EUROGRAPHICS 2002 / I. Navazo Alvaro and Ph. Slusallek
(Guest Editors)

Texturing Calibrated Head Model from Images

Victor Lempitsky, Denis Ivanov and Yevgeniy Kuzmin

Department of Mathematics and Mechanics, Moscow State University, Moscow, Russia
{vitya, denis, yevgeniy}@fit.com.ru

Abstract

In this paper we address a well-known problem of producing an animated model of a human head from a pair of orthogonal photographs and present a new technique of generating consistent high-quality texture for a polygonal 3D mesh which represents a personalized 3D head. The described technique is based on a combination of two different approaches to this problem and significantly extends them in order to produce optimal image in terms of minimizing visual artifacts and keeping level of details as high as possible. After a brief introduction to the complete calibration pipeline that we have used, we describe the stages of texture generation procedure, which consists of indexing available photographs with texture coordinates, finding optimal merging lines and balancing visual differences to allow for seamless merging while preserving high-frequency details. The presented technique has been evaluated within head animation software and demonstrated its ability of yielding high-quality textured model from photographs.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Color, shading, shadowing, and texture

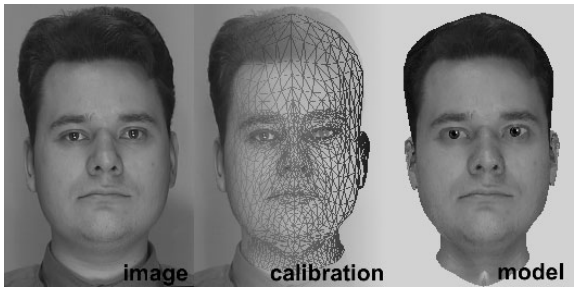

Figure 1: *Head model calibration.*

1. Introduction

Animated models of a human head are demanded in a large variety of modern applications, including among many others computer games, film production, and video conferencing. However, the problem of the effortless generation of a realistic looking, high quality model has been one of the most difficult in computer graphics, as no general, complete and efficient solution seems yet to be available.

One of the well-known approaches to this problem is calibration of a generic head model with respect to photographs, which is probably the cheapest but representative source of data. The goal of our research was to develop a complete calibration pipeline that would allow for the adjustment of a polygonal model of a generic head based on a pair of photographs taken from front and profile directions. Textured polygonal models were taken into consideration because there exist several solutions for their animation [5, 7, 8] that take advantage of modern GPU features.

In this paper we briefly summarize the techniques that we used for adjusting the mesh geometry and focus on the strategy that we developed for texture generation. This strategy exploits advantageous ideas of two existing approaches and significantly extends them in order to provide high quality texture with maximum possible level of details. Our experiments have resulted in textured polygonal models that looked very much like their real-life prototypes and allowed for high-quality animation, which have been verified in MPEG-4 animation pipeline described in [5, 9].

1.1. Previous Works

The process of texturing calibrated model of a human head comprises two major steps: (1) generating consistent image, which will be used as the head texture, and (2) defining texture mapping coordinates for a mesh that represents the head geometry. The existing approaches can be classified with respect to the order of taking these steps.

Techniques of the first group [10, 11, 12] start from generating the texture image from front and profile views. In order to combine two views in one image they introduce merging lines, which usually come from the top of a head, pass cheeks and end at the bottom of a chin. The front image is then attached with profile ones, which are deformed so that the merging lines match. The visible artifacts that inevitably appear in the boundary regions are removed by applying some blending techniques, such as pyramidal decomposition based on Gaussian operator [2]. Finally, the process of fitting texture coordinates to the polygonal model, which usually has some selected vertices identified as feature points, is performed.

Alternatively, techniques from the second group [3, 13] start from choosing texture coordinates, which are usually based on a cylindrical or spherical projection onto the head. Then initial images are indexed by these coordinates, assuming that the geometry of the model has already been matched. Finally, the complete texture is produced as a weighted sum of the indexed images, where the weights usually correspond to the deviation of the surface normal from the viewing direction.

