Feature Points Based Facial Animation Retargeting

Ludovic Dutreve* Alexandre Meyer† Saïda Bouakaz‡

Université de Lyon, LIRIS, France§

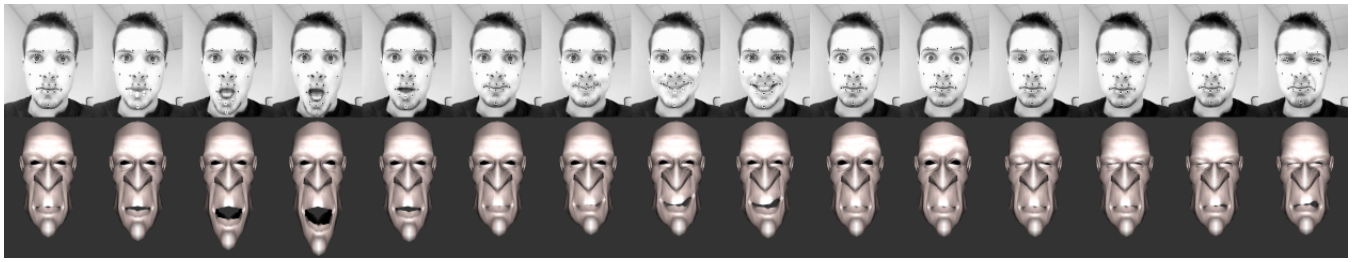

Figure 1: *The bottom row shows the virtual face animated by retargeting expressions from the source face (top row).*

Abstract

We present a method for transferring facial animation in real-time. The source animation may be an existing 3D animation or 2D data providing by a video tracker or a motion capture system. Based on two sets of feature points manually selected on the source and target faces (the only manual work required), a RBF network is trained and provides a geometric transformation between the two faces. At each frame, the RBF transformation is applied on the new feature points positions of the source face, resulting in new positions for target feature points according with the expression of the source face and the morphology of the target face. According to their displacements along time, we deform the target mesh on the GPU with the linear blend skinning (LBS) method. In order to make our approach attractive to novice user, we propose a procedural technique to automatically rig the target face by generating vertices weights for the skinning deformation. To summarize, our method provides interactive expression transfer with a minimal human intervention during setup and accepts various kind of animation sources.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling;

Keywords: Facial Animation, Retargeting, Performance-Driven Facial Animation, Skeleton-Subspace Deformation

*e-mail: ludovic.dutreve@liris.cnrs.fr

†e-mail: alexandre.meyer@liris.cnrs.fr

‡e-mail: saida.bouakaz@liris.cnrs.fr

§Laboratoire d'InfoRmatique en Images et Systèmes d'information, UMR 5205 CNRS, INSA de Lyon/Université Claude Bernard Lyon 1/Université Lumière Lyon 2/Ecole Centrale de Lyon

1 Introduction

1.1 Motivation

Facial animation is an important aspect in 3D environment featuring human characters and is highly demanded by many kinds of applications, such as 3D games, interactive human/computer softwares and movies. However, a good facial animation requires a lot of time for a skilled artist: the complexity of the human faces may provide subtlety expressions, and each spectator is able to know instinctively if an animation is correct or not. Since pioneering work of [Parke 1972], many efforts were provided to increase realism and satisfy the critical eyes of humans. Nevertheless, two aspects of facial animation are still stimulating active research: the process of rigging a face and the transfer of an expression from one person to another.

The process of rigging is tedious and requires many hours of manual work to an artist. Indeed, even simple method like shapes blending needs an artist to create the key shapes. The common skinning approach needs to manually define the joints (also called bones in the case of body) of the skeleton such as left of the right eyebrow, top of the lip, *etc*.on the mesh. And, then the artist needs to specify which parts of the surface are attached to which joint(s). More sophisticated models based on simulation of the facial tissues and muscles need also human interventions to attach tissue to skeleton and to tune complex muscles configuration. Furthermore, the animation produced for a specific face cannot be reused directly to another.

The second active aspect of facial animation domain is the interesting idea to produce realistic animation by transferring an existing one to a new character, instead of generating it from scratch. *Performance-Driven Facial Animation* systems introduced the idea of capturing animation of a real actor, often from 2D video(s) and transferring it to a virtual character. The transfer can also be applied from an existing 3D animation created by an artist. Another advantage of the transfer is to easily manage the amount of memory to store on sensitive applications like video games by storing only one expression for many face models. Despite all these advantages, the transfer suffers from an inherent difficult: the target model should still look like the original person, but smoothly reflect the source expression. The techniques of transfer have to adapt the source deformation to the target morphology.

