Realistic Face Modeling by Registration of a 3-D Mesh Model and Multi-View Color Images

Hongbin Zha, Peng Wang
Center for Information Science, Peking University
E-mail: zha@cis.pku.edu.cn, wpeng@cis.pku.edu.cn

Abstract

Realistic 3-D face modeling is a challenging research topic that involves tight integration of computer vision and computer graphics techniques. Acquisition of surface textures is an indispensable step in realistic 3-D face modeling. In this paper, we propose a camera-calibration based method to register 3-D mesh models with multi-view color images. Then a global texture map is synthesized from texture patches corresponding to each image. A main difficulty here is that the texture-mapping parameters are sensitive to initially selected correspondences between the mesh model and images. To solve this problem, we developed an iterative optimization algorithm for the parameter estimation through automatic adjustment of the selected correspondences. Experimental results are given to demonstrate the performance of our method.

Keywords: 3-D face modeling, texture mapping, model and image registration, optimization.

1. Introduction

With rapid growth of internet and multimedia communication, the demand for a convenient and effective solution to creating realistic 3-D face models has been increasing, which has many applications in several fields, such as entertainment, immersive telepresence, data compression, and medicine. At the same time, the development of high precision 3-D scanners and high performance graphics workstations provides an excellent experiment platform for the research on this topic.

In general, the realistic 3-D face modeling consists of three important aspects: 1) recovery of geometric shapes; 2) acquisition of surface textures; 3) recovery of skin reflectance properties. All of them contribute significantly to the fidelity of the final constructed face models.

Relevant work on this research topic can be classified into two typical categories based on the input: 1) images only; 2) both geometry and image data. Due to the lack of 3-D scanners, a majority of the work belong to the former category, and the focus is the recovery of the geometrical shape. The simplest case using images alone as input involves only two orthogonal views (namely, the front and side views) [9], which needs careful camera calibration to capture the two views. A representative work that uses multiple views as input is that of Pighin’s [8], where a system for realistic expression synthesis is developed with a manually intensive procedure. Recently many researchers are concentrating on modeling from monocular image sequences [5, 6] or stereo video [11], where integrated feature tracking and structure from motion techniques are used, and the parameters are estimated via bundle adjustment. A common feature of the above work is that a generic model is employed and fitted to a specific person’s face according to the feature correspondences and geometrical constraints. Unlike the above methods, a fully automatic system was presented in [7] to generate a 3-D model from one camera without using a generic model. Another interesting work is that of Vetter and Blanz’s [1]: given a single image of known orientation and illumination, they produced a reconstructed model as a linear combination of 3-D scans in a database.

A main disadvantage of the purely image-based approach is the lack of accurate geometric measurement, which results in unrealistic visual effects in the motion or deformation of the face model. To solve the problem, 3-D scanners are employed in the geometry-based approach to obtain more accurate geometric information. For example, authors in [10] and [2] use structured-light range scanners to construct the geometry models. In such circumstances, the focus is the acquisition of surface textures, or the recovery of skin reflectance properties. A number of techniques are presented in [10] to address the generation of facial skin textures from uncalibrated input photographs, as well as the creation of individual textures for facial components such as eyes or teeth. Another impressive work is that of Debevec’s [2], where a method is presented to recover the reflectance field of a human face with images acquired from a small set of viewpoints under a dense sampling of illumination directions, and then to use these measurements to synthesize novel views of the face under changes in lighting and viewpoint. However, neither of the above methods gives a detailed description of how to register the geometry model with arbitrary uncalibrated images.

With 3-D laser scanners available, triangular mesh models with high resolution and precision are used in our method. Our work focuses on texturing the mesh
model via the registration of the mesh model and multiple uncalibrated images. Using optimization for 3D-2D registration has been done in the area of medical image. For example, [4] presents a framework for 3D-2D projective registration, using tangent information and extended ICP algorithm. The registration problem in our method is also based on the estimation of 3D-2D projective transformation. However, we start from a sparse set of manually selected correspondences, and propose an iterative optimization formulation for projective parameters estimation via automatic adjustment of the positions of the correspondences. Actually such a technique is not particularly specific to face modeling, and can be applied to more general 3D object or scene modeling problems.