Figure 2: *Texture area between the left eye and the hairline blended along fixed merging lines. There is no significant difference in brightness level between front (left) and profile (right) parts, but merging area is still identifiable by difference in level of details.*

The weak point of the first approach is, in our opinion, its merging lines, which are defined by a few selected feature points. Thus, profile and front elements that become neighbors across these lines after merging may actually belong to close but different areas of a head. In addition, level of details is not taken into consideration; thus, spatial resolution of the texture may significantly vary for different regions of the model. Proposed blending operator [2] compensates visual difference by averaging low frequencies while preserving high-frequency details; however, it can neither improve matching accuracy nor increase level of details. Thus, the resulting textured model usually looks well when texture minification occurs, but may still demonstrate some noticeable inconsistencies in close views, as shown, for example, in Figure 2.

The second approach is actually designed for generating texture from multiple views at arbitrary orientations. It has no restrictions on the number of views; however, its blending process usually smoothens the final image. Moreover, if texture coordinates of the same point on the skin were estimated inaccurately for different views, this point may appear several times with reduced brightness in the texture. Such effects may also be noticeable in close views.

1.2. Our Approach

In this paper we propose an algorithm for generating the complete texture image of a head from front and profile views. It utilizes the advantageous properties of both mentioned above approaches, while trying to avoid their drawbacks. We assume that texture coordinates are fixed and, thus, may be provided with the generic polygonal model that we use. We also assume that the mesh has been precisely calibrated to fit the available views. Using these assumptions we produce two separate textures by inverse mapping from front and profile images, respectively. Then, we find optimal merging lines by minimizing penalty function which describes visual difference and proper balance in level of details. This process actually extends the approach used by [10, 11] where merging lines are initially fixed or determined by a few preliminary selected feature points. We also find additional merging lines in the areas such as sides of the nose, where profile image has higher level of details compared to front one. Finally, pixel colors in the areas extracted from the profile view are balanced so that remaining visual difference along the merging lines is eliminated. Alternatively, this difference can be compensated by pyramidal decomposition as mentioned in Section 1.1. Optionally, blank areas are filled with some default values and color balanced as well. The output of the whole process is a high-quality texture image along with its mapping coordinates.

The proposed texture generation process is characterized by the following properties.

- The texture extraction process is based on the polygonal mesh accurately fitted to the available images. Thus, front and profile textures are more likely to match in the overlapping regions compared to the image-based approach.

• Merging lines are found in a way so that they minimize difference with respect to visual appearance as well as level of details after merging. This process allows for keeping level of details as high as possible in all areas, which improves the approach based on fixed merging lines. Additionally, merging lines are introduced in the areas where profile image provides higher level of details compared to the front one.

• The proposed algorithm of compensating visual difference along the merging line does not result in any smoothening in the sense that high frequency details remain unchanged. Thus, this procedure is alternative to pyramidal decomposition based on Gaussian operator [2], which is frequently used for merging images together. The resulting texture has visual quality equivalent to the initial images as no weighted blending is employed.

• The whole process of generating texture is fully automatic and does not require that any feature elements be selected on the produced texture, which is the case when the fitting of texture coordinates takes place.

1.3. Paper Organization

The paper is organized in the following way. In the next section we give an overview of the complete calibration pipeline that we have developed for producing textured model from a pair of images. Then we describe in details all stages of texture generation process, including inverse texture mapping, finding optimal merging lines, merging front and profile textures and compensating visual differences in the boundary regions. At the end of the paper we demonstrate practical results that were obtained and conclude with discussion of possible future work.

2. Pipeline Overview

The head calibration pipeline that we developed is organized as a sequence of separate stages, which are shown on Figure 3. Each stage processes the data obtained from its predecessors and performs the corresponding operations. Below we briefly describe the procedures that we use for calibrating the geometry of the head mesh, and the rest of the paper discusses the process of generating texture.

