Facial animation retargeting framework using radial basis functions

Tamás Umenhoffer [1], Balázs Tóth [1]

[1] Department of Control Engineering and Information Technology, Budapest University of Technology and Economics, Budapest, Hungary

Abstract

In this paper we demonstrate our framework of facial motion retargeting. We define pairs of source and target expressions and their variable configurations. For a given new source expression the target variables are automatically adjusted accordingly. We use Radial Basis Function (RBF) based regression methods to define the relationship between source and target configurations. We also investigate both realtime use with markerless capture data from web cameras, and high quality methods for production environments based on professional motion capture systems. We show that the method can handle a wide range of animation parameter types including skinning, blend shapes and material properties.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Animation

1. Introduction

Realistic facial animation[16] is a challenging task. Facial gestures and signs are one of the most significant ways of human communication. Any little mistake in the animation can make it unrealistic and unbelievable. Facial animation is crucial in the movie industry, but it also plays a great role in virtual reality applications and computer games. However realtime and offline systems may require different approaches.

In this paper we investigate both realtime and offline possibilities. Our goal is to suit the retargeting method to the needs and possibilities of the different systems. We've investigated methods that does not limit the animators in choosing their rigging tools, and also considered the possibilities of common realtime graphics engines.

We suggest two different techniques for realtime and offline environment, both based on Radial Basis Functions, but they are used differently. Once as a geometry mapping for realtime, and once as an expression mapping scattered data interpolation method for production environments. We also show how the geometry mapping method can be built into the production pipeline to enable the training of the system without the knowledge of the facial geometry of the actor who will later control the virtual character.

2. Previous Work

Early facial retargeting systems were based strictly on blend shapes, because of their intuitive meaning and easy usage. However the ideal case where both source and destination model blend shapes are provided, and source and destination models have corresponding blend shapes rarely occurs. The blend shape mapping technique of Choe[2] still assumed that the source and target has corresponding blend shapes, but source data can be given by marker positions. Actual blend shape weights can be calculated by solving a system of linear equations using least squares, but forcing that the weights should be non-negative. Chuang[3] used similar approaches for 2D video input, and also gave a method to define best key poses from input data automatically.

Buck's[1] method calculated target blend shape weights directly from marker data with scattered data interpolation. They used the two most significant PCA vectors of the input and did an interpolation based on Delaunay triangulation partitioning. They used this approach to animate 2D hand drawn faces.

Noh's[10] expression cloning technique solves the retargeting problem with geometric mapping. Source geometry is morphed onto the target geometry with radial basis function (RBF) transform refined with cylindrical projection. Animation retargeting is achieved with transforming source motion vectors, which were modified according to surface tangents and neighborhood area size. Contrary this local mapping Sumner[15] used a global geometric mapping, where poses are retargeted from arbitrary source triangle meshes to arbitrary target triangle meshes.

Radial basis functions were successfully used in facial retargeting by Pyun[12] and Deng[4]. Both system used PCA compressed source feature point data (feature vertices or motion capture markers), and mapped these data with radial basis function interpolation to target blend shape weights. Song et al.[14] showed that kernel canonical correlation ana-lysis (kCCA) can also be used for this regression, and combined RBF and kCCA results to overcome their weaknesses. Their model also took timing and emotional sequences into account using 1D Laplacian motion warping, thus gave individual characteristic to target faces. A simple but very effective solution for realtime systems was introduced by Dutreve et al.[5], where they mapped the source motion capture data to corresponding target bones of a skinned face with radial basis functions. They also introduced an automatic bone weight calculation method for new face models. A recent work of Kholgade et al.[7] deconstructed the facial performance of the actor to three layers as emotion, speech and eye-blinks to retarget motion to characters with dissimilar facial structure.

Our method is based on radial basis functions and similar to Dengs [4] and Dutreves[5] work. We focus both on realtime and production applications and provide easy integration into current graphics rendering engines and 3D packages.