1.2 Related Work

Many approaches exist to animate a face in real-time. Most common are linear weighting with blendshape models [Deng et al. 2006], skeleton-based with linear blend skinning [Magnenat-Thalmann et al. 1988; Lewis et al. 2000], physics-based muscles models [Kähler et al. 2001; Costa-Teixeira et al. 2006], *etc.* Blendshape consists on an artist to create the key shapes and then linearly blend between those shapes to obtain a fluid motion. However, obtaining an exhaustive set of key shapes is time and memory consuming, as the human expression is very rich. Linear blend skinning (LBS), also known as skeleton subspace is the process of binding the face mesh to a skeleton. Each vertex is assigned a set of joint influences and their associated weights. To deform the mesh, each vertex is rigidly transformed by all of its influences and the blending weights are used to compute the weighted average of these positions. Despite more recent and sophisticated approaches such as muscle-based approaches, skinning deformation remains the most popular method used in practice for real-time animations of characters. Mostly, because of its simplicity and efficiency (with simple GPU implementation). The process of obtaining influences can be initiated by an automatic procedure based on the vertex proximity to the bone, as the one proposed by Maya or Blender. Because this automatic approach lead to artifacts, accurate influences have to be tuned manually. Notice that for body, the *Pinocchio* system [Baran and Popović 2007] proposed last year an automatic rigging of 3D characters based on heat simulation. And dedicated to professional animators, [Costa-Teixeira et al. 2006] proposed a transfer of their complex rigging system based on muscle from a face to another. For all the advantages of linear blend skinning mentioned before, we chose to use it to deform our face and we improved it by a rapid and near automatic influences computation.

After Gleicher published in [Gleicher 1998] the idea of reusing an animation in the case of body, and since animating a face from scratch needs hard manual efforts, many authors proposed retargeting techniques [Deng et al. 2006; Vlasic et al. 2005; Na and Jung 2004; Pyun et al. 2003]. [Chai et al. 2003] presented an interesting technique consisting on using low-quality vision based 2D motion data to animate a face with high-quality 3D precomputed motion capture data. The Radial Basis Functions (RBF)[Powell 1987] were used to adapt motion vectors of a mesh to an other [Noh and Neumann 2001]. Motion vectors are the displacements of vertices along time. By putting some landmarks on the target and the source faces, a data interpolation provided by RBF allows to compute a dense matching between target vertices and source triangles. Vectors are modified in scale and orientation to adapt with the target morphology. But some limitations occur: the target animation is precomputed and cannot be used in a runtime application, vertices motion vectors for each frame require a high memory space. Furthermore, the whole expression is retargeted, without any possibility of tuning. And, the need of source motion vectors prevents the use of data from motion capture. Our approach may be seen as an extension to this work in order to accept any kind of source animation, to make the setup more attractive to novice user by providing an almost automatic rigging and by using the skinning technique to deform the target mesh which is more common on 3D engine.

A more complete state of the art of the facial animation retargeting problem was proposed by [Pighin and Lewis 2006]. We entrust [Deng and Noh 2007] to present a more general survey about the facial animation.

1.3 Overview

We present a method to transform an animation, obtained from an animated 3D face or a recorded 2D video, to another 3D face. An overview of our technique is illustrated in Figure 2. After initializing the source and the target faces by manually select between 15 or 30 feature points, an automatic procedural rigging is applied on the target mesh. Then, any deformation/animation applied to the source is transferred to the target mesh using our RBF approach applied on feature points. Finally, deformation of the target mesh is deduced from the transferred features points by linear blend skinning on the GPU. Thus, all the process of transfer and mesh animation is real-time. Notice that the manual work needed by our system is tractable to a novice user. Indeed, a simple mesh can be loaded, quickly rigged by positioning the feature points and then be animated in real-time according to the source animation, which can be a 2D video motion capture. To summarize, the main contributions of our system are:

- Quasi-automatic rigging of the facial mesh in neutral position;
- Real-time retargeting;
- Partial retargeting: user could choose to partially transfer the facial animation by selecting feature points;
- Compatibility with various animation sources: 2D video or 3D motions of feature points.

