TOWARDS TALKING FACE ON PORTABLE DEVICES – A FLEXIBLE AND REALISTIC APPROACH

Qi Zhao Jiajun Bu Mingli Song

Microsoft Visual Perception Laboratory
Zhejiang University
Hangzhou 310027, P.R.China

ABSTRACT

Talking face using face modeling and animation techniques has been a popular branch of computer graphics for several years. There are many methods of texture mapping to obtain realistic result, or produce natural animation. In this paper, we propose a flexible texture mapping method aiming at application on portable devices. Our method needs only one single front view face image and it does not require exact match between the model and the presented texture. Satisfactory mapping can be achieved by interactive adjustment scheme, where users define correspondence for feature locations through editing the key points and their influence regions. Efficiency and realism are well balanced using our algorithm.

1. INTRODUCTION

In recent years, face modeling and animation has gained much attention from researchers [1, 2, 3]. To achieve high photorealism, many sophisticated texture mapping algorithms have been proposed [3, 4, 5, 6, 7, 8, 9, 11]. Though the model and texture image coming from the same object is better for producing realistic talking face, getting or producing a model for each user is a tough task, especially for portable devices.

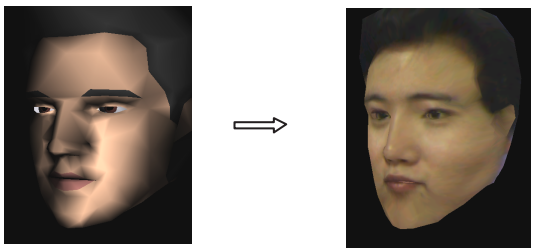

Fig. 1. Texture Mapping using a generic model and one single image (a) A generic face model. (b) Textured model.

Correspondence of feature locations is crucial in realistic modeling and animation and finding correspondence is not a piece of cake. Texture mapping puts each 3D mesh in correspondence with a planar image by assigning each vertex a pair of texture coordinates (u, v). In most current algorithms, texture is re-sampled from a video sequence [2, 3, 4] or photographs of different views, forming a cylindrical texture map [6]. Those techniques help to achieve complete and highly realistic texture mapping, yet they require sophisticated steps such as camera calibration and texture blending. An enlightening method of V.Blanz [8] shows 3D face reconstruction from single image and its application for photo-realistic image manipulations.

Though it is possible to accomplish the mapping by simply pasting the texture image onto the planar image, distortion is a big challenge in many algorithms, especially when the geometry of the object is not normative. To minimize distortion, researchers work hard to find functions with smooth interpolation and energy-minimization properties. Later, to satisfy the needs of taking user-specified information into account, the notion of constrain-based texture mapping is proposed and then becomes a popular method to solve this problem. The algorithm proposed in [5] satisfies user-defined feature correspondence for planar parameterization of meshes by adding positional constraints to the planar parameterization. A recent work of Y. Tang [7] employs Radial Bases Functions to construct smooth interpolation of non-specified texture coordinates. It improves the previous algorithm of B.Levy [9]. However, in his method, the user have to manually define the correspondence of many feature points in 3D mesh and their texture coordinates which is quite a tedious job, especially when the mesh and the texture is not well matched. To reduce interaction, F. Tang [10] finds facial features using a feature detection algorithm and correspondence is automatically achieved. The problem in this method is that it requires a large database to store the models. In addition, a large number of face images have to be trained, which is a very time-consuming process.

Therefore, although the above algorithms can get satisfactory mapping results, none can be directly applied to portable devices due to the prohibitive computational time for the interpolation function, or the significant space to store the data for automatic detection. Today, the wide use of portable devices as well as the increase needs on distant communication call for the research of talking face on PDAs and mobile phones. To accomplish this goal and at the same time consider

0-7803-8560-8/04/$20.00©2004IEEE

the computation and storage limitations, we propose here a flexible and efficient face texture mapping algorithm with generic face mesh and a single arbitrary front view face image.

The rest of the paper is organized as follows. Section 2 provides an overview of our algorism, followed by detailed explanations. Section 3 demonstrates the application of the method. Section 4 concludes the work and proposes potential future research.