2.1. Image Registration

The goal of this stage is to register available images within the coordinate system associated with the 3D head. In order to simplify this process we currently consider a camera model with orthogonal projection. Although this assumption seems to be far from reality, the result that we obtained has demonstrated its validity for the application described in this paper. For our experiments, we have taken images using a freely available to the consumer digital camera having a focal length similar to the default one and with the flash turned

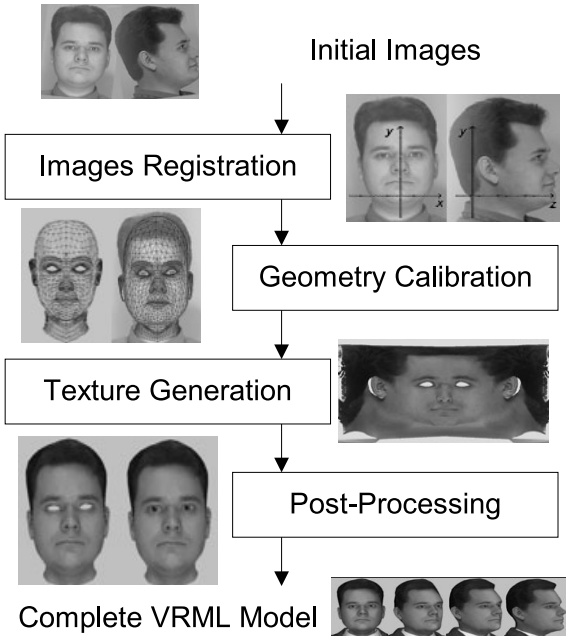

Figure 3: *Overview of the head calibration pipeline.*

on. Obviously, we could have estimated the intrinsic camera parameters and compensated distortion in the images by any camera calibration tool (see [6], for example), but we found that this has no significant influence on the quality of the resulting model.

The coordinate system that we use for the head model complies with MPEG-4 standard as shown on Figure 4, which reduces the registration process to defining coordinate axes projections onto the image planes. Thus, we first consider the front view and let the Y axis be the line of symmetry for a face. Then we restore X, which is orthogonal to Y. As the profile image also has a nontrivial projection of the Y axis, we have not only to estimate its direction but also bring its unit vector into synchronization with the one defined on the front view. This estimation is made by minimizing the least squares error between several corresponding points selected on the front and profile images. The Z axis is defined as the one orthogonal to Y in the profile projection.

After the process described above, the 3D coordinate (x, y, z) of each point of the head can be restored provided that this point is labeled on both front and profile images. In fact, x and z are uniquely defined from respective projections while y can slightly vary in front and profile. In our experiments the differences have not exceeded 5-10 pixels on average considering initial images of 1600×1200 pixels; however, for proper processing in the rest of the pipeline we keep both values for y coordinate, because they are important for texture generation. Alternatively, we could estimate

Figure 4: *Initial images after registration.*

y by weighting them, for example, in accordance to the confidence of their selection.

2.2. Geometry Calibration

When initial images are registered and, thus, a mechanism of assigning 3D coordinates to the feature elements is established, we deform the generic model so that it has the shape of the head under study. A number of techniques have recently been proposed for such deformation, including the one based on Radial Basis Functions (RBF) [1, 13], Dirichlet Free-Form Deformation (DFFD) [11], and point-based deformation [8]. Since we required precise matching of the mesh geometry and available images, we implemented a multistage procedure, which employs weighted RBF as well as topological structure of a head.

In order to adjust the global proportions of the head we follow the approach described in [1]. According to this approach the deformation problem is formulated in terms of scattered data interpolation [14], which defines a smooth function $F(P) : R^3 \to R^3$, so that

$$F(P_k) = T_k, \quad k = 1 \ldots M,$$

where T_k are the target locations of selected feature points P_k. The generic mesh is then deformed by applying $F(P)$ to its vertices, which usually comprise P_k. The function $F(P)$ is found as a composition of weighted radial functions and affine transformation by solving system of linear equations. As the proposed deformation of 3D space is very hard to predict in the areas between control points, especially if they are distributed irregularly in space, we apply this deformation in several stages each of which takes into consideration relatively regularly distributed feature points P_k.

