GRADUAL COMBINATION OF FEATURE-BASED POLYMORPH

Pei-Pei Ou (歐珮珮), Soo-Chang Pei (貝蘇章), Chiou-Shann Fuh (傅楸善)
Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan 106

ABSTRACT

Morphing has become a popular technique in movie and entertainment industry. More amazing visual effects are expected in the future. There are many levels of correspondence specified by animators to control morphs. In the paper, we try to combine the advantages of image-processing methods for improving the main problems of morphing technique, such as more efficient feature-mapping methods and better performance. In addition, we propose novel techniques to smooth the boundaries and to designate certain features among different source images.

keywords: morphing, combination, feature, mapping.

1. INTRODUCTION

1.1. Morphing Technique

Morphing refers to an animation technique in which one image is gradually turned into another. A gradual animated transformation from one image into another, partially accomplished by moving certain points in the first image to corresponding points in the second image, in stage. Metamorphosis or morphing is a popular technique for visual effects. When used effectively, morphing can give a compelling illusion that an object is smoothly transforming into another. Given two images, the problem of constructing a metamorphosis from one image to the other has been extensively studied in computer graphics and image processing. Feature-based morphing [1] can be achieved by energy minimization or feature-based constraints. Beier and Neely [2] presented an elegant feature-based approach. The features may be points or lines. Mappings that have been used include linear mappings in conjunction with a weighting function that effectively controls the range over which a feature has influence.

1.2. Two-Dimensional Morphing

In this paper, we use a feature-based morphing technique for 2D objects. There are several methods to implement morphing process. This paper presents the method by convert image based on the relationship of a pair lines on two images. The method uses multiple pair of lines to transform pixels from source image to destination image. In addition, the transformation algorithm can be used to create intermediate frames between two frames between two frames. For each intermediate frame create a new set of lines by interpolating the lines from their position in the source image and destination image. Both images are distorted towards the position of the lines in intermediate frame [9]. These two resulting image are dissolved throughout the metamorphosis, so that in the beginning, the image is completely same as source image. Halfway through the metamorphosis it is halfway between the two images, and finally at the end it is completely destination image.

1.3. Three-Dimensional Morphing

It is possible to generate 2D images from a 3D model and apply 2D morphing algorithms to these. In this case, the intermediate stages of the morph are images. For many applications in animation and design, the 3D models themselves not their images, should be transformed. Furthermore, if the viewpoint or the lighting parameters are changed, the 2D morph has to be recomputed. However, 3D morphing is independent of viewing or lighting parameters. Having a 3D representation also allows the use of computer animation techniques such as key framing.

Three-dimensional metamorphosis [7] is a widespread technique in animation to generate a smooth transition from a source object into a target object. This research focuses on morphs between two genus 0 3D polyhedral objects. Most polyhedral morphing techniques consist of two main steps. The first involves finding the one-to-one correspondence between two polyhedral meshes [8]. The second involves defining the interpolation paths for each pair of corresponding vertices on the meshes to calculate the in-between shapes.

2. METHODOLOGY

2.1. Overview

In this paper, the inputs are more than two color images. Our work is closest in spirit to most feature-based morphing techniques. But, the overall system structure of the proposed approach is different from previous work. We present novel techniques in our design. The main procedures are listed below. First, the animator selects several corresponding line segments in source images to define a corresponding line segment in the morphed image. Second, by the way of backward warping, each pixel in the morphed image defines a coordinate mapping in source images. Third, given corresponding coordinates in the morphed and source images, we use cross-dissolving to interpolate the color of the corresponding pixel in the morphed image.

2.2. Field Morphing

Field morphing mainly takes advantage of line segments drawn on corresponding features in the source and destination image by the animator. Therefore, we try to find corresponding relation between pixels in the source and destination images by the relation of corresponding positions.

