applied to each key-shapes S_i. Then the following error term (energy function) is minimized to find interpolation weights.

$$Err = \| V - \sum_i w_i \times S_i \|^2 . \quad (14)$$

For solving this optimization problem we use quadratic programming [9].

3.2.1. Preserving visual style of the target

In Equation (14) all levels of shape information (resulted from different values for ω) have the same weight. The key contribution of the weights is that it helps preserve the visual style of the target shape.

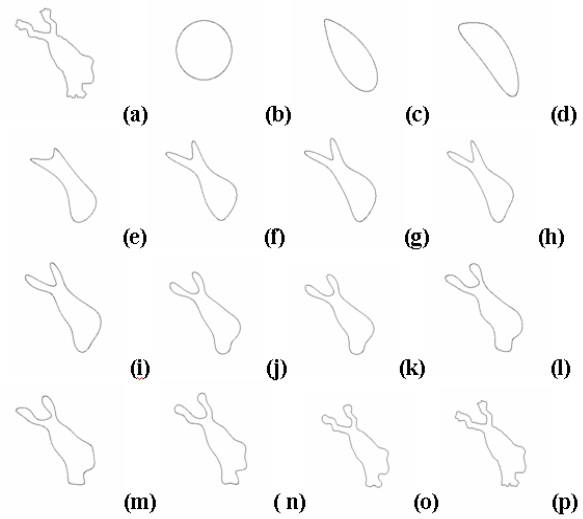

Figure 3-This figure illustrate how Fourier coefficients explain coarse and fine details about the contour (a) shows the original source shape. (b) shows the contour made by taking the Fourier coefficient with largest ω It results in most general information. (c)-(p) show the contour when we add the next largest coefficient. This test sequence is borrowed from Bregler ert al [4].

In the implementation of the algorithm, we calculate the weights separately for coarse levels of the pyramids scheme while for higher levels of details the weights can be calculated jointly. Thus equation (14) is replaced with equations of the form

$$Err_\omega = \| V(\omega) - \sum_i w_{\omega i} \times S_i(\omega) \|^2 . \quad (15)$$

For each level of detail ω, a quadratic programming algorithm is applied in each stage. The constraints specified at the end of section 2 are observed in the proposed scheme as well.

3.3. Outline of the proposed method

The outline of the algorithm is explained as below. This outline explains the motion capture and motion retargeting process.

1. Input Source-Key-Shapes as S_i ;
2. Input Target-Key-Shapes as T_i;
3. Input an arbitrary source shape as V;
4. For each source key shape
   a. Find similarity coefficient as Sim_i between V and key-shape S_i;
   b. Apply similarity transformation to the source-key-Shape: $SimS_i = Sim_i * S_i$;
5. For each transformed source-key-shape of step 4
   a. Compute Fourier transform as $FSimS_i$=DFT($SimS_i$);
6. Interpolate an arbitrary shape by a linear combination of transformed source shapes (step 5) using equation 15. This results in key weights.
7. For each target key-shape
   a. Apply similarity transformation to the target key-shape: $SimT_i = Sim_i * T_i$;
8. For each transformed target-key-shape
   a. Compute Fourier tranform $FSimT_i$=DFF($SimT_i$);
9. Apply the key-weighs to the Fourier transform of all target key-shapes: $FT = \sum_i (W_i * FSimT_i)$;
10. Produce target shape by taking inverse Fourier transform: T=iDFT(FT);

output T;

4. Experimental results

The proposed scheme was tested on a number of cartoons and four examples are provided below for demonstration purposes. As seen in Figure 4, we capture movements of a frog and retarget them to a rabbit. We select 3 key-shapes for the frog and the rabbit. Figure 5 shows a more complex test sequence. The movements of a jumping cartoon human character is captured and retargeted to a new human character. This is a very challenging problem as humans are highly articulated objects. To work with this complex object, Bregler et al. [4] first use a sub-part decomposition. They split a character into sub-parts. For example, they model the leg deformations separately from the arm or the head deformation. We did not need to apply any sub-part decomposition. The proposed method was applied directly to this sequence. The results are shown in Figure 5 illustrate the success of our method.

