

AUTOMATIC FACE MORPHING FOR TRANSFERRING FACIAL ANIMATION

The Duy Bui Mannes Poel Dirk Heylen Anton Nijholt
Department of Computer Science
University of Twente
The Netherlands
email: {theduy,mpoel,heylen,anijholt}@cs.utwente.nl

ABSTRACT

In this paper, we introduce a novel method of automatically finding the training set of RBF networks for morphing a prototype face to represent a new face. This is done by automatically specifying and adjusting corresponding feature points on a target face. The RBF networks are then used to transfer the muscles on the prototype face to the morphed face. The automatic adjusting of the feature points on the target face is done by Genetic Algorithms. The fitness function used in the GA expresses the difference between the surface of the morphed face and the target face. We also present an algorithm to calculate this function fast.

KEY WORDS

Facial Animation, Geometric Modelling, Radial Basis Functions, Genetic Algorithms

1 Introduction

Facial animation aims at producing realistic facial expression to enrich human computer interaction. Various techniques have been introduced to animate the 3D faces, including: interpolation ([1]), parameterizations ([2]), finite element methods ([3], [4]), pseudo muscle models ([5]) and physics based muscle models ([6], [7], [8]). One of the issues related to these techniques is how to transfer the animation from a given face to a newly created face model. Several approaches have been proposed to transfer the facial animation, including: transferring the motion vectors ([9]), transferring the Facial Animation Table (FAT) ([10]) to a newly created face, and transferring the muscles and using a morphed version of a prototype face to represent the new face ([11]).

Noh and Neumann [9] have used Radial Basis Function (RBF) networks to find correspondences of a source face's vertices on a target face in order to transfer the animation motion vectors. The RBF networks mapping uses the specification of corresponding feature points on the two face models. Some heuristics for feature detection are presented to reduce the number of feature points which have to be selected manually. Mani and Ostermann [10] use B-splines with weights to clone MPEG-4 facial animation tables (FAT) from a source face model to a target face model. Manual selection of corresponding points between source and target faces is required. Additional manual adjustment

of B-splines weights is also required to increase the correctness in mapping the MPEG-4 FATs, however this correctness is still not warranted. Kähler et al. [11] transfer both the head model, the underlying muscle, and the bone structure to a new face based on the interpolation of two sets of feature points.

All the approaches above require heavy human involvement to specify and adjust the correspondences between the source face (prototype face) and the target face. Normally, it may take 10-20 minutes (cf. [11]). In this paper, we introduce a novel method to automatic specify and adjusting corresponding points on the two faces. We use RBF networks to morph a prototype face to represent a new face with the specified corresponding points. The RBF networks are then used to transfer the muscles on the prototype face to the target face. The automatic adjusting of the feature points on the target face is done by Genetic Algorithms. The fitness function used in the GA expresses the difference between the surface of the morphed face and the target face. We also present an algorithm to calculate this function fast.

There are several advantages to the use of a morphed face to represent a new face over that of the new one itself. First, we can keep a fixed model of face with fixed number of vertices and polygons when achieving facial animation for different faces. Second, tags on the fixed model can be reused, for example tags for the jaw rotation and eyelid rotation. Third, the regions information on the face can be reused to improve the performance of the vector muscle and to control the animation (cf. [12]).

Section 2 gives an overview of our approach. Section 3 describes the RBF networks. An error function to assess the difference between the two faces is presented in Section 4. Section 5 presents how the correspondences are adjusted to minimize the error function. Finally, some results are presented in Section 6.