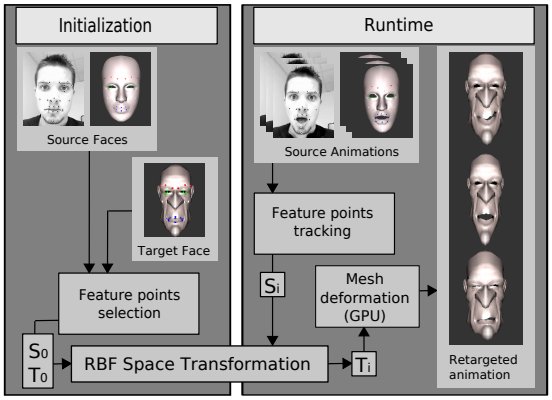

Figure 2: *Overview of our retargeting method.*

The remainder of this paper is organized as follows. In Section 2, we present our general usage of radial basis function in the context of space transformation. In Section 3, we describe our mesh deformation model and our automatic rigging. In Section 4, we present the retargeting method based on a RBF network and how we use it to adapt a motion from a face to an other. In Section 5, we present some results and conclude this paper.

2 RBF Space Transformation

Radial Basis Functions (RBF) are widely used in computer graphics for surface approximation and interpolation, but we may use them as a space transformation. In order to do this, we define two spaces with two set of feature points. Let $\mathcal{S}_0$ (resp. $\mathcal{T}_0$) be the set of source feature points (resp. target feature points) and N the size of these sets (see Figure 2). Each control points $\vec{s_i} \in \mathcal{S}_0$ has its dual $\vec{t_i} \in \mathcal{T}_0$. After the training of the RBF with these two sets of feature points, the RBF can transform a position from the source space to the target space.

The RBF equation is:

$$F(\vec{s_j}) = \sum_{i=1}^{N} \vec{w_i}.h(||\vec{s_j} - \vec{s_i}||) \quad (1)$$

with $h\left(||\vec{s_j} - \vec{s_i}||\right) = \sqrt{\left(||\vec{s_j} - \vec{s_i}||\right)^2 + sc_j^2}$ (multi-quadric function) and $sc_j = min_{j \neq i}\left(||\vec{s_j} - \vec{s_i}||\right)$. Training the network consists on solving the 3 linear systems of N equations (in the 3 dimensional case) such as

$$\vec{t_j} = F\left(\vec{s_j}\right) \tag{2}$$

Let H the matrix such as $H_{ij} = h\left(||\vec{s_j} - \vec{s_i}||\right)$ and $T_x = (\, t_1^x t_2^x ... t_N^x \,)^t$, t_j^x is the x coordinate of $\vec{t_j}$. Using equations 1 and 2, the system can be defined by

$$T_x = H.W_x \tag{3}$$

with the weights $W_x = (\, w_1^x w_2^x ... w_N^x \,)^t$. Then, to solve the system, we have to compute $W_x = H^{-1}T_x$. Once the RBF network is trained for each axe, the position in the target space $\vec{t}$ for each point $\vec{s}$ of the source space is obtained by applying the transformation $F\left(\vec{s}\right)$ provided by Equation 1.

3 Target Mesh Deformation

Animating a mesh involves using a mesh deformation technique. In our context, this deformation has to verify two constraints. First, it has to be computed easily in real-time. Second, it has to be guided by the deformations of a small number of feature points because our method aims to retarget animations providing by any kind of sources: 2D/3D tracking, 3D animation. To meet these constraints, we chose to use a skeleton-based deformation: linear blend skinning widely used by 3D engine for animating characters.

3.1 Facial Skinning

Linear blend skinning deforms a mesh according local transformations of the skeletal bones. In order to adapt skinning deformation technique with feature points displacements, we consider feature points as joints (bones). We define weights for each vertex such as $\sum w_i = 1$ with the i^{th} bone's weight $w_i \in [0, 1]$. A weight of 0 means that the feature point does not influence the current vertex, while a value's weight of 1 means that the current vertex is influenced only by this feature point. The position of the vertex $\vec{v}$ at frame f is defined by $\vec{v_f} = \vec{v_0} + \sum_{i=1}^{N} w_i.\vec{d_{if}}$. Where N is the number of feature points, and $\vec{d_{if}}$ is the displacement of the i^{th} feature point at frame f. We can notice that for body animation, transformations are defined by only joint's rotations mainly because of the limb length constant constraint. However, faces have specific small deformations which can be described by joints translations.

