Eurographics/SIGGRAPH Symposium on Computer Animation (2003)
D. Breen, M. Lin (Editors)

Learning Controls for Blend Shape Based Realistic Facial Animation

Pushkar Joshi[1†], Wen C. Tien[1], Mathieu Desbrun[1] and Frédéric Pighin[2]

1 University of Southern Califronia, Computer Science Department
2 Institute for Creative Technologies, University of Southern California

Abstract

Blend shape animation is the method of choice for keyframe facial animation: a set of blend shapes (key facial expressions) are used to define a linear space of facial expressions. However, in order to capture a significant range of complexity of human expressions, blend shapes need to be segmented into smaller regions where key idiosyncracies of the face being animated are present. Performing this segmentation by hand requires skill and a lot of time. In this paper, we propose an automatic, physically-motivated segmentation that learns the controls and parameters directly from the set of blend shapes. We show the usefulness and efficiency of this technique for both, motion-capture animation and keyframing. We also provide a rendering algorithm to enhance the visual realism of a blend shape model.

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

1. Introduction

The human face has always held a particular interest for the computer graphics community: its complexity is a constant challenge to our increasing ability to model, render, and animate lifelike synthetic objects. Facial animation requires a *deformable* model of the face to express the wide range of facial configurations related to speech or emotions. There are two traditional ways of creating deformable face models: using a physically-based model or a blend shape model. A physically-based model generally simulates various skin layers, muscles, fatty tissues, bones, and all the necessary components to approximate the real facial mechanics. A blend shape model, however, mostly disregards the mechanics; instead, it directly considers every facial expression as a linear combination of a few select facial expressions, the blend shapes. By varying the weights of the linear combination, a full range of facial expressions can be expressed with very little computation.

Nowadays, there are several options for creating blend shapes. A skilled digital artist can deform a base mesh into the different canonical shapes needed to cover the desired range of expressions. Alternatively, the blend shapes can be directly scanned by a range scanner from a real actor or a clay model. With this last technique, the scanned data needs to be registered in order to produce blend shapes that share a common topology [9] and can therefore be combined correctly.

To express a significant range of highly detailed expressions, digital animators often have to create large libraries of blend shapes. In this case a naive parameterization of the face model, one that would give a parameter for each blend shape, is not practical. In particular, the visual impact of changing the contribution of a blend shape might be diffcult to predict, leading to a tedious trial and error process for the user. Splitting the face geometry in several regions that can be specified individually somewhat alleviates this problem. By manipulating a smaller area the user is guaranteed that the modification will impact only a specific part of the face (e.g., the left eyebrow). However, segmenting the face manually is difficult and tedious. The segmentation should reflect the idiosyncracies of the face being modeled and provide editing and different level of details. In general, finding the right parameters and control knobs in a blend shape model is no simple task, and it often leverages an understanding of the mechanical structure of the face. In this paper we address

† e-mail: ppj@usc.edu

the problem of parameterization and control of blendshape models.

1.1. Related Work

Blend shape interpolation can be traced back to Parke's pioneering work in facial animation [12, 13]. This initial work has found many applications both, in computer graphics and in computer vision.

Parke's original idea was rapidly extended to a segmented face where the regions are blended individually [8], allowing a wider range of expressions. Traditionally these regions are defined *manually*. A prototypical example is the segmentation of a face into an upper region and a lower region: the upper region is used for expressing emotions, while the lower region expresses speech. Although this approximation is often used in practice, such an ad hoc separation does not reflect the subtle interdependencies appearing in reality.

Blend shape models have also found their way in the computer vision community where they help analyze face images and video. Blanz and Vetter [1] designed an algorithm that fits a blend shape model onto a single image. Their result is an estimate of the geometry and texture of the person's face. Pighin et al [15] extended this work by fitting their model to a whole sequence of images, allowing manipulation of the video by editing the fitted model throughout the video sequence.

There has been little research on interactive blend shape model manipulation with the exception of the work by Pighin et al [14]. They describe a keyframe animation system that uses a palette of facial expressions along with a painting interface to assign blending weights. The system gives the animator the freedom to assign the blending weights at the granularity of a vertex. This freedom is, in practice, a drawback: by not taking into account the physical limitations of the face, it is rather difficult to create realistic expressions. The system we propose is quite different: it does respect the mechanics of the face through an analysis of the physical properties of the data. In comparison, our system is more intuitive and helps generate plausible facial expressions. Choe et al [3] have done some interesting work on mapping motion onto a set of blend shapes, but where they build a segmentation of the face manually, we learn it from the data.