2.2.1. Backward Warping

There are two kinds of warping. In forward warping, points move from source to target image. Backward warping is inverse and its algorithm is mentioned below.

$$u = \frac{(X-P)\cdot(Q-P)}{\|Q-P\|^2}$$

$$v = \frac{(X-P)\cdot Perpendicular(Q-P)}{\|Q-P\|}$$

$$X' = P' + u\cdot(Q'-P') + \frac{v\cdot Perpendicular(Q'-P')}{\|Q'-P'\|}$$

If line *PQ* in the destination image and *P'Q'* in the source image are given, we can define a coordinate mapping form the destination image pixel coordinate *X* to the source image pixel coordinate *X'*. Where *u* is the corresponding position of the projective point of *X* in *PQ* (normalized), and *v* is the vertical distance from *X* to *PQ* (un-normalized). The algorithm makes use of *u*, *v* and *P'Q'* to find pixel *X'* in the source image corresponding to pixel *X* in the destination image. Then, repeat descriptive algorithm in every pixel in the destination image to get the complete corresponding relation between source-destination pixels.

In this part I try to exploit forward warping [3], but the result is error-prone. The reason is that forward warping is based on the source image to find the corresponding points in the destination image. As a result, every point in the destination image does not necessarily have a corresponding one in the source image and forward warping produces some pixels without appropriate transformation and lightness. Conversely, backward warping can show the better result image in which each point has an appropriate correspondence in the source image.

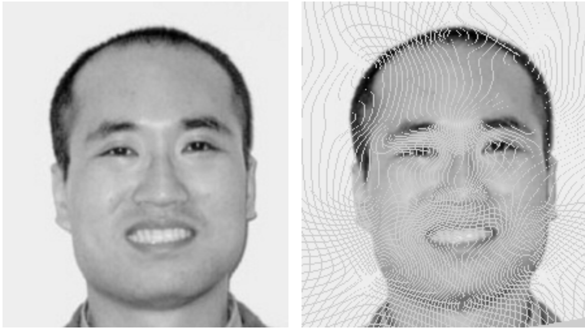

Figure 1: Forward warping

2.2.2. Effect of Parameters

As the content of the image turns complex, there are more than one line segments in an image. In such circumstances, pixel *X* has corresponding relation in every line segment. It means that many *X'* are taken. Consequently we must know how each line segment affects the same pixel *X*. That is computing the applicable weighting caused by all *X'*s and the only corresponding pixel *X'* which has appropriate corresponding relation to each line segment. The algorithm is as follows:

For each pixel *X* in the destination
    DSUM = (0,0)
    weightsum = 0
    For each line P_iQ_i
        calculate *u*,*v* based on P_iQ_i
        calculate X'_i based on u,v and $P_i'Q_i'$
        calculate displacement $D = X_i' - Xi$ for this line
        dist = shortest distance from *X* to *PiQi*
        weight = $(\text{length}^p / (a + \text{dist}))^b$
        DSUM += D_i * weight
        weightsum += weight
    X' = *X* + DSUM / weightsum
    destinationImage(*X*) = sourceImage(*X'*)

From the algorithm, the length of every line segment *PQ* and the distance between *X* and *PQ* are taken as the relative corresponding parameters to deicide the degree of weighting. As the length of *PQ* is larger, the distance between *X* and *PQ* is smaller and the relation between *X* and such line segment is stronger. Noticeably weighting has three adjustment parameters: *a*, *b*, *p*. *a* is to adjust distance. *b* is to adjust the influence of *X* and each line segment dist on weighting. It also means that as *b* is larger, pixel *X* is just affected by the closest line segment. In case of farther line segment, as *b* is larger, the influence of the line segment drops down quickly. *p* is to adjust the influence of *PQ* length on weighting. As $p = 0$, the length of each line segment has the same influence on pixel *X*. As $p = 1$, the length of the line segment is proportional to the weighting. Therefore, we assign parameter $p = 0$, $a = 1$, $b = 2$ in the following examples.