Finally, we perform more precise adjustment of certain feature elements based on the contour lines selected on the initial images. The algorithm that we developed first adjusts those vertices of the mesh that lie on the selected contours. The other vertices are displaced in accordance to the normalized weighted sum of displacements of the nearest points on the neighboring contours, where weights are inversely proportional to the respective distances. Distance from the vertex to the contour on the mesh in considered as the shortest path along edges found by a modified Dijkstra algorithm [4].

Output of the described above multistage deformation is a polygonal mesh that will accurately match initial head images when rendered over them.

2.3. Post-processing

Some applications may demand that special conditions be satisfied by the produced model. These conditions can be fulfilled in a post-processing stage. For example, eyes should be separate objects for proper animation; thus, corresponding geometry as well as texture should be generated, based on the initial images.

3. Texture Generation

Generation of a high quality texture for a head model is, in our opinion, an even more important task compared to geometry calibration, as texture provides the major contribution to the visual appearance of the model and it can even alleviate some artifacts that might appear after mesh deformation. In this section we present in details all stages of texture generation procedure that we developed.

3.1. Inverse Texture Mapping

Texture mapping coordinates do not change during the process of texture generation; thus, they can be provided with the generic model. However, in order to produce a realistic visual appearance of the model, this mapping should cover the whole surface approximately uniformly. In addition it should be biunique, which means that no texture element is mapped to more than one location on the surface, and symmetrical for the left and right halves of the head, because we only have one profile image, and the missing profile is assumed to be symmetrical. By applying standard texture mapping of 3D Studio MAX, based on cylindrical or spherical projections, we have produced coordinates that satisfied all the conditions above.

If we have already generated texture and rendered a model in the front view, we would find texture coordinates (s,t) for each pixel, which can be indexed by (x,y) on the initial image. Thus, rasterized triangles on the image plane would be filled with data taken from the texture. Assuming that the geometry of the model matches the initial image, this procedure can be considered from the other side. Indeed, we can rasterize triangles on the texture and fill them with the interior of the corresponding triangles mapped onto the original images; thus, (x,y) coordinates will be calculated for every texture element (s,t). By doing so we produce two texture

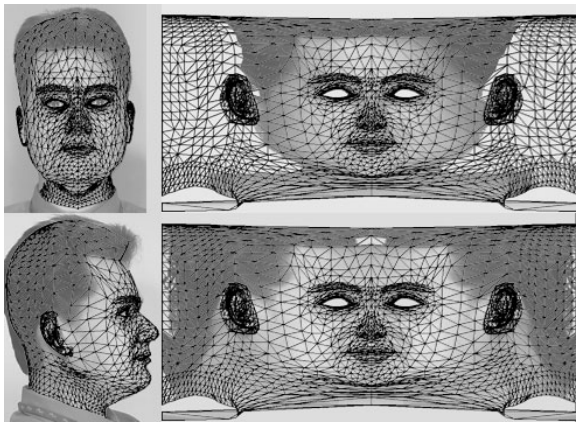

Figure 5: *Inverse texture mapping.*

planes extracted from the front and profile images respectively (see Figure 5).

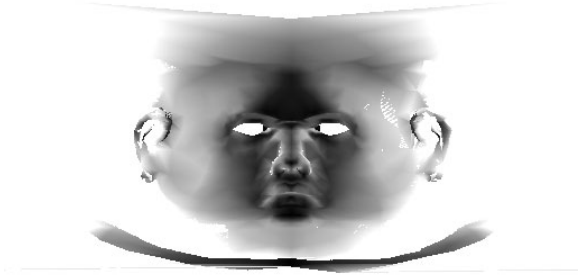

Figure 6: *Weight map for the front view (black color corresponds to maximum weight, white to zero).*

In addition, for each texture element (s,t) we calculate its weight as a dot product of the viewing direction and the normal vector estimated in the corresponding point on the surface. This weight corresponds to the level of details for each texel. It is maximal for those triangles that are observed from perpendicular direction, while it comes to zero on the oblique regions. Example of the weight map for the front view is shown on Figure 6.