Segmentation is a very active topic in image processing and computer vision [7]. However, the problem we are addressing is very different from image or optical flow segmentation; our goal is to segment a 3D mesh that is a linear combination of sample meshes. Similarly, subdivision surfaces have become a popular representation for three-dimensional objects [16]. The goals of subdivision schemes as researched so far in the compter graphics community do not match that of this paper: the segmentation of a linear space of meshes.

1.2. Contribution and Overview

In this paper, we address the problems of *parameterization* and *control* of blend shape models. We design an *automatic technique* that extracts a set of parameters from a blend shape model. Instead of deriving our control mechanism from the biomechanics of the face, we learn it directly from the available data. This solution is thus specific to the processed blend shapes, and reflects the facial idiosyncracies present in data. We also demonstrate the usefulness of these parameters through two animation techniques: motion capture and keyframing. Finally, we propose a new rendering algorithm for blend shape models; one that addresses the problem of texture misregistration across blend shape textures.

We will describe our work by starting in section 2 with some definitions and notations. Section 3 and section 4 are then dedicated to the use of the model in motion capture animation and keyframing respectively. We also explain how some of the blur artifacts can be avoided while rendering the blend shape model in section 5. We finally conclude with a discussion of our results and ideas for future research.

2. Blend Shape Face Model

Setup We define a blend shape face model as being a convex linear combination of n basis vectors, each vector being one of the blend shapes. Each blend shape is a face model that includes geometry and texture. All the blend shape meshes for a given model share the same topology. The coordinates of a vertex $\boldsymbol{V}$ belonging to the blend shape model can then be written as follows:

$$\boldsymbol{V} = \sum_{i=1}^{n} \alpha_i \, \boldsymbol{V}_i$$

where the scalars α_i are the blending weights, $\boldsymbol{V}_i$ is the location of the vertex in the blend shape i, and n is the number of blend shapes. These weights must satisfy the convex constraint:

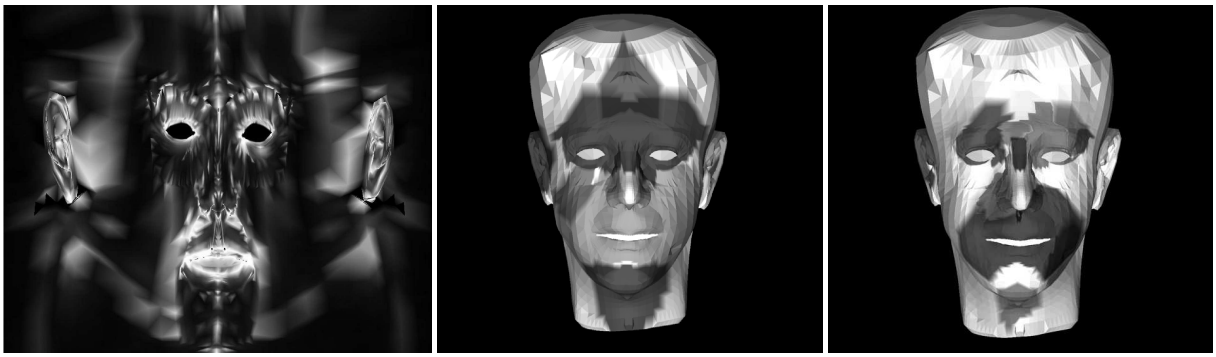
$$\alpha_i \geq 0, \text{for all } i$$

and must sum to one for rotational and translational invariance:

$$\sum_{i=1}^{n} \alpha_i = 1$$

Similarly, the texture at a particular point of the blend shape model is a linear combination (i.e., alpha blending) of the blend shape textures with the same blending weights as those used for the geometry.