2.2.3. Color Interpolation

After warping, we start to fill pixel color in the morphed image by the corresponding pixels in the source and destination images [11]. In addition, we propose a new way to improve performance in the boundaries between features of different images. In order to find corresponding *Xs* (the pixel corresponding to the source image) and *Xd* (the pixel corresponding to the destination image) to record color, the method is that respectively utilizing the warping of the morphed image to the two different source images. If the mapped coordinate *Xs* and *Xd* are float numbers, we directly do **round-off truncation**.

In the condition of the corresponding coordinate found by the warping function exceeding the image region, we propose three solutions. Suppose that the corresponding coordinate of the source image exceeds the image region and the respective corresponding coordinates in the source, destination and morphed image are *srcP*, *destP*, *interP*.

1. We define the value (R, G, B) of *srcP* as (0, 0, 0) and take the interpolation between the value and the color of *destP* to be the color of *interP*.
2. We directly assign the color of *interP* to the color of *destP*.
3. In the source image, we replace *srcP* with the point *srcP'* which is nearest to *srcP*. After that, we take the color interpolation between *srcP'* and *destP* as the color of *interP*.

The effect of solutions 1 and 2 is not good enough because there is an apparent joint between the pixels in region and over region. Although the pixels over region in solution 3 are vague, it looks more natural than solutions 1 and 2. At last we adopt solution 3 to the problem.

As a result, if the mapped *Xs* and *Xd* exceed image region, the number is directly truncated on the image boundary. Originally we plan to make use of the around integer numbers to record color by interpolating on those pixels which are float numbers, but the way of directive round-off truncation we propose can preserve more apparent boundary without resulting in more fuzzy region.

2.3. Transition Control

Instead of taking uniform blending totally, we also attempt to generate morphed image with transition control [4]. The motive is that the requirement to designate which feature comes from the source image and which feature comes from the destination image, such as a combination of *A*'s nose and *B*'s mouth. The difference in the algorithms between transition control and uniform blending is that the former uses the procedure of employing the designate line segments on the source images to interpolate the corresponding ones on the morphed image. Besides, the key point is the desire for smoother color-controlling [10]. At last, we go a further step to exercise improved transition control with designate features among multiple source images.

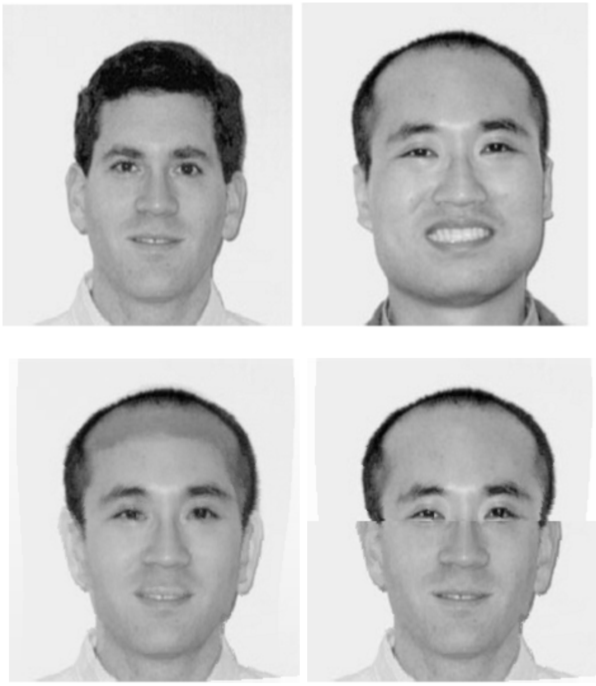

Figure 2: Conventional transition control

The figure on the upper left and upper right are the original images. In the conventional transition control, some drawbacks manifestly affecting the result images follow. Because of the great difference of hair positions of two source images, the hair contours of the result image are fuzzier in the lower left image. The boundaries between features from different source images are apparently inconsistent in the lower right image.