3. Sources of motion and target configurations

We focused on retargeting live performance data, which is usually provided by the capture of the 3D movement of feature points (markers) defined on the surface of the performer's face. The number of markers and the quality of captured motion depends on the motion capture system. In production environment passive optical markers stacked onto the performers face are tracked by multiple high resolution and high speed cameras. These systems have high quality and are noise free. Realtime raw data is usually further processed and track failures are corrected by hand. However these systems are very expensive. For realtime applications made for average users we should find other solutions. Thanks to the great evolution of video processing and machine learning algorithms realtime markerless 3D tracking of the most important feature points of human faces is possible even with a standard web camera.

The movement of the virtual character's face is driven by deformers which modify geometry. Current 3D modeling and animation softwares provide plenty of deformers we can choose from and create intuitive tools that control the parameters of these deformers. The process of setting up the deformers and the additional tools that drives them is called rigging, and the final toolset is called rig. In case of realtime applications only those deformer types are applicable which can be accelerated by the graphics hardware, and these are skinning[9] and blend shape[11] deformations.

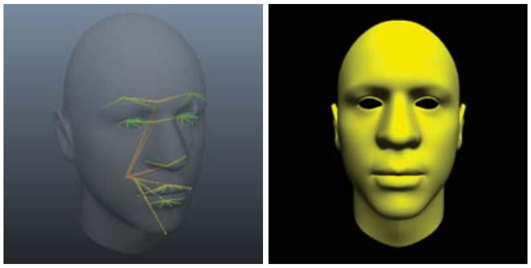

Figure 1: *Bone setup of a face model.*

Figure 1 shows a target virtual character face and the skeleton system that controls its facial movement. We use this deformation setup for realtime retargeting, as these bones are easy to set up, can create a wide range of facial movement and has a great support by graphics hardware. For production use we can use blend shapes, or a system combining blend shapes and bones. The model on Figure 5 uses blendshapes only, while the face model on Figure 3 is controlled by a rig which has bones for yaw and eyeball movement and other features are controlled by blend shapes. Blend shapes provide an intuitive control and can reproduce features like wrinkles that are hard to achieve with bones. Blend shapes also has the advantage that they morph the geometry with a linear combination of valid expressions so the result will also be valid if blend weights are positive and sum up to one. On the other hand it is quite easy to provide an incorrect bone configuration.

4. Retargeting as a scattered data interpolation problem

Retargeting is the process where we adjust the target rig parameters according to the captured performance data. To do this we have to find the relationship between these parameters. As the function of this relationship is not trivial we use an interpolation or approximation scheme based on sample source and target configurations, where the samples are not placed evenly but at intuitive configurations. We have chosen Radial Basis Function sets as they were successfully used previously in the area of facial animation retargeting.

4.1. Radial Basis Functions (RBF)

Radial basis functions are widely used for scattered data interpolation problems like surface approximation and fluid simulation[8]. Any continuous function can be interpolated with arbitrary accuracy with a sum of radial basis functions in the following form:

$$y(x) = \sum_{i=1}^{N} \omega_i \phi(\|x - x_i\|), \tag{1}$$

where ϕ is a radial function, whose value depends only on the distance from an origin (x_i in our case). There are several commonly used radial functions, from which we choose the multiquadric function:

$$\phi(r) = \sqrt{1 + (\varepsilon r)^2}$$

Each component of the target parameters define a radial basis function set, for which the corresponding ω_i weights should be calculated.

In our case the number of basis functions N is the number of predefined samples, x_i is the source configuration of sample i. Target configurations $y_i = y(x_i)$ will define the control points of the interpolation function set. Weights w_i can be computed with solving a system of linear equations:

$$T = H \cdot W, \tag{2}$$

where $H_{ij} = \phi(\|x_j - x_i\|)$, $W_i = w_i$ and $T_i = y_i$.

5. Geometry mapping

In case of realtime applications we need a high speed retargeting method, which can be immediately used for any person, who sits in front of the machine. For these reasons we used the RBF interpolation scheme for geometry mapping. This means that we find a nonrigid transformation, a functional relationship between two 3D surfaces. In practice we define sample points on the source surface and also define the corresponding points on the target surface. Thus the number of radial functions in Equation 4.1 is the number of sample points. Each function is real valued and defined on the 3D space. We have to solve three linear equation systems (see Equation 4.1), for the three component of the target 3D space. This is the learning part of the algorithm, that should be done only once. Once the radial basis function weights are computed retargeting is done on a frame by frame basis by solving equation 4.1 for the actual source values.