3.2. Finding optimal merging line

In general, textures extracted from the front and profile images represent the respective sides of a head relatively well. The area where these images overlap goes from the top of a head through the cheeks and ends at the bottom of the chin. Thus, the merging line should lie somewhere in this area, and it is usually initially fixed on the original images or derived from the calibrated geometry. On the contrary, we propose to find it by optimizing some penalty function based on the visual appearance.

Let $c^f_{s,t}$ and $c^p_{s,t}$ be the color of texel (s,t) in the front and profile textures, respectively. Let $w^f_{s,t}$ and $w^p_{s,t}$ be their weights produced during inverse texture mapping. First, we apply a 3×3 convolution operator in order to model the filtering process that takes place in the human vision system. Thus, we calculate

$$\hat{c}^{f(p)}_{s,t} = \sum_{i=-1}^{1} \sum_{j=-1}^{1} \nu_{i,j} c^{f(p)}_{s+i,t+j},$$

where $\nu_{i,j}$ are empirical constants. Second, we estimate the visual difference between elements of front and profile texture as

$$d_{s,t} = \hat{c}^f_{s,t} - \hat{c}^f_{s,t}$$

Finally, for each texel (s,t) in the left stripe from *sl* to *sr* (see Figure 7), we calculate the penalty as follows:

$$p_{s,t} = \lambda(sr - sl)\ |d_{s,t}| +$$

$$+ \sum_{i=sl}^{s} \begin{cases} K, \ if\ w^p_{i,t} = 0 \\ 1 - w^p_{i,t}, otherwise \end{cases} + \sum_{i=s+1}^{sr} \begin{cases} K, \ if\ w^f_{i,t} = 0 \\ 1 - w^f_{i,t}, otherwise \end{cases}$$

where λ and K are empirically determined constant values describing relative contribution of the terms and penalty of the blank elements (in our experiments we used 0.05, and 10, respectively). The penalty in the right stripe is calculated in an identical manner.

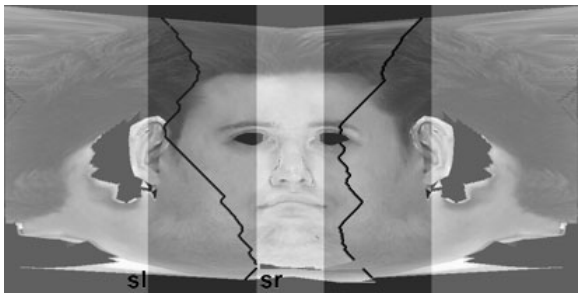

Figure 7: *Optimal merging lines.*

Thus, the penalty in (s,t) is defined as a combination of three terms: the first is just a normalized visual difference, the second term indicates how well profile texture is represented to the left of the selected texel (s,t), while the third term is responsible for the quality of the front texture to the right of the texel under study. Here we emphasize a special case $w^{f(p)}_{s,t} = 0$, which means that no texture has been extracted for this element (such texels, for example, appear in the regions that are mapped to the back side of the ears). Obviously, we want these blank elements to be as few as possible in the final texture, thus we penalize them by a much larger value K.

The merging lines are defined as the 8-connected paths of minimum aggregate penalty going down from the top to the

bottom of each stripe. They are found by utilizing a standard approach of dynamic programming. Thus, we guarantee that the specified penalty, which reflects the visual inconsistency of the final texture after merging, will be minimal if the two textures are merged along the obtained path. An example of the merging lines is shown on Figure 7.

3.3. Merging front and profile

When optimal merging lines between the textures extracted from the front and profile views are found, we can generate a complete texture by covering the central region (the one between the merging lines) by the front elements and the rest by those extracted from the profile. However, there will remain some areas where level of detail is much less compared to that of the original images. A typical example is the texture region corresponding to the side of the nose. It will be filled from the front view while being much better represented on the profile. We determine such regions and fill them with the most representative texture data.