In this part, we try to improve conventional transition control by making consistent color transition without generating apparently inconsistent boundaries and fuzzy contours. In order to solve the above problems, we produce result images by changing color proportions of different source images gradually. In the novel transition control, the line segments of different source images have different weights to different places of the morphed image. In addition, the color changes proportionally to different coordinates. The way we adopt is to **change proportion gradually**. For example, under the condition of only two source images, we increase 0.1 weight of one source image and decrease the same amount of weight of the other source image every more loop. By way of this, the boundaries between features are undetectable and the contour of the morphed image is clearer. Moreover, even if the sizes of the faces in the source images are quite different, we can still make a natural result image with becoming face size.

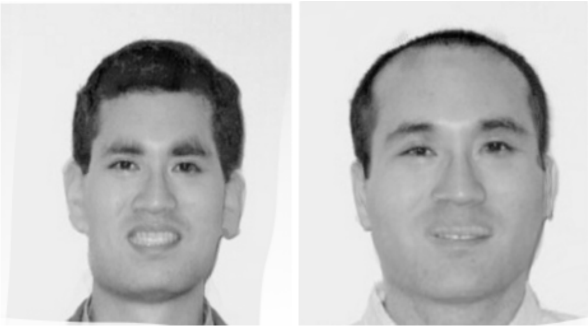

Figure 3: Novel transition control

In Figure3, we can clearly see that the left image consists of the upper part of the white man and the lower part of the yellow man. The right image is combined inversely. Therefore, by changing proportion gradually, the performance of transition control gets better.

2.4. Animated Sequence

It is very directly perceived through the senses to make a running movement film [6] in the morphing effect. First of all, Animator must get two sets of image sequence *A&B* to change *A* into *B*. Otherwise, drawing corresponding feature indicating lines in the corresponding image at the same time point. By this way, we can get every morphed image in middle. If we want to gradually change a face sequence of *A* to a face sequence of *B*, we need to draw specific feature line segments (ex. eyes) in every picture of *A* and *B*. We make use of corresponding source images to interpolate the line segments of the morphed image in middle afterward. Taking note of that interpolating proportion constant turns larger with the latter morphed image sequence time. Such a way is directly viewed and easy to achieve, but it relatively costs animator much time to decide line segments one by one. In order to decrease animator's work, we can choose certain key frames to be line segments in the image sequence, and the remaining is done by interpolation.

2.5. Polymorph

Polymorph [5] allows several images to be merged simultaneously. It treats a composite image as a metamorphosis of selected regions in several input images. In morphing between multiple images, non-uniform blending was used to derive an in-between image in which blending rates differ across the image. The framework for polymorph includes non-uniform blending of features in several input images.

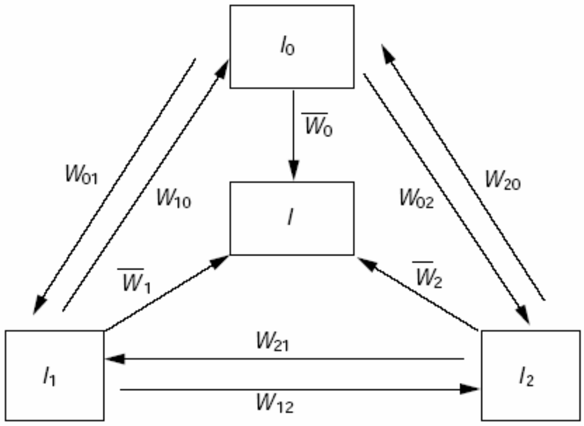

Figure 4: Warping function for three input images

The above picture is the essence of morphing among multiple images. The direction of arrows represents the morphing direction from the source image to the destination image. The warping procedure in the three images follows.

$$\overline{W_i} = \sum_{j=1}^{n} b_j W_{ij}(p)$$

$$\overline{I_i}(r) = \overline{W_i}(p) \bullet bili(p)$$

$$I(r) = \sum_{i=1}^{n} \overline{Ii}(r)$$

We first compute the warping function from every picture in the triangle vertex to the middle image. The method of computing the warping function is combining the warping functions multiplied by the corresponding coefficients in Barycenteric coordinate in the three images. Therefore, we can get the warping function of each image in the triangle vertex. Every image in the triangle vertex is multiplied by the corresponding warping function and weighted coefficient. Finally, we get the morphed result of the three images.