The source geometry is defined by the face of the user, and the sample points are tracked feature points. We used a realtime tracker library that tracks 3D positions of these feature points from realtime video, typically from a web camera. The target geometry is the virtual model. However the virtual model has much higher resolution geometry than the source geometry, thus we need to move the vertices which lie between feature points. To efficiently achieve this we set up a bone system for the virtual face model, where each source feature point has its corresponding bone. Bone weights should be painted accurately by animators.

Figure 2 (a) shows the target model and the bone positions marked by green dots. Figure 2 (b) shows the tracked feature points of the actual user. It is clearly visible that the virtual model and the user's face has different facial geometric features. If we used the tracked positions as are, without retargeting we would get an incorrect deformation shown on figure2 (d) With the geometry mapping method described above the target model keeps its own features but also inherits the source face's motion (figure 2 (c)).

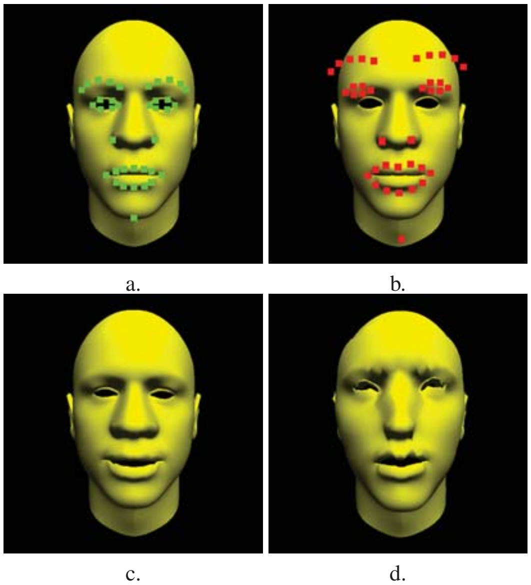

Figure 2: *Geometry mapping. Green dots show the position of the bones that deform the face geometry (a). Red dots shows the corresponding marker positions tracked for the actual user (b), the geometric features of the two models have significant differences. Using geometry mapping the user's smile appears on the target mesh believably (c). Without geometry mapping target geometry would suffer from incorrect deformations (d).*

We should mention here, that this RBF mapping can not cope with rigid transformations, so tracked feature points should be given in object space. Source and target spaces should be aligned according to rotation, translation and scaling. The realtime tracking library was able to retrieve object space positions, so our only task was to make the two object spaces consistent, thus define the scaling and translation differences. To do this we measured the distance between the eyes and the distance between the eyebrows and the chin to define the scaling, and taking the mean positions defined the translation.

Geometry mapping is fast and does not require complex learning, only the tracked feature positions of the actual user in reference pose (standard face expression). The bone system and skinning can be implemented on the graphics hardware very efficiently. However skin weight painting is a lengthy work and requires high skills and experience. Spe-