2 Overview

We morph a prototype face to represent a target face using RBF networks with pairs of corresponding feature points on the prototype face and the target face. The feature points on the prototype face are fixed, while we search for the feature points on the target face to minimize the difference between the morphed face and the target face. An overview

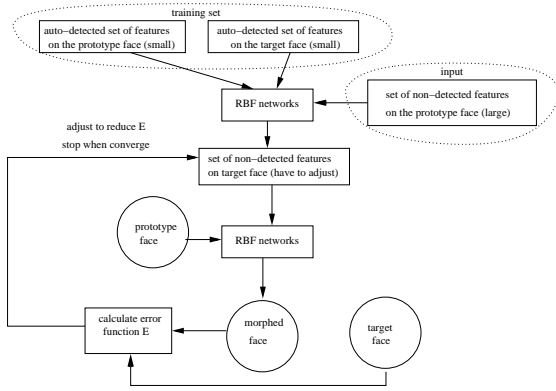


Figure 1. Overview of the system

of the approach can be seen in Figure 1.

We use a muscle based prototype face [12] with given feature points. Most feature points are specified manually except several ones that can be detected automatically to cover all the features of the face. These feature points on the prototype face are determined only once and are reused for every new target face. For the feature points on the target face, we first determine several easy-to-detect points, namely the top of the head, the tip of the nose, and so forth. The auto-detected feature points will stay fixed while the rest will be adjusted (for convenience let us call these non-detected feature points). We use the auto-detected feature points and their correspondences on the prototype face as the training set for the RBF networks to determine the initial version of the non-detected feature points on the target face. A morphed version of the prototype face is created by RBF networks with the features points on the prototype and target face. The non-detected feature points on the target face are then adjusted with Genetic Algorithms to minimize the difference between this morphed face and the target face.

3 Radial Basis Functions Networks

The RBF networks are known for their powerful interpolation capability and are often used for face model fitting [9].

For each coordinate a different RBF network is defined, the transformation has the form

$$(x, y, z) \rightarrow (RBF_1(x, y, z), RBF_2(x, y, z), RBF_3(x, y, z))$$

where each RBF_i is given by

$$RBF_i(x, y, z) = \sum_{j=1}^n w_{i,j} h_{i,j}(x, y, z)$$

where the $w_{i,j}$ are weights of the network that need to be determined or learned on the basis of a training set. For the basis functions $h_{i,j}$ we follow the successful approach

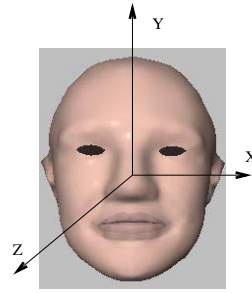


Figure 2. The prototype face (4650 polygons)

given in [9]:

$$h_{i,j}(v) = \sqrt{\|v - \mu_{i,j}\|^2 + s_{i,j}^2}$$

where $\mu_{i,j}$ is the center and $s_{i,j}$ is given by

$$s_{i,j} = \min_{k \neq j} \|\mu_{i,k} - \mu_{i,j}\|$$

This choice for $s_{i,j}$ leads to smaller deformations for widely scattered center points and larger deformations for closely located points.

Each of the RBF networks is trained using a set of coordinates of feature points on the prototype face and the corresponding feature points on the target face. In order to prevent overfitting and improve generalization a regularization term $\sum_j w_{i,j}^2$ is added to the error term for each RBF_i , cf. [13].

4 The Error Function

We assess the difference between the morphed face and the target face by calculating the distance between sampling points on the prototype face and their projections on the target face using Cylindrical Projection (cf. [9]). These sampling points evenly distribute over the prototype face. We ignore the neck when taking the sampling points to avoid any unnecessary mis-fitting between a part of a head and another head's neck. We also ignore the back of the head to concentrate on the front part of the head. We will now describe how to determine these sampling points and then present the error function.

For simplicity, we place all the faces in the same coordinate system, as can be seen in Figure 2.

Let $V_{tophead}$ denote the top head vertex of the morphed. This vertex should also be the top head vertex of $Face_{target}$ as the top head vertex is one of the feature points on the two heads (and RBF network accurately mapped these feature points). Let l denote the line that goes through $V_{tophead}$ and is parallel with the Y axis.