As was mentioned in Section 3.1, the weight values $w_{s,t}^{f(p)}$ of a texture element identify its level of details in terms of the original images. We divide all texture elements into two subsets based on these weights as follows:

$$T_f\left\{(s,t)|w_{s,t}^f \geq w_{s,t}^p\right\} \text{ and } T_p\left\{(s,t)|w_{s,t}^f < w_{s,t}^p\right\}.$$

T_f contains texels which are better represented on the front image, while T_p would have more details if extracted from the profile. We determine the boundary between these subsets, and fill them with the appropriate texture data. An example of this boundary in the nose area is demonstrated in Figure 8.

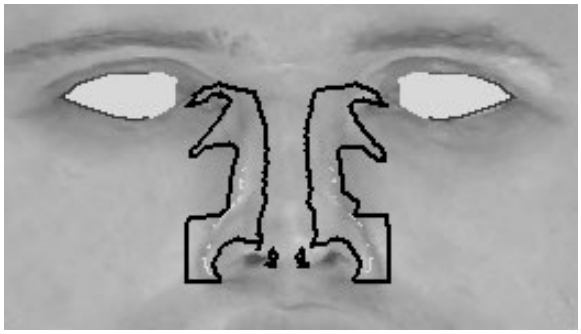

Figure 8: *Example of the boundary between texture regions extracted from the front and profile images in the nose area. The region contoured in black is better represented on the profile view.*

Obviously, we could have utilized the same strategy while finding a merging line between front and profile textures. However, the procedure described in Section 3.2 yields a much better result in practice, as it is based on not only the level of detail, but also the visual difference between respective texture regions.

3.4. Compensating visual difference

As we generate the final texture by taking some parts from the front view and the other parts from the profile, there will certainly be visual artifacts along the lines that separate these parts. Such effects are due to many factors including difference in lighting conditions (flash light is typically used), assumptions made about the camera model, inaccuracy of image registration, etc. Applying hierarchical operators [2] is the common solution for this problem; however, it might smooth down the texture around the merging area consequently decreasing its visual level of detail. As an alternative to this solution, we propose to compensate the difference along the merging lines and smoothly propagate these changes within texture regions extracted from the profile image. This approach allows the level of detail to remain unchanged.

More formally, let $T = \left\{(s_i,t_i),\ i = 1\ldots N\right\}$ be a connected set of texels extracted from the profile view, and $\partial T = \left\{(s_k,t_k),\ k = 1\ldots K\right\}$ be its boundary. For each boundary element $(s_k,t_k) \in \partial T$ we calculate visual difference d_k by the procedure described in Section 3.2. In addition, we calculate the distance from each interior element $(s_i,t_i) \in T \setminus \partial T$ to each boundary element $(s_k,t_k) \in \partial T$ as

$$r_{i,k} = (s_i - s_k)^2 + (t_i - t_k)^2.$$

Finally, denoting color values of the respective interior texels with c_i and boundary texels with c_k we apply the following transformations

$$c_i = c_i + \left(\sum_{k=1}^{K} \frac{1}{r_{i,k}}\right)^{-1} \cdot \sum_{k=1}^{K} \frac{d_k}{r_{i,k}},$$

$$c_k = c_k + d_k.$$

These transformations are applied only to those texels that are extracted from profile image; thus, the front region, which is most important for visual perception, is kept unchanged. The resulting texture has zero visual difference along the merging lines and is not degraded in terms of visual resolution. After this step it is ready for rendering.

We also implemented an algorithm based on hierarchical Gaussian operator [2]. Following the original idea we, however, extended it in order to utilize our knowledge about level of details in each particular texture region, which is expressed through texel weights (see Section 3.1 for details). Thus, we blended Gaussian layers of front and profile textures using our weights rather than those obtained by the procedure described in [2]. The resulting texture appeared to be smoother compared to the original photographs; however, no seam between front and profile regions were noticeable in general. Thus, if smoothening is not a problem, which might be the case when photographs at high resolution are available, this method can be efficiently used as well.