2. TEXTURE MAPPING USING ONE FRONT VIEW FACE IMAGE

2.1. Overview

For application on portable devices such as PDAs and mobile phones, the critical thing is to "make the knowledge base" small enough and handle the low-bandwidth issue. The right tradeoff between accuracy and efficiency should be carefully made. In this paper, we significantly simplify the texture mapping task by allowing mapping between a generic model and an arbitrary front view face image. What is more, to reduce computational complexity, we omit steps like geometric constraints for each individual face model which can compensate the variation of face shapes and facial features distribution of different persons. Therefore, in our method, more workload is put on adjusting the correspondence of the unmatched mesh and texture. The following two subsections provide details for initialization and adjustment steps. And the mapping result is illustrated in Fig.1(b).

2.2. Initialization

The choice of only one front view face as the texture image reduces the algorithm complexity and computational time. Parameterization becomes a pure pasting process and work on camera calibration and texture blending are removed.

For a set of points on the mesh $M_i\{x_i, y_i, z_i\}_{i=1}^N \subset R^3$ and a set of corresponding 2D texture coordinates $T_i\{u_i, v_i\}_{i=1}^N \subset R^2$, the initialized mapping function $F(F_u, F_v)$is defined as:

$$\begin{cases} F_u(x_i, y_i, z_i) = f(x_i) \\ F_v(x_i, y_i, z_i) = f(y_i) \end{cases} i = 1, \cdots, N \qquad (1)$$

where f is a linear transformation from the mesh coordinates to the texture coordinates.

2.3. Adjustment

Normally, the result of initialization is not visually satisfactory due to the mismatch of the model and the texture. Important facial features may be at obviously different positions, as shown in Fig.3(a).

In our method, users are allowed to specifically define the correspondence of 3D points on the mesh and 2D points in the texture image to achieve better mapping results. Each correspondence is a pair of points (M_i, T_i), where $M_i \in R^3$ is a point on the mesh, and $T_i \in R^2$ a pixel on the texture image. In many sophisticated algorithms like [4], when key correspondence is defined, the algorithm solves global radial basis functions to get smooth correspondence of other points. Adjusting of a local feature point may have global influence. This algorithm achieves nice results, yet for portable devices, the computation is much too complex. Therefore, to compromise between efficiency and smoothness, we introduce a method that specifies the range of point influence by user definition.

In our adjustment process, the user sets the key point and its influence region. When the key points are moved for better correspondence between the mesh and the texture, other points falling into the influence region move in the same direction as the key point and at the velocity of

$$v_{point} = v_{key}(1 - d/range)^{decay} \qquad (2)$$

where $range$ is the scope of the adjustment influence; $decay$ determines the moving speed of this point relative to the key point; v_{key} is the moving velocity of the key point and d the distance between this point and the key moving point.

For instance, $P(x, y, z)$ is an important feature point in the mesh and its corresponding key point on the texture image is $P_T(u, v)$. When $P_T(u, v)$ moves at the velocity of v_P, the point $P_{T1}(u_1, v_1)$, which is in the influence region of $P_T(u, v)$, moves at the velocity of

$$v_{P_{T1}} = v_{P_T}(1 - \|P_{T1} - P_T\|/range)^{decay} \qquad (3)$$

Let

$$\mu = (1 - \|P_{T1} - P_T\|/range)^{decay}$$

Then the mapping function of P_1 is:

$$\begin{cases} F_u(x_1, y_1, z_1) = f(u_1 + \Delta u \cdot \mu) \\ F_v(x_1, y_1, z_1) = f(v_1 + \Delta v \cdot \mu) \end{cases} \qquad (4)$$

Δu and Δv are translations of $P_T(u, v)$ along the u, v axes.