First we take m sampling planes P_i which go through l . P_i is determined by the angle α between P_i and the plane YOZ :

$$\alpha = \frac{i}{\pi} \text{ with } i = 1..m$$

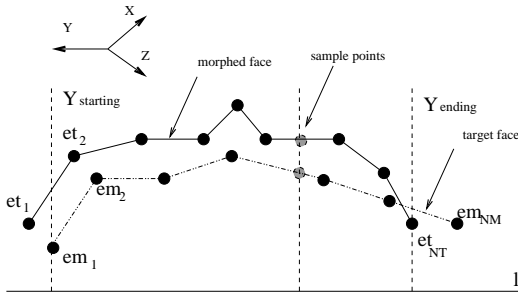


Figure 3. How sampling points are located on the intersection between a sampling plane and the surface of target face and mapped face

Each plane P_i intersects the surface of the morphed face and the target face at several edges¹. Let them be:

$$(em_{11}, em_{12}), (em_{21}, em_{22}), \dots, (em_{NM1}, em_{NM2})$$

and

$$(et_{11}, et_{12}), (et_{21}, et_{22}), \dots, (et_{NT1}, et_{NT2})$$

where em_{kt} and tm_{kt} are vertices in 3D space.

By filling all the holes on $Face_{target}$ and $Face_{mapped}$, these edges form two continuous curves, which can be seen in Figure 3:

$$em_1, em_1, \dots, em_{NM}$$

and

$$et_1, et_2, \dots, et_{NT}$$

Note that all these vertices lie on the same plane P_i .

We cut off these curves to have their starting vertices and ending vertices having the same Y coordinate $y_{starting}$ and y_{ending} . We give a big penalty to the part of the curves that are outside of this range. We then take n sampling vertices SM_{ij} and ST_{ij} on each of these two curves, which are determined by their Y coordinates:

$$Y \text{ coordinate of } SV_j = y_{starting} + \frac{j \times (y_{ending} - y_{starting})}{n}$$

It is easy to see that ST_{ij} is the projection of SM_{ij} onto the target face. Other methods can also be used here to determine the sampling points on the prototype face. However, it is time consuming to find the projections of sampling points on the target face. For our method, the calculation to determine the intersection between a sampling plane P_i and a face could also be painfully slow as it has to check if the plane intersect every polygon on the face. This would slow down the GAs process and make it impossible as for every candidate version of the features points on the source face, it has to calculate the error function again. We overcome this problem by traversing from the top head vertex down each face through the polygonal structure to find the intersections between the face and each plane P_i . As the top head vertex lies on the plane P_i , it will be the first vertex on the curve. Let us call it the “current edge” that intersects P_i (a point is a special case of an edge - zero length edge). We then check the intersection between the plane P_i with only edges that are connected to the “current edge”, and produce another “current edge”. We mark all the found

¹We consider only triangular face meshes

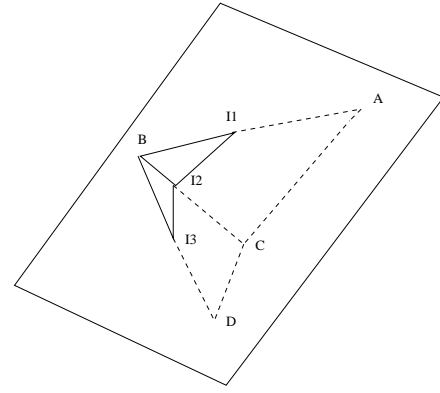


Figure 4. Intersections between the faces’s surface and a sampling plane

edges to prevent going back to previous edge. This process stops when P_i intersect no more edges that are unmarked and connected to the “current edge”.

The error function is then described as:

$$E = \sum_{i=1..n} \sum_{j=1..m} d^2(SV_{ij}(mappedface), SV_{ij}(targetface))$$

5 Adjusting the feature vertices

We use Genetic Algorithms (GA) to adjust the non-detected feature points on the target face to minimize the error function. Genetic Algorithms are search algorithms based on the process of natural evolution [14]. A Genetic Algorithm’s problem solving approach is, basically, to rate or to rank possible solutions as they are generated, and then use these rankings to guide further exploration for related solutions. For a more thorough introduction of Genetic Algorithms, the readers can consult [14] and [15].