To follow the thought of the first transition control, assigning the origin of each feature, we can gradually deduce polymorph. Polymorph is a morphing which consists of multiple images and makes a morphed image with features of multiple images at the same time. The basic concept of uniform in-between image algorithms follows.

1. Suppose we have n source images. If we want to make a uniform in-between morphed image, we must first find the warping corresponding relation (called warping function) between certain images

with all other images. Therefore, we totally get n^2 warping functions.

2. As a result of our need to indicate what we want to see in the morphed image, the desired result must be composed of n images by a certain proportional synthesis. The proportion is a blending vector which indicates the specific weight of each image in the morphed image. Then each image based on the blending vector does a weighted average to get a new warping function. The function is exactly the warping function from the image to the morphed image.
3. The remaining work and field morphing which we do on two images earlier is the same. Making use of the warping function from each image to morphed image, we get the X pixel in the morphed image and the corresponding pixel in each image. Afterward we do a cross-dissolving of color and implement color interpolation proportion by a blending vector.

But the above-mentioned morphing result is based on the proportional transformation of the whole image. If we want to decide certain features of certain specific images, we need to change the blending function. The concept is similar to that of changing the same image with different interpolation proportion constant in the transition control.

Every pixel in each image must have a corresponding blending vector to all source images. Consequently we can control certain features of the morphed image coming form the certain image. But such direct mapping costs too much time. The result image is probably worse than what we expect because there is no a proportional blending. The effect is not real in visual proportion. If we first find a uniform in-between image (or called central image) with uniform blending and then make use of the central image to find different feature corresponding relation, we actually do the blending between the central image and all source images. Therefore, we get morphed image by the corresponding relation between the central image and source images. Taking notice of such a way, the number of warping function substantially decrease from original n^2 to 2*n. This is because we just need to compute the corresponding warping function from every source image to the central image and from the central image to every source image. In the end we get 2*n warping functions.

2.6. Feature-Based Polymorph

On the topic, we use above improved morphing techniques and propose a novel extended feature-based polymorph. First, we designate certain features from certain source image and then align the source images according to whose coordinates of features from low to high row order. For example, if we want A's mouth, B's eyes and C's nose in one face, the input image sequence is B, C, A. Afterward we run improved transition control and adjust the weight of each source image with time. Noticeably, the range of weight value ranges between zero and one.

Before execution, we separate the overall computational time complexity into N stage (N is the number of input source images). In the first stage, we gradually increase the weight of the first input source image from zero to one and keep the weight of the remaining source input image in zero. After that, in the second stage, the weight of the first input source image decreases and the weight of the next input source image increases complementarily from zero to one. In the end, the boundaries between features of different source images are fuzzier and the performance of skin color is more natural.

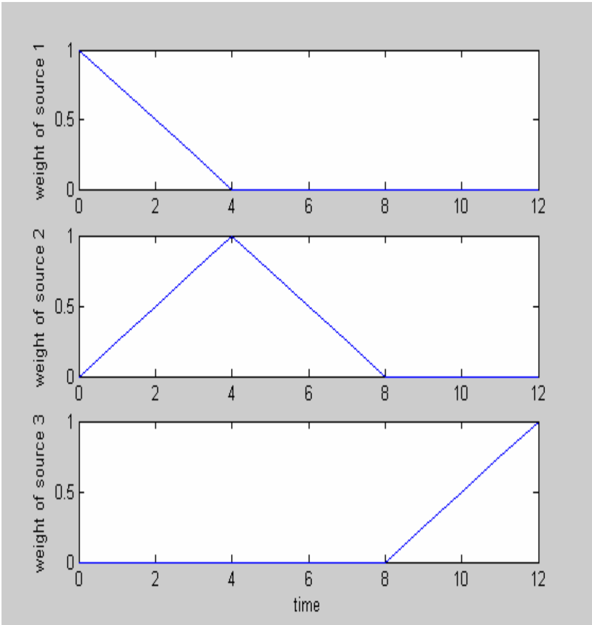

Figure 5: The distribution of weight changes with time in different source images.