Learning Controls Spanning a complete range of facial expressions might require a large number of blend shapes. For instance, the facial animations of Gollum in the feature film



(a) (b) (c)

Figure 1: *Automatically generated regions: (a) Deformation map (the deformation in X, Y and Z directions is expressed as a respective RGB triplet) (b) Segmentation for a low threshold (c) and for a high threshold.*

The Two Towers required 675 blend shapes [6]. However, studies [5] have shown that it is possible to create complex and believable facial expressions using only a few blend shapes by combining smaller, local shapes. For instance, the face geometry can be split in three areas, one covering the mouth and the jaws, another covering the eyes, and the last one the eyebrows. If the regions can be manipulated independently, the number of possible combinations (and therefore the number of possible expressions) increases significantly. Although one can define the regions manually, it requires considerable skill and time, and needs to be performed each time a new character is animated. Instead, we propose a simple, automatic and fast (less than a minute for a typical blend shape model) segmentation process that leverages face deformation information directly from the input data to create meaningful blend regions.

Physical Model One of the simplest physical models for deformable objects is that of *linear elasticity*. The deformation of an object is measured by the displacement field $\boldsymbol{d}$ between each point's current position and its rest position. As explained, for instance, in Debunne et al [4], the governing equation of motion of a linear elastic model is the Lamé formulation:

$$\rho \boldsymbol{a} = \lambda \Delta \boldsymbol{d} + (\lambda + \mu) \nabla (\nabla \cdot \boldsymbol{d}) \tag{1}$$

In our current context, $\boldsymbol{d}$ is the displacement of the vertex from its position on the neutral face, ρ is the averaged face mass density, $\boldsymbol{a}$ is the vertex' acceleration, and λ and μ are the Lamé' coefficients that determine the material's behavior (related to Young's modulus and Poisson ratio). The interpretation of the previous equation is relatively simple: the laplacian vector $\Delta \boldsymbol{d}$ of the displacement field represents the propagation of deformation through the blend shape, while the second term represents the area-restoring force. These two second-order operators, null for any rigid deformation, are therefore *two complementary measures of deformation* of our face model. To further simplify our model, we will assume that the area distortion is negligible on a face (our tests confirm that this assumption does not change the results significantly); therefore, we only use the laplacian to segment the face into disjoint regions of similar amount of deformation, as explained next.

Segmentation Debunne et al [4] have introduced a simple discrete evaluation of the laplacian operator present in Eq. 1. We compute this discrete laplacian value at every vertex of every non-neutral (i.e., expressive) blend shape, and take the magnitude of the resulting vectors. This provides us with a deformation map for each expression. We gather these maps into a single deformation map M by computing for each vertex independently its maximum deformation value across all expressions. This resulting map (see Figure 3(a) - expressed as a vector map to show direction of deformation) measures the maximum amount of *local deformation* that our face model has for the blend shapes used. A fast segmentation can now be performed by simply splitting this map in the regions with low deformation, and those with high deformation. The threshold for this split can be chosen as:

$$threshold = \boldsymbol{D}[n\,t]$$

where $\boldsymbol{D}$ is the array of sorted deformation values, n is the size of this array and t is a scalar between 0 and 1.

That is, first sort all the deformation values, and then obtain the deformation at the position that is a function of the number of values. For instance, to generate the regions in Figure 3(b,c), we used $t = 0.25$ and $t = 0.75$ respectively. Depending on the threshold, disconnected regions are created all across the mesh. We automatically clean up the regions by absorbing isolated regions into larger regions and minimizing concavity of the regions. Finally, each region is extended by one vertex all around its boundary, in order to create an overlap with the neighboring regions. The result

is a large, least-deformed region (i.e. the background), and a number of overlapping regions where there is generally more significant deformation in the range of expressions. These latter regions (see Figure 3(b,c)) correspond to vertices that generally undergo similar deformation: locally, each region deforms in a quasi-rigid way. Thus, linear blending in each of these regions will reconstruct much more detail of the target face expression as demonstrated in the next two sections.

3. Animation with Motion Capture

We express the motion in the motion capture data using the blend shape model. That is, we assume that the motion (or the per-frame position) of a motion marker can be expressed as a linear combination of corresponding points in the blend shapes. Namely:

$$\boldsymbol{M}_j = \sum_{i=1}^{n} \alpha_i \, \boldsymbol{V}_{ij}$$

where $\boldsymbol{M}_j$ is a location on the face whose motion was recorded and $\boldsymbol{V}_{ij}$ is the corresponding location in blendshape i. m is the number of motion markers and n the number of blend shapes (as in Choe et. al. [3])