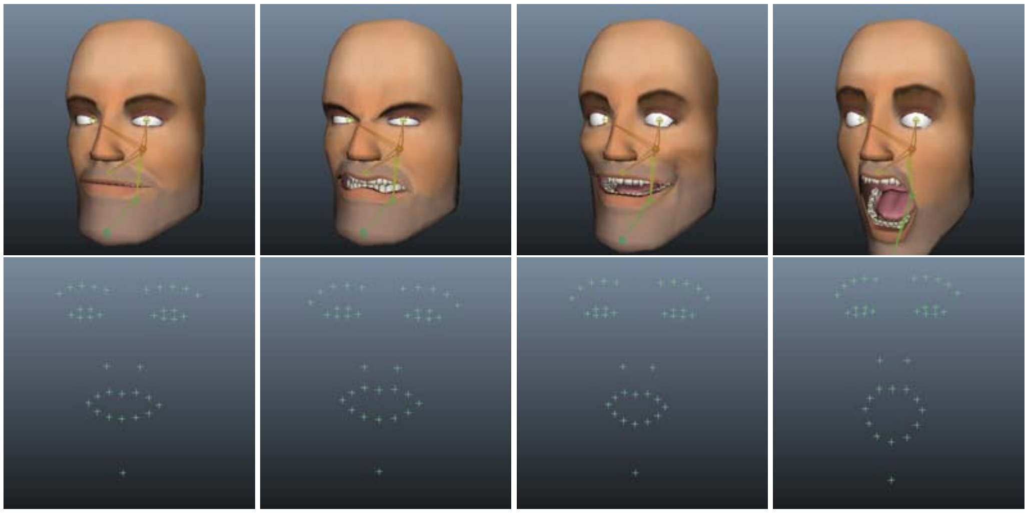

Figure 3: *The basic expressions of a face model and their corresponding source marker positions.*

cial attention should be made to tracking noise and failures, as incorrect bone configuration will result in unnatural faces. We should limit bone movement and enable only valid bone positions. This is especially important in case of realtime tracking as these systems has greater noise and are less accurate.

6. Expression mapping

Geometry mapping does not offer the accuracy and flexibility needed by production environments, therefore animation films usually use blend shapes for facial animation. Blend shapes are very flexible, as modelers can sculpt the geometry to exactly match the desired look of a given expression. Though if we sculpt complex expressions, it can be hard to reproduce an arbitrary new expression, but if we choose a good basis of blend shapes any facial motion a human can make can be mixed out. One good basis is the FACS standard[6]. The FACS system deconstruct facial expressions into 47 specific main action units (AU), which represent the activity of specific facial muscles or group of muscles which changes facial appearance.

If our target model has the necessary blend shapes the retargeting problem can be formulated as to calculate blend shape weights according to actual input, which is a motion captured 3D marker position data. Here we can make great use of the radial basis function sets again to interpolate between previously defined target states. Now the basis functions are defined on the space of marker data which has a dimension of the number of tracked markers multiplied by three, as we are tracking 3D data. As the target configuration has a dimension equal to the number of predefined blend shapes, we have this number of function sets to deal with.

The algorithm is the following. We record marker configurations that correspond to the predefined blend shapes. These corresponding data will be our training set. We should note here that the retargeting will work only for person who trained the system, ie. whose marker positions were recorded for each blend shape. We will show how to overcome this in subsection 6.1. With the training data in our hands we solve the linear equation systems of the RBF sets. When the system is trained for any new input marker configuration we can interpolate between the predefined blend shapes using the RBF equation (4.1).

Figure 3 shows three expressions and their corresponding input data. We should note here that the opening of the mouth is not a linear but a rotational movement, which can be handled better with bone animation instead of blend shapes. We have extended the target configuration space of blend shape weights with the rotation angle of the yaw bone to achieve realistic animation. It is important to note that basically any kind of data can be controlled by the retargeting process which makes the technique quite flexible. For example besides blend shape weights and rotations we can control material properties such as color or bump map intensities too. The latter can be very effective for showing small wrinkles.

6.1. Training for an average user

The expression mapping technique provides high quality animation with valid configurations. However its training and the retargeting process is more complicated than geometry mapping. It also has the disadvantage, that the system should be trained by the person who will control the final animation. As the number of blend shapes and other parameters grow this can become an annoying limitation, as we might want our model to be controlled by anyone without retraining the whole system. We can overcome this limitation if we define the geometry (marker positions) of an average user. Using the geometry mapping technique described in section 5 we can deform the actual user to the average model in realtime. Thus if the expression mapping is trained with the average user model anyone can control the animation later, only a geometry mapping should be inserted after the motion capture stage.