3.2 Quasi Automatic Weighting

To tackle the tedious process of influences assignment in skinning, we propose a quasi automatic approach. The only manual work required is positioning the feature points on the target face. Our solution is procedural. The main idea is to compute for each vertex an influence value (also called a weight) to a feature point according to their distance. This weight function has to respect some criteria like smooth borders and smooth gradient from 1 when the vertex is on the joint to 0 when the vertex is far from the joint. We base our weight function on the ellipsoid function which allows to define independent parameters for each axe since feature points influence area is not regular. Let $\vec{v_i}$ a vertex of the mesh, $\vec{t_j}$ a feature point position and $\vec{d} = \vec{v_i} - \vec{t_j} = (d_x, d_y, d_z)^t$. For each feature points, r_x, r_y and r_z are the radius of the ellipsoid for the 3 axes which are defined for each part or the face using feature points relationship. For example, the top borderline of the eyes is large compared to its height, so its influence is mainly spread over the width. For this case, r_x depends on distance between eye's corners, and r_y depends on the distance with the eyebrow. $\vec{v_i}$ is then influenced by the feature point $\vec{t_j}$ as follow: $w_{ij} = max\left(0, 1 - \left(d_x^2/r_x^2 + d_y^2/r_y^2 + d_z^2/r_z^2\right)\right)$. For each vertex, we compute a similar weight for each feature points. Then, we consider only the maximum weights with significant contribution and normalize them in order to have a sum of weights of 1. This simple method provides an automatic rigging. However, we have to admit that this technique can lead to lack of precision on some specific kind of face like non human faces. Thus, we offer to the user the possibility to modify and adapt the rigging if necessary by a painting process.

4 Animation Retargeting

In this section, we describe our animation transfer technique.The morphologies between two human faces can greatly vary, and much more when the 3D characters are cartoons, monsters or animals. Therefore, motions of some feature points on a source face along an animation can not be used directly. They have to be adapted in scale and in orientation. The RBF space transformation provides a good solution to this problem, feature points of the source face define the source space while feature points of the target face define the target space.

The original animation may come from motion capture or landmarks (labelled vertices) on an animated 3D face. In the case of video tracking, only displacements on the $\vec{x}$ and $\vec{y}$ directions are known. We assume that it is sufficient to describe a facial animation (human eye does not need 3D informations to recognize an actor's expression). Thus, we set the $\vec{z}$ coordinates of feature points to 0 when source animation is in 2D. Each feature points on the target model is tracked or followed in the source data. The retargeting method we describe now is illustrated in Figure 2. As transfer preprocess (usually at first frame), a RBF is trained between the 3D source feature points and the target feature points, both need to be in neutral expression. Let define $F_{RBF} = RBF\left(\mathcal{S}_0, \mathcal{T}_0\right)$, with $\mathcal{S}_0$ (resp. $\mathcal{T}_0$) the source (resp. target) feature points set (with a neutral expression). Then, at each frame f of the animation, coordinates of tracked feature points are adapted to the target virtual face by $\mathcal{T}_f = F_{RBF}\left(\mathcal{S}_f\right)$. Thus, target feature points move according to the expression of the source face and the morphology of the target face. The target mesh is then deformed on the GPU using the linear blend skinning as described in Section 3.

Our retargeting provides the capacity of transferring only a part of an animation as illustrated by the Figure 3. Indeed, for some reasons, animators may want to transfer only eyes or mouth motions *etc.*Thus, he can disable feature points not owned by these areas and then the retargeting works only with active parts of the face. Notice that this feature may be difficult to realize with a global approach.

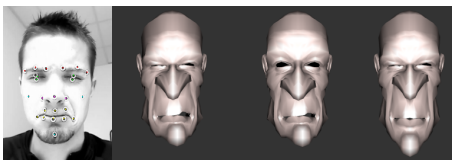

Figure 3: *Example of partial transfer. The left image shows the source expression. The second presents a full retargeting. The two last faces show partial transfer: chin and mouth points, eyes and eyebrows points.*

5 Results And Conclusion

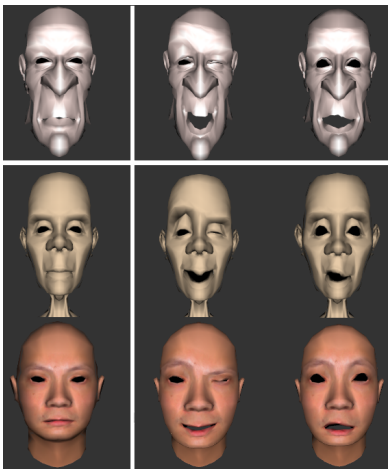

Figure 4: *The top row shows the source face's expressions. The left column shows source and target faces with a neutral expression. Expressions of the source face are retarget to the two other faces.*