The critical concept is that only two source images gradually change proportions in a complementary way at each stage. In order to avoid asymmetric result face in the morphed image, we take uniform in-between blending to make an appropriate facial contour image in the beginning. Then, the position and size of facial features in the morphed image would look more suitable. Furthermore, the number of source images is raised to meet practical requirement for image processing. We can better performance of multi-image morphing through above explicit approaches described in the paper. By this method, the morphed image is smoother among different features of input images. Besides, at any time point, the sum of total weight of all input images must be one.

3. EXPERIMENTAL RESULTS

The experimental result shows that more line segments result in better result image. In addition to the amount of line segments, good line matching is also an important issue. Computational complexity might be the bottleneck of the proposed algorithm. In the beginning, we employ uniform blending to get the intermediate morphed image of the multiple different source images. Then, we use corresponding line segments on different source images to interpolate pixels in the morphed image. The technique of feature-based polymorph is exploited to generate the result image. In the following pictures, the processing images are cropped first to focus on features of facial images.

Figure 6: Feature-based polymorph

The figures on the left column are source images. The figures on the right column are result images by feature-based polymorph. Apparently, the result facial images can be generated to have their eyes, nose, mouth derived from three different input faces.

The main problem of feature-based polymorph is unnatural result images. It is caused by unsuitable positions and apparent color transition between boundaries of features of different source images. In order to solve asymmetric result images, we adjust the size of different features on source images. First, uniform blending is used to decide appropriate positions and sizes of different features on the morphed image. Second, we interpolate color on the morphed image by gradual combination of source images. By the way, the result images look more natural.

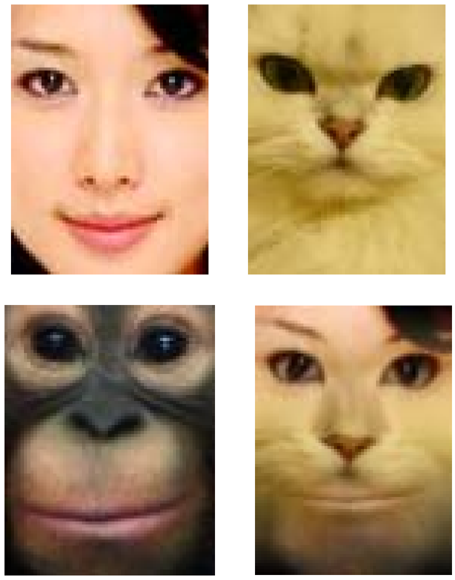

Figure 7: Feature-based polymorph of different races

It is a known fact that feature-based polymorph of different kinds of source images is more difficult. Because of the great variety in skin color, shape of facial features, corresponding positions between facial features, making appropriate morphed image is hard. In Figure 7, the figures on the upper left, upper right and lower left are source images of different races. The figure on the lower right image is the morphed image by the combination of woman's eyes, cat's nose and orangutan's mouth. In addition, the key challenge of feature-based polymorph of different races is the demand of suitable feature position and smooth color transition.

4. CONCLUSIONS AND FUTURE WORK

Field Morphing algorithm is based on line segments to count corresponding relation. Consequently, the decision of line segments is significant of the result image. As corresponding line segments are not chosen precisely, the ghost goes out easily. That is obscure image appears in the contours. For the reason, assigning line segments more precisely gains better corresponding relation. As a result of the function of field morphing, the influence of line segments is global. It sometimes produces undesirable effects, so animator must also pay attention to the decision of the line segments depending on those positions and lengths. Therefore, the fit result image comes. If the background color of the chosen source image is simple, the effect is be tter. For the reason, background changes shape with line segments. If the background color is not simple, the background of the result morphed image is also complex and twisted.