Recently new 3-D scanners such as Monolta VIVID 910 can also achieve very accurate registration and generate textured 3-D models with high fidelity. However, as compared with the images captured with digital camera, the images captured with VIVID 910 have a fixed and relatively lower resolution, which is not suitable for texture re-sampling during mesh subdivision. In addition, each image captured with VIVID 910 is bound with a 3-D scan; such inseparability lacks the convenience of our method, where the capture of images is independent of the scanning of the mesh model.

2. General approach

Our method, as shown in Fig.1, involves three phases: initialization, refining loop, and finalization.

![System overview](image)

2.1. Initialization

The input data consist of a mesh model and color images of the same object. The mesh model is acquired with a Polhemus FastScan scanner, and further processed (hole-filling, smoothing, etc.) with the InnovMetric PolyWorks software package. The color images are captured with a digital camera, and we assume both the intrinsic and extrinsic parameters of the camera are unknown. We first manually select a sparse set of 3-D points \( \hat{P}^i \) from among the mesh vertices, and then specify the corresponding pixels \( p^k_i \) of each selected vertex \( \hat{P}^i \) in all the images where \( \hat{P}^i \) is visible. Here \( p^k_i \) is the expected projection pixel of \( \hat{P}^i \) in the kth image ( \( k = 1...N \) ) under perspective projection. These correspondences \( (\hat{P}^i, p^k_i) \) are used in the following camera-calibration based algorithm to compute the projection matrices. An example of the initialization process is illustrated in Fig.2.

![Initialization](image)

Figure 2. Initialization. (a) 3-D face model, where the cubes denote the selected mesh vertices. (b) Face image, where the centers of the diamonds denote the corresponding pixels of the selected vertices.

2.2. Refining loop

When we estimate the projection matrices with the camera-calibration algorithm (described in detail in Section 2.2.1), the computed projection parameters are sensitive to the initially selected correspondences. Slight facial deformation during the scanning phase and photo-taking phase, or small errors in the correspondence-locating phase, will both result in the inaccuracy of the initially selected correspondences. To solve the problem, we utilize the refining loop to adjust the selected correspondences while estimating the projection matrices at the same time. Consequently, we actually have two sets of parameters: 1) projection matrices; 2) 3-D coordinates of the selected mesh vertices. We will adjust the two sets of parameters separately in the iterative optimization process (described in detail in Section 2.2.3). More specifically, we will adjust one set while fixing another, and all the parameters in the same set are estimated simultaneously. In this way, the initial non-linear optimization problem is decomposed into linear optimization problems, and can be solved with linear least-squares methods.
Generally the refining loop can be described as follows: Firstly, projection matrices are computed with the initially located correspondences. These matrices as well as the position of the manually selected vertices can be regarded as the initial estimation of the parameters to be adjusted. Then in the iteration process, the positions of the manually selected vertices are automatically adjusted with the computed projection parameters. Subsequently, the adjusted vertices are sent back to update the correspondences, which are again used to estimate the projection matrices. An objective function is defined as the guide of the loop, and the loop is terminated when the minimum of the objective function has been reached. An example for the refining loop is given in Fig.3.

2.2.1 Projection matrix computation. In the camera calibration method introduced by Faugeras [3], the mapping between a mesh vertex \( \tilde{P} = (x, y, z) \) and its image projection pixel \( p = (u, v) \) is described as

\[
(U V S)' = M \cdot (x y z 1)',
\]

Here \( U, V, \) and \( S \) can be interpreted as the projective coordinates of \( \tilde{P} \) in the retina plane. \( u = U / S, v = V / S, \) and \( M = (m_{ij}) (i = 1..3, j = 1..4) \) is the projection matrix.