Notice that Δu and Δv here are only translations caused by users manually moving this point, while the position of $P_T(u, v)$ is also influenced by other points, as long as $P_T(u, v)$ falls into the points' influence range. Hence, the total translation of $P_T(u, v)$ can be calculated as follows:

$$\begin{cases} \Delta u' = \sum_i (\Delta u_i \cdot \mu_i) \\ \Delta v' = \sum_i (\Delta v_i \cdot \mu_i) \end{cases} i = m_1, \cdots, m_j \qquad (5)$$

Here, j is the number of points that influence P_T (including P_T itself); Δu_i, Δv_i are the translations along the u, v axes of the point $P_{Ti}(u, v)$ which influences the $P_T(u, v)$. It can also be written in vector form as

$$\Delta\vec{p}' = \sum_i (\Delta\vec{p}_i \cdot \mu_i), i = m_1, \cdots, m_j \qquad (6)$$

This equation is applied for every point on the mesh, so if for the point P_i, let

$$\varphi_{ij} = (1 - d_{ij}/range_j)^{decay_j}$$

then for all points P_1, P_2,..., P_N on the mesh, we have

$$\begin{bmatrix} \varphi_{11} & \cdots & \varphi_{1N} \\ & \ddots & \\ \varphi_{N1} & \cdots & \varphi_{NN} \end{bmatrix} \begin{bmatrix} \triangle\vec{p}_1 \\ \vdots \\ \triangle\vec{p}_N \end{bmatrix} = \begin{bmatrix} \triangle\vec{p}_1{}' \\ \vdots \\ \triangle\vec{p}_N{}' \end{bmatrix} \tag{7}$$

In this equation, d_{ij} is the distance between points P_{Ti} and P_{Tj}, $\Delta\vec{p}_i$ the distance of P_{Ti} moved by user and $\Delta\vec{p}_i{}'$ the final displacement of P_{Ti}.

So when user interactively sets the texture coordinates for the important feature points on the model, it should be noticed that the final position of this point is also influenced by other points. For instance, for two points P_{k1} and P_{k2} on the mesh, suppose that user has fixed its corresponding points on the texture, say points (u'_{k1}, v'_{k1}) and (u'_{k2}, v'_{k2}), so that their final displacements are $\Delta\vec{p}_{k1}{}'$ and $\Delta\vec{p}_{k2}{}'$. Then user interaction should satisfy the following two confinements:

$$\begin{cases} \sum\limits_i (\Delta\vec{p}_i\mu_i) + \Delta\vec{p}_{k1} = \Delta\vec{p}_{k1}{}', i = m_1, \cdots, m_{j1} \\ \sum\limits_i (\Delta\vec{p}_i\mu_i) + \Delta\vec{p}_{k2} = \Delta\vec{p}_{k2}{}', i = n_1, \cdots, n_{j2} \end{cases} \tag{8}$$

$j1$ is the number of points that influence P_{Tk1} (excluding P_{Tk1} itself), which is similar to $j2$.

3. RESULTS

We tested the flexibility of our method on different texture photos and got convincing result. The images were either taken by us using digital cameras, or taken under any unknown condition, say the paintings of the middle age. Though it is better for the face images be front view and neutral, we also selected some photos of slightly side view or with subtle expression for test. A good example is the famous painting Mona Lisa(see Fig.2). After coarse adjustment, the slightly side view face became front view. In this case, we illustrate the function of $decay$ and $range$.

Fig.2(b) shows the result under a relatively small value of $decay$ (In Fig.2 and 3, red points are key points, and the influence ranges are marked red circle), in this situation, the real adjustment influence is large though the $range$ is smaller; In comparison, when $decay$ is larger, as in Fig.2(c) and (d), although the value of $range$ is set larger, its real influence is not as large as in Fig.2(b). This is due to the position of $decay$ in the equation. When $decay$ is larger, its influence to the far points becomes smaller exponentially.