3.5. Correcting lighting

From our observations, a cause of certain artifacts may be the difference in lighting conditions for front and profile view (for example, flash light will be on). As a result, the appearance of a head model with the merged texture would be visually unnatural due to an unrealistic distribution of darker and lighter parts. Hence, on the pre-processing stage lighting estimation and flash lighting compensation may be applied.

We use very simple approach of estimating lighting conditions from the overlapping parts of the front and profile texture regions provided that the normals for the respective head element are already estimated.

We adopted the simplest Lambertian lighting model, considering each image being taken with one directional parallel source, which models flash light, and one ambient source. We also assume that flash intensity I_d and ambient intensity I_a are the same for both images. Following the Lambertian law, we restore the pure intensity of a point on a head as

$$c = \frac{c_f}{(n \cdot \nu_f) \cdot I_d + I_a} = \frac{c_p}{(n \cdot \nu_p) \cdot I_d + I_a},$$

where, c_f and c_p are the pixel intensities at the same point in front and profile images, and $(n \cdot \nu_f)$ and $(n \cdot \nu_p)$ are the dot products of the normal n at a respective point on a head and the viewing direction $\nu_{f(p)}$, which assumes that the flash light is positioned on a camera. From the equation above we first estimate the ratio I_d/I_a by averaging the solution for a number of points well represented on both images. Thus, we can reconstruct the pure intensity c for every pixel in both images up to a scale f actor, which is chosen to keep maximum possible range of intensity.

By applying the described above procedure we can more or less successfully compensate diffuse effects that have been caused by the flash light; however, no specular effects are taken into consideration. For some applications it may also be reasonable to keep the front image unchanged, which would produce the appearance of a head lit from the front direction. Although the proposed here algorithm is not very accurate since it uses very simple lighting model and estimates its parameters from a relatively small image region, in practice it performs satisfactorily in most cases.

3.6. Filling blank areas

It also usually happens that some parts of the produced texture remains unfilled since no corresponding data on the initial images are available. For example, this is the case for the back of the ears, which are seen on neither front nor profile view. We propose to fill such regions with some default texture or color and apply the same strategy for compensating difference on the boundary as we used for merging front and profile textures (see Section 3.4). The initial guess of color or texture does not affect the result significantly as the final colors will be determined by the neighboring texels extracted from the available images; however, certain artifacts may still appear while filling blank areas which have boundary belonging to hair and skin at the same time.

4. Discussion and Future Work

In this paper we have presented a technique of generating high-quality texture from two photographs, which is a very important part of the complete calibration pipeline that produces a ready-for-animation polygonal model of a human head. This technique is based on merging regions extracted from the available views along the merging lines, which are found by taking the visual difference and level of detail into consideration. Thus, the visual quality of the produced texture is comparable to the quality of the initial images. An example of the produced texture and calibrated model is shown in Figure 9.

Figure 9: *An example of the generated texture and the calibrated and textured model. Eyes are modeled as separate objects.*

We have verified the proposed pipeline on more than a dozen image pairs (see Figure 10 for some examples). The obtained models look very much like the originals demonstrating a high level of reality. The visual quality of the models is directly dependent on accuracy of fitting polygonal mesh within available images and resolution of the initial images. Thus we are planning to develop a technique of better matching the head geometry, which will, consequently, increase accuracy of texture generation process.

As for further research, we would like to consider two profile views for texture generation, because the hypothesis about symmetry of the profile texture seems to be incorrect in many cases. Even though the regular face tends to be symmetrical, the hairstyle often does not; thus, merging of the front texture and the missing profile, which has been reconstructed symmetrically, will often produce noticeable artifacts on the hairline. As the merging lines are constructed

Figure 10: *Photographs of the authors (top row from left to right: Denis Ivanov, Yevgeniy Kuzmin and Victor Lempitsky) and the calibrated models (bottom row). Eyes are modeled as separate objects.*

separately, no significant changes in texture merging algorithm is required to process both profile images provided that the geometry is accurately matched to them.