The GA process starts with a set of solutions, which are represented as chromosomes. Each solution is a version of the features points on the target face, which is the modification of the initial feature points. In this modification, each feature point is any point inside the cube with specified length and the initial feature point as the center (see Figure 5). Solutions from one population are taken and used to form a new population. Solutions which are then selected to form new solutions (offspring) are selected according to their fitness - the more suitable they are the more chances they have to reproduce. In this case, the fitness function is the inverse of the error function that minimizes the difference between the morphed face and the target face.

The Chromosome A chromosome is a string of binary representation of all non-detected feature points on the target face. Each feature point is a vertex in a 3D space and can be described by three coordinate v_1, v_2, v_3 . Starting from an initial version of the point, numbers of versions

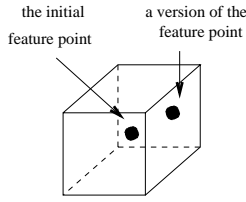


Figure 5. An initial feature point and a version of it

of the point are created by modifying each parameter in a specified range. Let $Rmin_1, Rmax_1, Rmin_2, Rmax_2, Rmin_3, Rmax_3$ be the ranges of values of these coordinate. The coordinate of a feature point can be represented as:

$$(p_1, p_2, p_3)$$

where $0.0 \leq p_i \leq 1.0$,

$$p_i = \frac{v_i - Rmin_i}{Rmax_i - Rmin_i},$$

$$v_i = p_i(Rmax_i - Rmin_i) + Rmin_i \quad i = 1..3$$

A part of chromosome representing a feature point looks like:

$$c_{11}, c_{12}, \dots, c_{1n}, c_{21}, \dots, c_{3n} \quad c_{ij} = 0 \text{ or } 1$$

where 2^n is used to convert a real number between 0.0 and 1.0 to a binary string². That means $c_{i1}, c_{i2}, \dots, c_{in}$ is the binary representation of $\lfloor 2^n p_i \rfloor$. We concatenate the binary representation of all feature points to form a chromosome.

The fitness function The fitness function is the inverse of the error function:

$$\text{fitness}(\text{solution}) = \frac{1}{E(\text{morphedface}(\text{solution}), \text{targetface})}$$

Crossover We use multi-point crossover, which can be seen in Figure 6. For multi-point crossover, several crossover positions are chosen at random with no duplicates and sorted in ascending order. Then, the variables between successive crossover points are exchanged between the two parents to produce two new offspring. The section between the first variable and the first crossover point is not exchanged between individuals. The idea behind multi-point, and indeed many of the variations on the crossover operator, is that parts of the chromosome representation that contribute the most to the performance of a particular individual may not necessarily be contained in adjacent substrings [16]. Moreover, the disruptive nature of multi-point crossover appears to encourage the exploration of the search space, rather than favoring the convergence to highly fit individuals early in the search, thus making the search more robust [17].

Mutation We start with a mutation rate of 0.3. We increase this mutation rate when error stays stable, and decrease it when the GA process produces smaller error (better result). This mutation rate is constrained to be in

²From our experience n ranging between 5 to 10 gives best convergence and best result for the GA