Usually in our generic models, the number of points each mesh contains ranges from 100 to 200, half of which can find correspondence in the front view face image. In the adjustment process, users select some key points in the image, which are projections of the important feature points on the model. Those key points may be points on the eye corner, the tip of the nose, or other less prominent points on the eyebrow or the chin. After a key point is chosen, coarse adjustment can be done under a large value of $range$ and

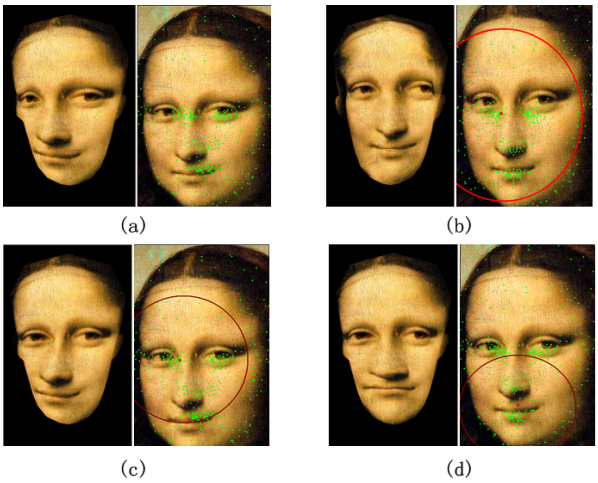

Fig. 2. Coarse adjustment to change face direction (a) Initialized mapping. (b) Mapping after one-step adjustment (range=60,decay=1). (c) Mapping after one-step adjustment (range=100,decay=4). (d) Mapping after two-step adjustment (range=100,decay=4).

elaborate mapping can be achieved when the $range$ is small. Particularly, by setting $range$ to be 0, the adjustment only influences a single feature point.

An example of coarse adjustment for the right eye is illustrated in Fig.3. We see from Fig.3(a) that the position of the right eye in the texture image is above the one in the model. In Fig.3(b), one-step adjustment is done on the right eye by choosing a key point and setting the size of its influence range as well as a proper decay value. This coarse adjustment offers convenience for later elaborate adjustment.

After manual adjustment of all the key points and their influence regions, the overall correspondence between the 3D points on the model and the 2D points of the texture is defined. Then with the help of OpenGL, satisfactory mapping results can be achieved.

Those 3D models we use are MPEG-4 compliant. They are described by FDP (Facial Description Parameters) and can be easily animated by FAP (Facial Animation Parameters). In our system, we use a file containing a sequence of FAP to get animation. And after employing this mapping method, more realistic animation is achieved. Typical emotions in the animation sequence are demonstrated in Fig.4.

4. CONCLUSIONS AND FUTURE WORK

Texture mapping for application on portable devices requires a careful tradeoff between accuracy and efficiency. We introduce a simple and flexible way to achieve satisfactory results with modest interaction. Compared with other mapping methods, our method has the following advantages to make the process properly simple: Firstly, any face model and texture image can be used in our system and it does not need to do geometric constraints for each individual face model; secondly, only one front view face image is required

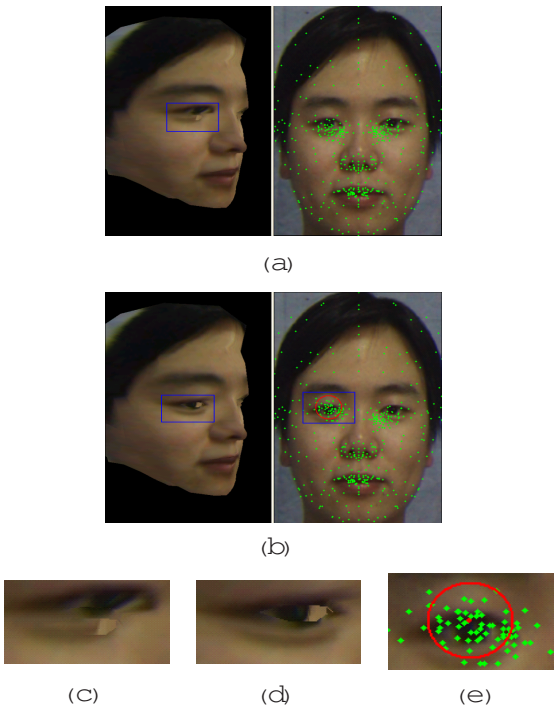

Fig. 3. Coarse adjustment of local feature position (a) Initialized mapping. (b) Mapping after one-step adjustment of the right eye. (c), (d) and (e) are local enlargement of (a) and (b).

in our work to avoid problems such as camera calibration and texture blending; finally, we define an efficient algorithm to manually set the influence region of adjusted points so that the complex global optimization function is avoided.