The other direction of further research might be extension of the proposed algorithm to the multiple views. In such case we have to determine which view can contribute maximum quality data to each particular region of the resulting texture. Merging mechanism that can seamlessly combine these data should also be developed.

In conclusion, described in this paper texture generation technique, being a part of the complete calibration pipeline, demonstrated its ability for generation of a high quality texture image that can satisfy the needs of any application requiring a realistic looking model of a human head.

5. Acknowledgements

This work was carried out by Computer Graphics Group at the Mathematics Department of MSU under a research agreement with Intel Nizhniy Novgorod Lab, which is a part of Intel Labs. We thank Valery Kuriakin (Intel, Corp.) for his constant interest in this work, and Tatiana Firsova, Victoria Zhislina and Konstantin Rodyushkin (Intel, Corp.) for their valuable comments. We also acknowledge anonymous reviewers for their valuable comments and suggestions.

References

1. L. Ambrosini, M. Costa, F. Lavagetto, and R. Pockaj. 3D Head Model Calibration Based on MPEG-4 Parameters. *Proc. of ISPACS'98*, pp. 626–630, November 1998. 4
2. P. Burt, and E. Andelson. A Multiresolution Spline With Application to Image Mosaics. *ACM Transactions on Graphics*, 2(4), pp. 217-236, 1983. 2, 3, 6
3. M. Cohen, C. Jacobs, Z. Liu, and Z. Zhang. Rapid Modeling of Animated Faces From Video. Microsoft Technical Report MSR-TR-2000-11, 2000. 2
4. E. Dijkstra. A note on two problems in connection with graphs. *Numerical Mathematics*, vol. 1, pp. 269-271, 1959. 4
5. D. Ivanov, T. Firsova, V. Kuriakin, E. Martinova, K. Rodiushkin, and V. Zhislina. Life-like MPEG-4 3D "Talking Head" (beyond standard). *To appear in proceedings of 5th International Conference on Computer Graphics and Artificial Intelligence*. May 2002. 1
6. Open Source Computer Vision Library. Intel. *http://www.intel.com/research/mrl/research/opencv/*. 3
7. T. Firsova, V. Kuriakin, E. Martinova, O. Midlina, and V. Zhislina. MPEG-4 compliant 3D Face animation. *Proc. of Graphicon'2001*, pp.54–58, September 2001. 1
8. S. Kshirsagar, S. Garchery, and N. Magnenat-Thalmann. Feature Point Based Mesh Deformation Applied to MPEG-4 Facial Animation. *Proc. of Deform'2000*, pp. 23–34, November 2000. 1, 4
9. V. Kuriakin et al. MPEG-4 Synthetic Video in real implementation. *Proc. of Graphicon'2001*, pp. 203–207, September 2001. 1
10. W. Lee, and N. Magnenat-Thalmann. Fast Head Modeling for Animation. *Image and Vision Computing Journal*, 18(4):355-364, March 2000. 2
11. W. Lee, M. Escher, G. Sannier, and N. Magnenat-Thalmann. MPEG-4 Compatible Faces from Orthogonal Photos. *Proc. of CA'99*, pp.186–194, May 1999. 2, 4
12. W. Lee, P. Kalra, and N. Magnenat-Thalmann. Model Based Face Reconstruction for Animation. *Proc. of MMM'97 (World Scientific Press)*, pp.323–338, 1997. 2
13. F. Pighin, J. Hecker, D. Lischinski, R. Szeliski, and D. Salesin. Synthesizing realistic facial expressions from photographs. *Proc. of SIGGRPAPH'98*, pp. 75–84, 1998. 2, 4
14. R. Schaback. Creating surfaces from scattered data using radial basis functions. *Mathematical methods in CAGD III, editors M.Daelhen, T.Lyche, and L. Schumaker*, pp. 1–21, 1995. 4