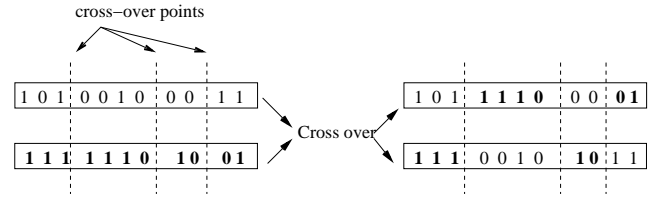


Figure 6. Multi-point crossover

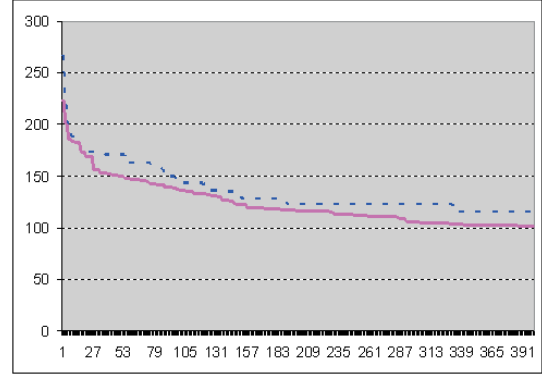


Figure 7. How the GA process converges: broken line - solutions are replaced by their projections onto the target face during the GA process; continuous line - projection is used only to calculate the error function and to generate the final solution

the range of 0.3 to 0.5.

Additional Operation We have put an additional operation to the GA process. The operation projects each solution of the GA to the target face's surface using Cylindrical Projection. We have tested two approaches to implement this operation. The first approach is to replace each solution of the GA process with its projection. How the GA converges then is shown by the broken line in Figure 7. The second approach is to use the projection of a solution only to calculate the error function and to generate the final solution. How the GA converges then is shown by the continuous line in Figure 7. As can be seen from the figure, the first approach causes the error to decrease and converge very fast, while the second approach causes the error to decrease and converge slowly. However, the first approach seems to end up at some local minima and cannot get out of this local minima. This happens due to the decrease of diversity of GA's solutions by replacing the solutions with their projections.

6 Result

Figure 8 shows the features points on the prototype face and the adjusted feature points on the target face. As can be seen from the figure, the feature points on the target face are adjusted to the right position. The morphed face gener-

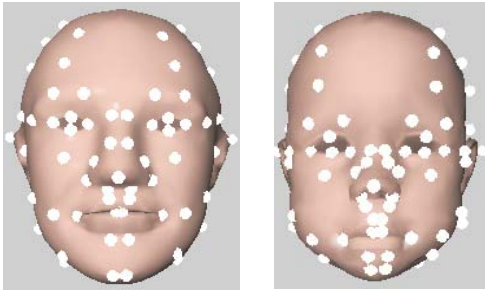


Figure 8. The features point on the prototype face (left) and the target face (right)



Figure 9. The morphed face (left) and the target face (4142 polygons) (right)

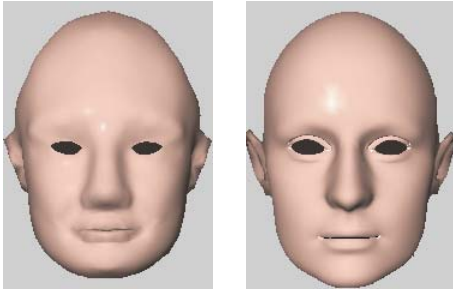


Figure 10. Another example of the morphed face (left) and the target face (7736 polygons) (right)

ated by RBF networks using these feature points is shown in Figure 9. The morphed face has the overall shape, forehead and cheek surface, chin shape as the target face. Eyes, nose and mouth are in correct position. The shape of the morphed face's lips, however, does not completely match the shape of the target face's lip. This is because the difference between two pairs of lips is hard to measure even if they look very different. Another example of the result is shown in Figure 10.

After the morphed face is created, the muscles are also transferred from the prototype face to the morphed face. Using these muscles, we can create facial expressions on the morphed face, which are shown in Figure 11.

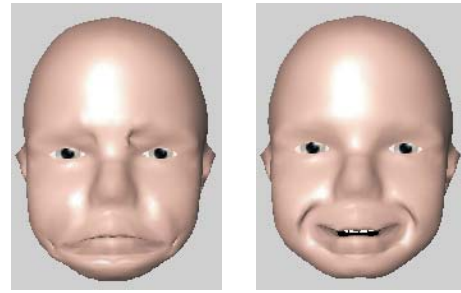


Figure 11. Facial expressions on the morphed face: sadness (left) and happiness (right)

7 Conclusion

In this paper, we introduced a novel method of automatically finding the training set of RBF networks for morphing a prototype face to represent a new face. This was done by automatically specifying and adjusting corresponding feature points on a target face. The RBF networks were then used to transfer the muscles on the prototype face to the morphed face. Genetic Algorithms were used to adjust the feature points on the target face to minimize the difference between the surface of the morphed face and the target face. We defined a fitness function to assess the difference between the two faces. We also presented an algorithm to calculate this function fast.