Given several such equations, we find the blending weights α_i. We recast this as a minimization problem, where we need to minimize the sum of the differences:

$$\sum_{j=1}^{m} [\boldsymbol{M}_j - (\sum_{i=1}^{n} \alpha_i \, \boldsymbol{V}_{ij})]^2 \qquad (2)$$

The whole system is a linear system of equations where the unknowns α_i are the weights in the blend shape combination. By using an iterative quadratic programming solver [11], we obtain the optimal values of the blending weights α_i in the least squares sense. Solving this system is equivalent to orthogonally projecting the motion onto the set of blend shapes. In general equation 2 does not have an exact solution, since the motions can be more expressive than what the set of blend shapes allows. To produce an animated mesh that follows the motion more precisely we complement the projection on the blend shape basis by translating the vertices in the mesh by the residual $(\boldsymbol{M}_j - \sum_{i=1}^{n} \alpha_i \cdot \boldsymbol{V}_{ij})$. The residual, which is only known for a small set of points, is interpolated to the rest of the facial mesh using radial basis functions [10]. The final coordinates, $\boldsymbol{V}_j$, of a vertex on the face are then constructed using:

$$\boldsymbol{V}_j = \boldsymbol{P}_j + RBF(\boldsymbol{P}_j)$$

where $\boldsymbol{P}_j$ is the projection on the set of blend shape:

$$\boldsymbol{P}_j = \sum_{i=1}^{n} \alpha_i \, \boldsymbol{V}_{ij}$$

and $RBF(\boldsymbol{P}_j)$ is the interpolated residual at vertex $\boldsymbol{P}_j$:

$$RBF(\boldsymbol{P}_j) = \sum_{i=1}^{m} \exp(-\|\boldsymbol{M}_i - \boldsymbol{P}_j\|) \, \boldsymbol{C}_i \qquad (3)$$

In equation 3 the vectors $\boldsymbol{C}_i$ are computed using the known values of the residual at $\boldsymbol{M}_i$. Since the system of equations is linear in the unknowns, using linear least-squares provides an estimate of the unknowns [14]. Note that only applying the residual would have a different effect; by first projecting on the set of blend shapes we obtain a face geometry that reflects the blend shapes, then we apply the residual which brings the geometry closer to the motion. Choe et al [3]'s approach to mapping motion onto a set of blend shapes is very similar. The main difference is how the residual is taken into account. In their approach the blend shapes are modified to adapt them to the motions. We, on the other hand, use radial basis functions to modify the geometry on a per-frame basis. Their method would probably be more effective for processing a large quantity of motions, whereas ours would perform better on a small dataset.

Instead of solving the above system for the entire model, we solve for each region created using our automatic segmentation process. Doing so gives us localized control over the face mesh and results in better satisfaction of the spatial constraints. This also allows us to express a wide range of motion using only a limited number of blend shapes (ten, in our case).

For every frame and for every region, we construct the above minimization problem and obtain blending weights. The same weights are then used to obtain, for all vertices of the region, new positions that match the motion. Thus, for every frame of motion, we can solve a minimization problem to obtain the blending weights and consequently the face mesh that follows the motion capture data.

4. Keyframe Editing

Using our blend shape model, we can interactively construct face meshes that can be used as keyframes in a keyframing-based facial animation tool.

Creating Keyframes Creating a keyframe is similar to producing a frame in a motion capture sequence in that we need to specify control points (markers), their respective mappings, and spatial constraints (i.e. the positions of the markers). In our interface, the user can interactively specify all the above by clicking and dragging with the mouse on the face model. As in the process used in the motion capture application (see Eq. 2), we construct a minimization problem using the interactively specified constraints and obtain blending weights.

Regions and Region Hierarchy We can segment the blend shape model into regions using our automatic segmentation

Figure 2: *Successive keyframe editing from coarse (left) to fine (right) level of details.*

technique. In order to allow keyframe editing at various levels of detail, we build a hierarchy of regions. This hierarchy is created by first running the segmentation algorithm described in section 2 with a high threshold value so as to generate small and localized regions. These regions constitute the lowermost region level. We can then merge regions iteratively so that contiguous regions are merged together as we generate higher region levels.

Motion Damping Some of the locations on the face do not move significantly throughout the set of blend shapes (e.g. tip of the nose). If we were to select such a location and try to deform it, using the interface describe so far, a small motion of the mouse would trigger a dramatic change in the facial expression. To reduce the sensitivity of the system we scale the displacement of the mouse according to a factor that is inversely proportional to the maximum displacement in the blend shape model at the selected point on the mesh.