Figure 1 and 4 show transfer between virtual heads: 21 feature points are used, 3 on each eyebrow, 2 on each eye, 3 on the nose and 8 around the mouth. The transfer and target animation is applied at screen frame rate on a current laptop (2,2GHz dual core, 2GB of ram and a nVidia GeForce 8600M GT). Indeed, the time of the RBF-training preprocessing is negligible since the main computation time is due to the inversion of the $N \times N$ matrix where N is the number of feature points, usually about 25. The mesh dimension has almost no impact on the speed of transfer, since vertices displacements are computed by skinning on GPU.

To test our method in an interactive situation, we developed a tracking application based on the Pyramidal Lucas Kanade Feature Tracker [Bouguet 2002]. We use 24 white markers on the actor's face. To make the tracking more robust by increasing the contrast of markers, we draw black circles on them. We use a common commercial webcam with a resolution of 640x480 pixels. Our system tracks the 2D facial movements of the markers and sends, at each frame, their positions to the retargeting viewer. Our system runs at 30 frames per second which is the camera frame rate, on a single PC or on two PC sending tracked position through the network. Figure 1 shows the result of a retargeting between motion capture data and a virtual face.

Though our technique requires few manual intervention, we plan to work on a full-automatic detection of feature points on 3D virtual faces and to improve the quality of our automatic rigging [Costa-Teixeira et al. 2006]. We also would like to remove the need of marker in the video tracker in order to provide a complete helpful system for animators in their animations prototyping tasks or for novice users in playful applications. An other improvement of our approach would be to capture and transfer fine details such as wrinkles and small deformations of skin.

Acknowledgements

This work has been supported by the PlayAll[1] consortium, a Cap Digital project.

[1] http://www.playall.fr

References

BARAN, I., AND POPOVIĆ, J. 2007. Automatic rigging and animation of 3d characters. In *ACM Trans. Graph.*, vol. 26, 72.

BOUGUET, J. Y., 2002. Pyramidal implementation of the lucas kanade feature tracker: Description of the algorithm.

CHAI, J.-X., XIAO, J., AND HODGINS, J. 2003. Vision-based control of 3d facial animation. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 193–206.

COSTA-TEIXEIRA, V., ZACUR, E., AND SUSIN, A. 2006. Transferring a labeled generic rig to animate face models. In *AMDO '06: 4th Conference on Articulated Motion and Deformable Objects*, 223–233.

DENG, Z., AND NOH, J. Y. 2007. Computer facial animation: A survey. *Data-Driven 3D Facial Animations*.

DENG, Z., CHIANG, P.-Y., FOX, P., AND NEUMANN, U. 2006. Animating blendshape faces by cross-mapping motion capture data. In *I3D '06: Proceedings of the 2006 symposium on Interactive 3D graphics and games*, 43–48.

GLEICHER, M. 1998. Retargetting motion to new characters. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, 33–42.

KÄHLER, K., HABER, J., AND SEIDEL, H.-P. 2001. Geometry-based muscle modeling for facial animation. In *Proceedings Graphics Interface 2001*, 37–46.

LEWIS, J., CORDNER, M., AND FONG, N. 2000. Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 165–172.

MAGNENAT-THALMANN, N., LAPERRIRE, R., , AND THALMANN, D. 1988. Joint-dependent local deformations for hand animation and object grasping. In *In Proceedings on Graphics interface*.

NA, K., AND JUNG, M. 2004. Hierarchical retargetting of fine facial motions. *Comput. Graph. Forum 23*, 3, 687–695.

NOH, J.-Y., AND NEUMANN, U. 2001. Expression cloning. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 277–288.

PARKE, F. I. 1972. Computer generated animation of faces. In *ACM '72: Proceedings of the ACM annual conference*, 451–457.

PIGHIN, F., AND LEWIS, J. P. 2006. Facial motion retargeting. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Courses*, 2.

POWELL, M. J. D. 1987. Radial basis functions for multivariable interpolation: a review. *Algorithms for approximation*, 143–167.

PYUN, H., KIM, Y., CHAE, W., KANG, H. W., AND SHIN, S. Y. 2003. An example-based approach for facial expression cloning. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 167–176.

VLASIC, D., BRAND, M., PFISTER, H., AND POPOVI'C, J. 2005. Face transfer with multilinear models. *ACM Trans. Graph. 24*, 3, 426–433.