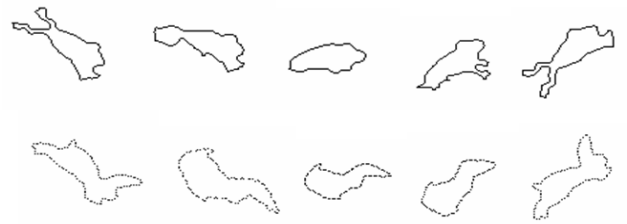

Figure 4-Tracking of frog jump and retargeting to a rabbit. This test sequence is borrowed from Bregler et al. [4].

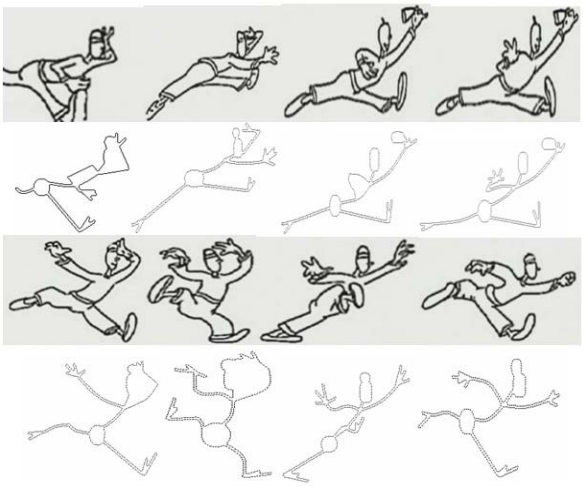

Figure 5-Motion of jumping cartoon human character and retargeted to a new character. This test sequence is borrowed from Bregler et al. [4].

The advantage of this method can be attributed to the fact that unlike that of Bregler et al., we use global features (bounding box and Fourier descriptors) that describe the contours at different levels of detail. In fact, using global Fourier descriptors and a multi-scale deformation computation, the proposed method is capable of being directly applicable to articulated objects. Working in a coarse- to-fine framework is a key factor in the successful application of the proposed method. More results are shown in Figure 6 and Figure 7. These results show that the proposed method can preserve the motion style of the source character very well. In Figure 6, the motion of a bear, Baloo, is assigned to a new abstract character. In Figure 7, Pink-panther's walking style is assigned to a new abstract character.

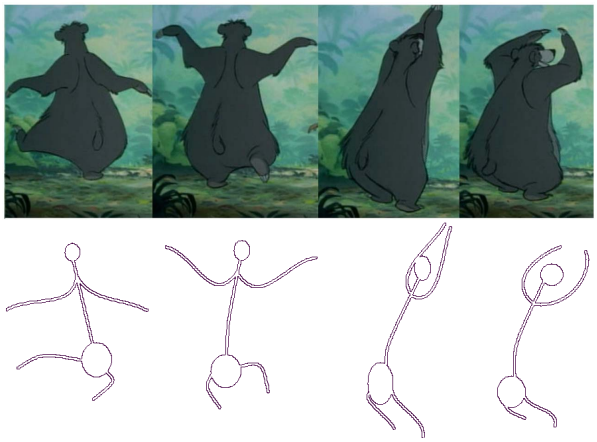

Figure 6- Motion of Baloo's dance and retargeted to a new character. This test sequence is borrowed from Bregler et al. [4], originally from The Jungle Book, Walt Disney company.