Fig. 2 displays a sequence of manipulations performed on a keyframe. The successive keyframe editing is performed with increasing level of details to refine the facial expression in a localized manner.

5. Rendering Realistic Blend Shapes

Basic Process Rendering the blend shape model is pretty straightforward and can be done in two steps: first the consensus geometry is evaluated, and then it is rendered as many times as there are blend shapes in the model to blend the texture maps. This latter step is done by assigning to each vertex' alpha channel the corresponding weight for a given blend shape. To improve our renderings we decided not to blend the texture maps on the parts of the face whose texture should not vary as a function of the facial expression, in particular, in the hair, neck, and ears area. These areas are textured using the texture map of any blend shape (usually one corresponding to the neutral expression).

Realistic Textures Texture misregistration is a common problem with blend shape rendering for realistic facial animation. If the textures do not correspond at each point on the face geometry, combining them linearly will result in a blurred rendering. Thus, the frequency content of the rendered face images varies as a function of time. Figure 4 provides an illustration of this phenomenon. The leftmost image shows our model rendered with only one contributing blend shape. The middle image shows the rendered model with seven equally contributing blend shapes. In the middle rendering a lot of the details of the face texture have disappeared.

To alleviate this problem, we borrow an approach from the image processing community [2]: we base our blending on a band-pass decomposition of the textures. More specifically, we build a two level laplacian image pyramid out of each blend shape texture map. This results in the creation of two texture maps for each blend shape: the first is a low-pass version of the original texture, and the second is a signed detail texture. We then render the blend shape model as follow: we first render the lowpass texture maps and blend them together. Then we render the detail texture map of a single blend shape using the consensus geometry and add it to the previous rendering. The result is a rendering that both, better preserves the original spectral content of the blend shape textures and maintains the high frequency content constant throughout the animation. The rightmost rendering in figure 4 illustrates the improvement obtained by using this technique.

6. Results

We demonstrate the techniques described in this paper with a set of blend shapes modeled to capture the facial expression of an actor. We created ten blend shapes corresponding to extreme expressions. We used an image-based modeling technique similar to the one developed by Pighin et al [14]. Three photographs of the actor were processed to model each blend shape: front facing, 30 degree right, and 30 degree left. All the animations shown in the video were computed and rendered in real-time (30Hz) on a 1GhZ PC equipped with an NVidia GeForce 3 graphics card. We decided to animate the tongue, the lower teeth and the upper teeth in a simple proce-

dural manner; they are moved rigidly and follow the motion of separate sets of manually selected points on the mesh. The eyeballs are moved rigidly according to the rigid motion of the head.

Motion Capture As described in section 3, we can project recorded motion onto the blend shape model. The accompanying video includes a few animated sequences that demonstrate this technique. The deformations of the face are very natural and reflect the actor's personality. Fig. 5 shows some of the frames obtained. The example shown uses only 10 blend shapes. To animate speech motion usually a much larger set of shapes needs to be used. We are able to animate the lips by using radial basis functions as described in section 3.

Keyframe Editing Also included in the video is a demonstration of the interactive tool described in section 4. The tool allows us to sculpt the face in a very intuitive way. We start manipulating the face with a set of coarse regions and refine the expression by using increasingly finer segmentations.

7. Future Work

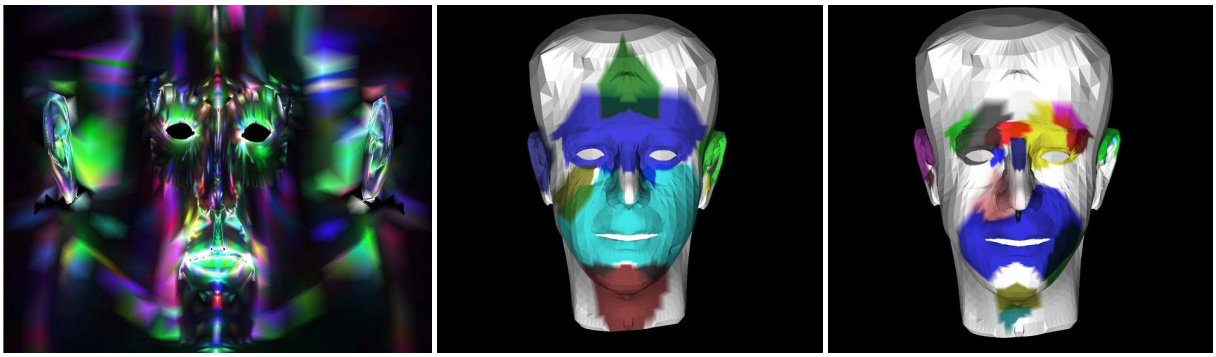
We would like to improve our results in different ways. In particular, we feel our rendering algorithm would benefit from a more principled frequency analysis of the blend shapes texture maps. Using a feature preserving filter to separate the high frequency data might lead to better results. It would be interesting to try this technique on a non-human character; one for which segmenting the face might be more challenging and non-intuitive. We would also like to test our technique on a larger dataset of blend shapes. Finally, our segmentation technique only takes into account geometric information. We would like to extend it to also take advantage of the texture information.