Given a correspondence between a mesh vertex \( \tilde{P}_k = (x_k, y_k, z_k) \) (here \( i \) indicates the index of the selected mesh vertices, \( i = 1..L \) ) and its corresponding image pixel \( p_k = (u_k, v_k) \) in the \( k \)th image (here \( k \) indicates the index of the images, \( k = 1..N \) ), we obtain

\[
\begin{align*}
x_k m_{11}^k + y_k m_{12}^k + z_k m_{13}^k + m_{14}^k &= u_k m_{24}^k, \\
x_k m_{21}^k + y_k m_{22}^k + z_k m_{23}^k + m_{24}^k &= u_k m_{34}^k, \\
v_k m_{31}^k + v_k m_{32}^k - v_k z_k m_{33}^k &= v_k m_{44}^k.
\end{align*}
\]

Subsequently, we obtain \( 2 \cdot L_k \) equations if there are \( L_k \) correspondences between the mesh model and the \( k \)th image. Given these equations, projection matrix \( M^k \) can be estimated by a constrained optimization method [3].

2.2.2 Selected 3-D vertex adjustment. Given the projection matrix \( M^k \), together with the selected image pixel \( p_k^i \), two equations that have the same form as eqs. (1) and (2) can be obtained for a selected mesh vertex \( \tilde{P}_k \). Subsequently, \( 2 \cdot N_k \) equations can be obtained if \( \tilde{P}_k \) is visible in \( N_k \) images. Given these equations, a new position as the candidate of the adjusted position of \( \tilde{P}_k \) can be computed by a least-squares method.

2.2.3 Optimization scheme. An objective function is needed for the optimization algorithm. Firstly we define two error functions \( E_p(\tilde{P}_k) \) and \( E_m(M^k) \) as

\[
E_p(\tilde{P}_k) = \sum_{i=1}^{N_k} \text{dis}(m^k(\tilde{P}_k) - p_k^i),
\]

\[
E_m(M^k) = \sum_{i=1}^{N_k} \text{dis}(m^k(\tilde{P}_k) - p_k^i).
\]

Here

\[
m^k(\tilde{P}_k) = (u_k^i, v_k^i);
\]

\[
\begin{align*}
u_k^i &= x_k m_{11}^k + y_k m_{12}^k + z_k m_{13}^k + m_{14}^k, \\
v_k^i &= x_k m_{31}^k + y_k m_{32}^k + z_k m_{33}^k + m_{34}^k.
\end{align*}
\]

\( m^k(\tilde{P}_k) \) is the projection function, which can be derived from eqs. (1) and (2). \( \text{dis}() \) is the 2-D Euclidean distance, and thus \( \text{dis}(m^k(\tilde{P}_k) - p_k^i) \) is the error between the actual projection pixel \( m^k(\tilde{P}_k) \) and the expected projection pixel \( p_k^i \) of a selected vertex \( \tilde{P}_k \) in the \( k \)th image (we also call \( \text{dis}(m^k(\tilde{P}_k) - p_k^i) \) the re-projection error). \( E_p(\tilde{P}_k) \) can be regarded as the re-projection error sum associated with a selected vertex \( \tilde{P}_k \), and \( E_m(M^k) \) as the re-projection error sum associated with the \( k \)th image. An illustration of \( \text{dis}(m^k(\tilde{P}_k) - p_k^i) \) is shown in Fig.4.

Here neither of the two error functions is sufficient to reflect the accuracy of all the parameters to be estimated. However, on the basis of the fact that the sum operations are permutable, interestingly we find out that

\[
\sum_{k=1}^{K} E_p(\tilde{P}_k) = \sum_{k=1}^{K} \sum_{i=1}^{N_k} \text{dis}(m^k(\tilde{P}_k) - p_k^i) = \sum_{k=1}^{K} \sum_{i=1}^{N_k} \text{dis}(m^k(\tilde{P}_k) - p_k^i) = \sum_{k=1}^{N_k} E_m(M^k).
\]
According to the above equation, we obtain the definition of the objective function \( E(\{\tilde{P}_j\}^{k} \{M^k\}) \) as
\[
E(\{\tilde{P}_j\}^{k} \{M^k\}) = \sum_{i=1}^{n} E_{\alpha}(\tilde{P}_i) = \sum_{i=1}^{n} E_{\alpha}(M^k).
\]
\( E(\{\tilde{P}_j\}^{k} \{M^k\}) \) can be explained as the sum of the re-projection errors of all the \( L \) selected mesh vertices \( \tilde{P}_i \) in all the \( N \) images.