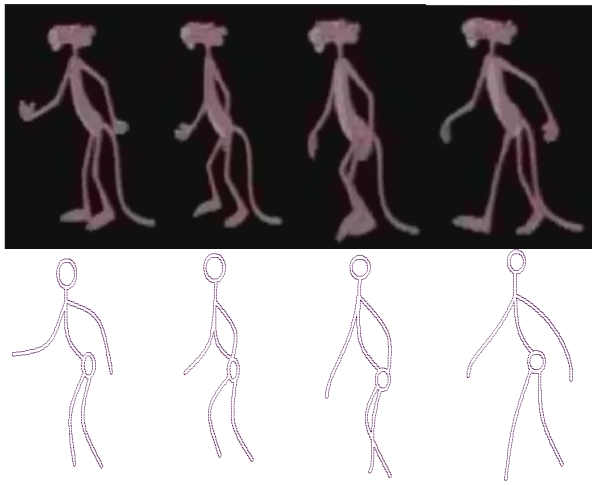

Figure 7- Motion of Pink-panther's walking and retargeted to a new character. This test sequence is borrowed from Bregler et al [4], originally from Pink Panther,Warner Bros Cartoons.

5. Conclusion

We have demonstrated a new technique that captures the movements of a cartoon character and retargets it to a different character. Unlike previous methods which rely on correspondence matching, our method only uses global features to capture both rigid and non-rigid motion. As a result, it eliminates the needs for point correspondences. This is achieved by tracking similarity motion and applying key-shape based interpolations on the Fourier descriptors of key-shapes.

This method has been tested on a number of image sequences and is shown to have good performance.

Acknowledgement

The authors are grateful for Professor Mehrdad Mirshams Shahshahani for his supports and conducts in all steps of this work.

6. References

[1] M. Gleicher. Animation from observation: Motion capture and motion editing. *ACM Computer Graphics*, 33(4):51–54, 1999.

[2] T. B. Moeshund and E. Granum. A survey of computer vision -based human motion capture. *Computer Vision and Image Understanding*, 81:231–268, 2001.

[3] M. Gleicher. Making motion capture useful. *Proc. ACM SIGGRAPH 2001 Course Notes 51*, 2001.

[4] C. Bregler, L. Loeb, E. Chuang, and H. Deshpande. Turning to the masters: motion capturing cartoons. *Proc. ACM SIGGRAPH 2002*, 2002.

[5] H.Wang, H.Li **Cartoon Motion Capture by Shape Matching,** *Proceedings of the 10 th Pacific Conference on Computer Graphics and Applications* , 0-7695-1784-6/02 © 2002 IEEE

[6] PARKE, F. 1972. Computer generated animation of faces. In *ACM National Conferences*, 451–457.

[7] PIGHIN, F., HECKER, J., LISCHINSKI, D., SZELISKI, R., AND SALESIN, D. H. 1998. Synthesizing realistic facial expressions from photographs. In *SIGGRAPH 98 Conference Proceedings*, Addison Wesley, M. Cohen, Ed., Annual Conference Series, ACM SIGGRAPH, 75–84. ISBN 0-89791-999-8.

[8] H. Chui and A. Rangarajan. A new algorithm for non-rigid point matching. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR00)*, 2:44–51, June 2000.

[9] GILL, P., MURRAY, W., AND WRIGHT, M. 1981. *Practical Optimization*. Academic Press, London, UK.

[10] D.S. Zhang, G. Lu, A comparative study of Fourier descriptors for shape representation and retrieval. *Proceedings of the Fifth Asian Conference on Computer Vision (ACCV02),* Melbourne, Australia, January 22–25, 2002, pp. 646–651.

[11] J. P. Lewis, M. Cordner, N. Fong, Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. *Proceedings of the 27th annual conference on Computer graphics and interactive techniques,* 2000, pp. 165 - 172.

[12] A. Hornung, E. Dekker, L. Kobbelt, Character animation from 2D pictures and 3D motion data. *ACM*

Transactions on Graphics (TOG), 2007, Volume 26 , Issue 1, Article No. 1.

[13] ALEXA, M., COHEN-OR, D., and LEVIN, D. As-Rigid-As-Possible Shape Interpolation. In *Proceedings of ACM SIGGRAPH 2000*, 157-164.

[14] C. N. de Juan, B. Bodenheimer, Re-using traditional animation: methods for semi-automatic segmentation and inbetweening. *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation,* 2006, SESSION: Animation systems. Pp. 223 - 232