Acknowledgements

The authors would like to thank J.P. Lewis for discussions about blend shape animation and Andrew Gardner for its initial development. This project was supported in part by National Science Foundation (CCR-0133983, DMS- 0221666, DMS-0221669, EEC-9529152) and the U.S. Army Research Institute for the Behavioral and Social Sciences under ARO contract number DAAD 19-99-D-0046. Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the Department of the Army.

References

1. T. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *SIGGRAPH 99 Conference Proceedings*. ACM SIGGRAPH, August 1999.
2. P.J. Burt and E.H. Adelson. A multiresolution spline with application to image mosaics. *ACM Transaction on Graphics*, 2(4), October 1983.
3. B. Choe, H. Lee, and H. Ko. Performance-driven muscle-based facial animation. In *Proceedings of Computer Animation*, volume 12, pages 67–79, May 2001.
4. G. Debunne, M. Desbrun, M. Cani, and A. Barr. Adaptive simulation of soft bodies in real-time. In *Proceedings of Computer Animation 2000*, pages 15–20, May 2000.
5. P. Ekman and W.V. Friesen. *Unmasking the face. A guide to recognizing emotions fron facial clues*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1975.
6. J. Fordham. Middle earth strikes back. *Cinefex*, (92):71–142, 2003.
7. R.M. Haralick. Image segmentation survey. *Fundamentals in Computer Vision*, 1983.
8. J. Kleiser. A fast, efficient, accurate way to represent the human face. In *SIGGRAPH '89 Course Notes 22: State of the Art in Facial Animation*, 1989.
9. A. Lee, D. Dobkin, W. Sweldens, and P. Schröder. Multiresolution mesh morphing. In *Proceedings of SIGGRAPH 99*, pages 343–350, August 1999.
10. G.M. Nielson. Scattered data modeling. *IEEE Computer Graphics and Applications*, 13(1):60–70, January 1993.
11. J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, New York, 1999.
12. F.I. Parke. Computer generated animation of faces. *Proceedings ACM annual conference.*, August 1972.
13. F.I. Parke. *A parametric model for human faces*. PhD thesis, University of Utah, Salt Lake City, Utah, December 1974. UTEC-CSc-75-047.
14. F. Pighin, J. Hecker, D. Lischinski, R. Szeliski, and D.H. Salesin. Synthesizing realistic facial expressions from photographs. In *SIGGRAPH 98 Conference Proceedings*, pages 75–84. ACM SIGGRAPH, July 1998.
15. F. Pighin, R. Szeliski, and D.H. Salesin. Resynthesizing facial animation through 3d model-based tracking. In *Proceedings, International Conference on Computer Vision*, 1999.
16. D. Zorin, P. Schröder, A. DeRose, L. Kobbelt, A. Levin, and W. Sweldens. Subdivision for modeling and animation. In *SIGGRAPH 2000 Course Notes*. ACM SIGGRAPH, May 2000.



(a) (b) (c)

Figure 3: *Automatically generated regions: (a) Deformation map (the deformation in X, Y and Z directions is expressed as a respective RGB triplet) (b) Segmentation for a low threshold (c) and for a high threshold.*

(a) (b) (c)

Figure 4: *Blend shape renderings (a) a single contributing blend shape (b) seven equally contributing blend shapes without detail texture (c) seven equally contributing blend shapes with detail texture*

Figure 5: *Mapping motion capture data on a set of blend shapes*