In our work, we pursue real-time animation at the expense of accuracy. Fortunately, people tend to pay less attention to their back view; therefore, we just need to build some generic hair models for simple use. Other disadvantages of our simplification are the distortion and un-smoothness. Without the computation of an energy function, these visual imperfections are unavoidable and we may count the improvement of this situation on the development of portable devices on both space and computational speed. In addition, when more models are developed on such devices, our mapping results would be enhanced and users can choose from those different models to find a most fitted one for mapping. Further work also focuses on the texture mapping of facial features such as eyes and teeth [11] for specific application on portable devices.

5. ACKNOWLEDGEMENT

This work is partly supported by NSFC grants 60203013 and HP laboratory of Zhejiang University. Thanks Nan Li, Mingyu You for their advices.

6. REFERENCES

[1] F. I. Parke, and K. Waters, *Computer Facial Animation*, AKPeters, Wellesley, Massachusetts, 1996.

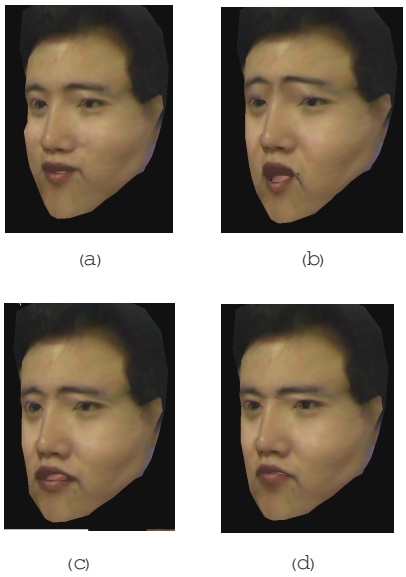

Fig. 4. Typical Emotions (a) Happiness. (b) Surprise. (c) Sadness. (d) Anger.

[2] Zicheng Liu, Zhengyou Zhang, Chuck Jacobs, and Michael Cohen, *Rapid Modeling of Animated Faces From Video*, Technical Report, MSR-TR-2000-11, Microsoft Research, 2000.

[3] Y. C. Lee, D. Terzopoulos, and K. Waters, *Realistic modeling for facial animation*, Proc. SIGGRAPH'95, pp. 55-62, 1995.

[4] P. E. Debevec, C. J. Taylor, and J. Malik, *Modeling and Rendering Architecture from Photographs: A Hybrid Geometry-and Image-Based Approach*, Proc. SIGGRAPH'96, pp. 11-20, 1996.

[5] V. Kraevoy, A. Sheffer, and C. Gotsman, *Matchmaker: Constructing Constrained Texture Maps*, ACM Transactions on Graphics (Proc. SIGGRAPH 2003), Vol. 22, Issue 3, pp. 326-333, 2003.

[6] Won-Sook Lee, Marc Escher, Gael Sannier, and Nadia Magnenat-Thalmann, *MPEG-4 Compatible Faces from Orthogonal Photoes*, Proc. CA99 (International Conference on Computer Animation), pp. 186-194,1999.

[7] Y.Tang, J.Wang, H.Bao, and Q.Peng, *RBF-based constrained texture mapping*, Computers & Graphics, vol. 27, issue 3, pp. 415-422, 2003.

[8] Volker Blanz, and Thomas Vetter, *A Morphable Model For The Synthesis Of 3D Faces*, Proc. SIGGRAPH'99, pp. 187-194, 1999

[9] B.Levy. *Constrained Texture Mapping for Polygonal Meshes*, Proc. SIGGRAPH'2001, pp. 417-424, 2001.

[10] F.Tang, *Facial Feature Detection with Hierarchical Constraints*, Master Thesis of Zhejiang University, 2004.

[11] Marco Tarini, Hitoshi Yamauchi, Jörg Haber, and Hans-Peter Seidel1, *Texturing Faces*, Proc. Graphics Interface 2002, pp. 89-98, 2002.