The amount of corresponding line segments plays an important part in the effect of morphing. If the amount of corresponding line segments is fewer, the image which is the result of morphing tends to be the average result of two images. If the amount of corresponding line segments is larger, we can eliminate ghost to get a better morphing result. However, as the corresponding lines increase, it will heavily increase the computational complexity. This is caused by the algorithm of feature line pairs, in which we must consider each pixel in the destination image to decide its corresponding coordinate in the source image by every pair of corresponding line segments. Take a picture whose size is 320*240 for example, as more pair of corresponding line segments increases, more 320*240*N computations (N is the necessary computational complexity to compute each the warping map of each pixel.). It also delays the program-executing speed and the smooth of morphing. Therefore, the amount of corresponding line segments depends on the tradeoff between the effect and smooth of morphing.

We present techniques for computing feature-based polymorph. These techniques were proven to be performed well and are comparable to or better than the conventional morphing. There are several opportunities for further work expanding on the method we propose. For example, we plan to design an extended morphing method that will be able to automatically adjust the best -fitting parameters. In addition, the content of the image is important. If the angles of two photographs are similar, we can get a better result. As the size of the source and destination images is different, our program still runs correctly. However, it must be more careful to draw corresponding line segments to avoid the appearance of ghost. The computational complexity is relative to the amount of line segments. Therefore, it is hard to achieve real time. At last good user interface is significant to improve the convenience of drawing line segments.

5. ACKNOWLEDGEMENT

This research is supported by National Science Council, R.O.C., under Contract NSC93-2213-E-002-059.

6. REFERENCES

[1] A. Lerios, C.D. Garfinkle, and M. levoy, “Feature-Based Volume Metamorphosis,” *Proc. Siggraph* 95, ACM Press, NewYork, 1995, pp. 449-456.

[2] T. Beier and S. Neely, “Feature-Based Image Metamorphosis,” *Proceedings of SIGGRAPH*, Chicago, Illinois, Vol. 26, pp. 35-42, 1992.

[3] Wolberg, G., “Digital Image Warping,” *IEEE Computer Society Press*, 1990.

[4] M. Alexa, “Local Control for Mesh Morphing,” in *Proceedings of the International Conference on Shape Modeling & Application*, p. 209, *IEEE Computer Society*, 2001.

[5] S. Y. Lee, G. Wolberg, and S. Y. Shin, “Polymorph: Morphing among Multiple Images,” *IEEE Computer Graphics and Applications*, Vol.18, pp. 58-71, 1998.

[6] P. Litwinowicz and L. Williams, “Animating Images with Drawings,” *Proc. Siggraph* 94, ACM Press, New York, 1994, pp. 409-412.

[7] Kaul, A., Rossignac, J., “Solid-Interpolating Deformations: Constructions and Animation of PIPs,” *Proceedings of EUROGRAPHICS ‘91*, September 1991, pp. 493-505.

[8] Kent, J., Parent, R., Carlson, W. “Establishing Correspondences by Topological Merging: A New Approach to 3-D Shape Transformation,” *Proceedings of Graphics Interface ‘91*, June 1991, pp. 271-278.

[9] S.-Y. Lee et al., “Image Metamorphosis Using Snakes and Free-Form Deformations,” *Proc. Siggraph* 95, ACM Press, New York, 1995, pp. 439-448.

[10] D.A. Rowland and D.I. Perrett, “Manipulating Facial Appearance through Shape and Color,” *IEEE Computer Graphics and Applications*, Vol 15, No. 5, 1995, pp. 70-76.

[11] S. Lee, G. Wolberg, and S.Y. Shin, “Scattered Data Interpolation with Multilevel B-splines,” *IEEE Trans. Visualization and Computer Graphics*, Vol. 3, No. 3, 1997, pp. 228-244.