References

- [1] Parke, F. I. (1991), "Techniques for facial animation", In N. Magnenat-Thalmann and D. Thalmann (Eds.), *New Trends in Animation and Visualization*, John Wiley, Chichester, 229-241.
- [2] Cohen, M. M., & Massaro, D. W. (1993), "Modeling coarticulation in synthetic visual speech", In N. M. Thalmann & D. Thalmann (Eds.) *Models and Techniques in Computer Animation*, Tokyo: Springer-Verlag.
- [3] Rolf M. Koch, Markus H. Gross, Friedrich R. Carls, Daniel F. von Bren, Yoav I. H. Parish (1996), "Simulating Facial Surgery Using Finite Element Models", In *Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH*, pp. 421-428, 1996.
- [4] Rolf M. Koch, Markus H. Gross, Albert Bosshard (1998), "Emotion Editing using Finite Elements", *Computer Graphics Forum* 17(3): 295-302, 1998.
- [5] Kalra P, Mangili A, Magnenat-Thalmann N, Thalmann D (1992), "Simulation of Facial Muscle Actions Based on Rational Free Form Deformations", *Proc. Eurographics '92*, Cambridge, U.K.,

Computer Graphics Forum, Vol. 2, No. 3, pp. 59-69.

- [6] Khler, K., J. Haber and H.-P. Seidel (2001), "Geometry-based muscle modeling for facial animation", Proceedings Graphics Interface 2001, pp. 37-46.
- [7] Terzopoulos, D. and Waters, K., "Physically-Based Facial Modeling, Analysis, and Animation", The Journal of Visualization and Computer Animation, Vol.1, pages 73-80, December 1990.
- [8] Waters, K. (1987), "A muscle model for animating three-dimensional facial expressions", Computer Graphics (SIGGRAPH'87), 21(4), July, 17-24.
- [9] J.Y. Noh and U. Neumann. "Expression Cloning," Computer Graphics, Proceedings of ACM SIGGRAPH 2001, Los Angeles CA, August 2001, pages 277-288.
- [10] M. V. Mani, J. Ostermann, "Cloning of MPEG-4 face models", International Workshop on Very Low Bit rate Video Coding (VLBV 01), Athens, October, 2001.
- [11] Khler, K., Haber, J., Yamauchi, H., Seidel, H., 2002. Head Shop: Generating animated head models with anatomical structure, ACM SIGGRAPH Symposium on Computer Animation , pg 55-64.
- [12] Bui, T.D., D. Heylen & A. Nijholt (2003), "Improvements on a simple muscle-based 3D face for realistic facial expressions", in: Proceedings 16th International Conference on Computer Animation and Social Agents (CASA'2003), IEEE Computer Society, Los Alamos, CA, ISBN 0-7695-1934-2, 33-40.
- [13] Bishop, C.M. (1995), Neural Networks for Pattern Recognition, Clarendon Press-Oxford.
- [14] Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Publishing Company Inc.
- [15] Davis, L. (ed.) (1991). *Handbook of Genetic Algorithms*, Van Nostrand Reinhold.
- [16] Booker, L. (1987), "Improving Search in Genetic Algorithms", in Genetic Algorithms and Simulated Annealing, Davis, L., Ed. Pitman, London, and Morgan Kaufman: Los Altos.
- [17] Spears, W. M. and DeJong, K. A. (1991), "An analysis of multi-point crossover", in Rawlins, G. J. E., editor, Foundations of Genetic Algorithms-1, pages 301-315, Morgan Kauffman.