7. Results

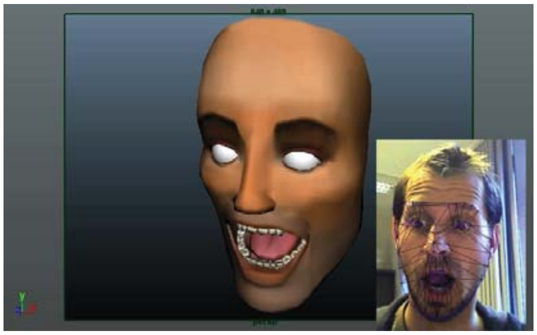

Figure 4: *Realtime expression retargeting in Maya. The animation of the face model was driven by a motion capture device using realtime tracking of a web camera video.*

We have implemented the RBF retargeting methods both in realtime and in production environments. Geometric mapping was implemented in a C++ demonstration application using Ogre3D. Ogre3D is a well known open source graphics engine designed for games and other interactive 3D applications. We used a C++ face tracking API based on constrained local models (CLM)[13] developed at Carnegie Mellon University. Figure 2 shows screenshots from the application. The geometry mapping worked quite well for naturally human like character models. However when the characters get more and more stylish or abstract the results got unnatural and suffered from visual errors. The realtime tracker also showed that a great effort should be made to reduce its noise and to prevent incorrect marker positions. The performance of the application was bounded by the tracker - which was also limited by the web camera's frame rate- the application ran at 24 frames per seconds.

To investigate the production use of emotion retargeting, we integrated our method into the most widely used animation package Autodesk's Maya as Python scripts. Our first tests used realtime markers, we created a motion capture server for the C++ tracker API to access marker data in Maya. Figure 4 shows a screenshot of realtime expression retargeting integrated into Maya, using the blendshapes shown on figure 3. We used Maya's built in character sets and poses to handle source and target configurations in an easy to use and regular way by modelers.

Though realtime tests with few basic complex expressions worked well, the inaccuracy and noise of the tracker became a serious limitation when moving to more blendshapes with smaller facial movements like FACS. Our next step was to use a high quality motion capture data recorded with a professional motion capture system. We retargeted a short performance of a singing actor to a ghost model prepared with FACS blend shapes. Figure 5 shows some frames of the animation. Here we could not train the system with the motion capture data corresponding to the specific FACS action units, as we only had the data of the final shot, which contains complex expressions. Thus we picked out typical frames from the animation that contained a representative complex facial emotion. This way we trained the system for the given shot only, and we could also limit the number of AU-s too, we used only the ones that play great role in the expressions of the recorded shot. This made the retargeting simpler, faster and more stable.

8. Conclusions

We presented two facial animation retargeting methods, one for realtime use and one for higher quality animations. Both techniques used radial basis function sets to solve a scattered data interpolation problem. Their main difference is the complexity of mapping. Our realtime method uses geometry mapping to instantly map a 3D point from the source geometry's surface onto the target geometry's surface. On the other hand expression mapping method maps complex configurations of feature points onto complex configurations of rig parameters.

We have shown RBF regression is a very flexible solution to facial animation. Depending on the needs one can control a bone system, blend shape weights or any other parameter sets, and can train the system for generic use or for a particular shot or performer too.

Acknowledgements

This work was supported by the project TÁMOP- 4.2.1/B-09/1/KMR-2010-0002.

References

1. Ian Buck, Adam Finkelstein, Charles Jacobs, Allison Klein, David H. Salesin, Joshua Seims, Richard Szeliski, and Kentaro Toyama. Performance-driven hand-drawn animation. In *NPAR 2000 : First International Symposium on Non Photorealistic Animation and Rendering*, pages 101–108, June 2000.

2. Byoungwon Choe, Hanook Lee, and Hyeong seok Ko. Performance-driven muscle-based facial animation. *The Journal of Visualization and Computer Animation*, 12:67–79, 2001.

3. Erika Chuang and Chris Bregler. Performance driven facial animation using blendshape interpolation. Technical report, Stanford University, 2002.

4. Zhigang Deng, Pei-Ying Chiang, Pamela Fox, and Ulrich Neumann. Animating blendshape faces by cross-mapping motion capture data. In *SI3D'06*, pages 43–48, 2006.

5. Ludovic Dutreve, Alexandre Meyer, and Saida Bouakaz. Feature points based facial animation retargeting. *Proceedings of the 2008 ACM symposium on Virtual reality software and technology VRST 08*, pages 197–200, 2008.

6. P. Ekman and W. Friesen. *Facial Action Coding System: A Technique for the Measurement of Facial Movement.* Consulting Psychologists Press, Palo Alto, 1978.

7. Natasha Kholgade, Iain Matthews, and Yaser Sheikh. Content retargeting using parameter-parallel facial layers. In *Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, August 2011.

8. L. Szécsi L. Szirmay-Kalos. General purpose computing on graphics processing units. In A. Iványi, editor, *Algorithms of Informatics*, pages 1451–1495. MondArt Kiadó, Budapest, 2010. http://sirkan.iit.bme.hu/šzirmay/gpgpu.pdf.

9. Nadia Magnenat-thalmann, Richard Laperrire, Daniel Thalmann, and Université De Montréal. Joint-dependent local deformations for hand animation and object grasping. In *In Proceedings on Graphics interface Š88*, pages 26–33, 1988.

10. Jun-yong Noh and Ulrich Neumann. Expression cloning. In *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, New York, NY, USA, 2006. ACM.

11. Frederick I. Parke. Computer generated animation of faces. In *Proceedings of the ACM annual conference - Volume 1*, ACM '72, pages 451–457, New York, NY, USA, 1972. ACM.

12. Hyewon Pyun, Yejin Kim, Wonseok Chae, Hyung Woo Kang, and Sung Yong Shin. An example-based approach for facial expression cloning. In *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, New York, NY, USA, 2006. ACM.

13. Jason M Saragih, Simon Lucey, and Jeffrey Cohn. Face alignment through subspace constrained mean-shifts. In *International Conference of Computer Vision (ICCV)*, September 2009.

14. Jaewon Song, Byungkuk Choi, Yeongho Seol, and Junyong Noh. Characteristic facial retargeting. *Journal of Visualization and Computer Animation*, 22(2-3):187–194, 2011.

15. Robert W. Sumner and Jovan Popović. Deformation transfer for triangle meshes. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pages 399–405, New York, NY, USA, 2004. ACM.

16. László Szirmay-Kalos, György. Antal, and Ferenc Csonka. *Háromdimenziós grafika, animáció és játékfejlesztés*. ComputerBooks, Budapest, 2003.

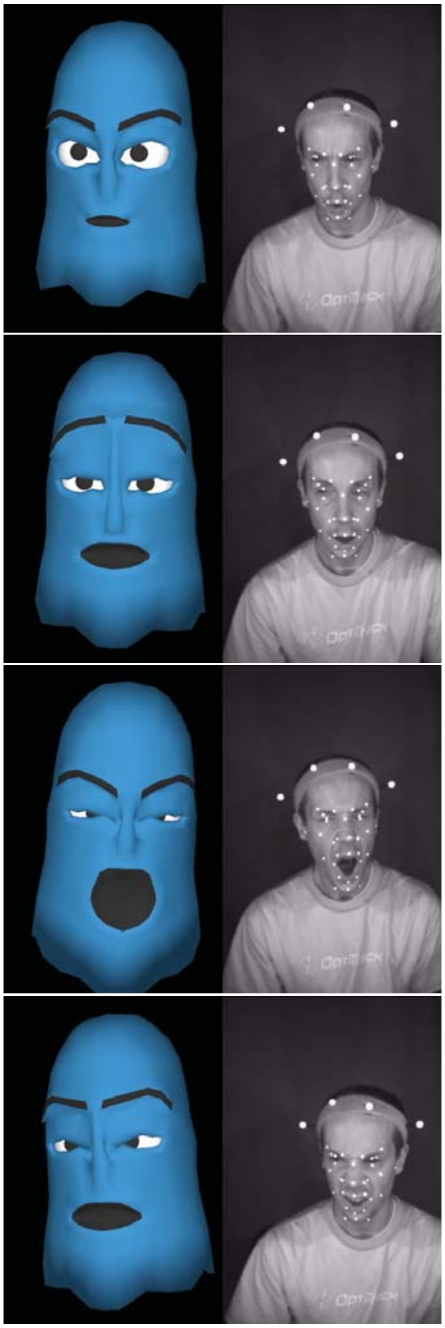

Figure 5: *Expression mapping animation. The model is controlled by 8 blend shapes corresponding to FACS action units. The system was trained with 6 representative frames of the animation. The shot was filmed at 100 FPS, the animation had 3000 frames. The ghost model is the property of Neural Information Processing Group of Eötvös Loránd University*