![Face mesh model](image)

### Step 1

With this objective function, the optimization algorithm can be described as follows:

1. **Step 1** For \( k = 1 \ldots N \), compute \( M^1 \), with \( L \) initially located correspondences.

2. **Step 2** In the \( n \)th iteration cycle, given the computed \( M^k \), together with the selected pixel \( p_i^k \) in the \( k \)th image \( (k = 1 \ldots N_i) \), compute a new position \( \tilde{P}_i^* \) as the candidate of \( \tilde{P}_i^{(n)} \). Then find out the maximum set of \( \{\tilde{P}_j\} \) that satisfy

\[
\sum_j E_{\alpha}(\tilde{P}_j) + \sum_{j=1,j \neq l \neq k} E_{\alpha}(\tilde{P}_j) \leq \sum_j E_{\alpha}(\tilde{P}_j) + \sum_{j=1,j \neq l \neq k} E_{\alpha}(\tilde{P}_j) = E(\{\tilde{P}_j\}^{n} \{M^n\}).
\]

If \( \{\tilde{P}_j\} \neq \phi \), let \( \tilde{P}_j^{(n)} = \tilde{P}_j^* \), but keep \( \tilde{P}_j^{(n)} = \tilde{P}_j^{(n-1)} \), and update all correspondences \( \{\tilde{P}_j^{(n)}\}, p_i^n \), then go to **Step 3**; otherwise terminate the iteration process.

3. **Step 3** In the \( n \)th iteration, compute a new matrix \( M^{1} \) as the candidate of \( M^{k} \), with the \( L \) updated correspondences \( \{M^k\} \) is the updated \( M^k \) in the \( n \)th iteration cycle, and initially \( M^{k+1} \) is \( M^{k} \). Then find out the maximum set of \( \{M^j\} \) that satisfy

\[
\sum_j E_{\alpha}(M^j) + \sum_{j=1,j \neq k} E_{\alpha}(M^k) \leq \sum_j E_{\alpha}(M^j) + \sum_{j=1,j \neq k} E_{\alpha}(M^k) = E(\{\tilde{P}_j\}^{n} \{M^n\}).
\]

If \( \{M^j\} \neq \phi \), update \( M^{k+1} = M^j \), but keep \( M^{k+1} = M^k \), then go back to **Step 2**; otherwise terminate the iteration process.

Given the optimization algorithm described above, we obtain two inequations:

\[
E(\{\tilde{P}_j^{(n)}\}^{k+1} \{M^k\}) \geq E(\{\tilde{P}_j^{(n)}\}^{k+1} \{M^k\}), \quad (3)
\]

\[
E(\{\tilde{P}_j^{(n)}\}^{k+1} \{M^k\}) \geq E(\{\tilde{P}_j^{(n)}\}^{k+1} \{M^k\}). \quad (4)
\]

The correctness of ineqs. (3) and (4) are guaranteed respectively by step 2 and step 3 of the optimization algorithm. Consequently, we can easily verify

\[
E(\tilde{P}_i;M^{k+1}) \geq E(\tilde{P}_i;M^k) \geq E(\tilde{P}_i;M^k) \geq \cdots \geq 0,
\]

which demonstrates that the optimization algorithm will finally converge in the iteration process.

### 2.3. Finalization

Upon the termination of the refining loop, we have obtained the adjusted projection matrices, which can be used to specify the projection pixel \( p^k \) in the \( k \)th image for all the vertices \( \tilde{P} \) on the mesh model. Consequently, a texture map \( T(\tilde{P}) = I^k(p^k) \) is generated with \( I^k(p^k) \) as the color of \( p^k \) in the \( k \)th image. We further define the global texture map as

\[
T(\tilde{P}) = \frac{\sum_{i=1}^{N} w_i^k(\tilde{P}) \cdot T^k(p^k)}{\sum_{i=1}^{N} w_i^k(\tilde{P})}.
\]
where \( w_k(\vec{P}) \) is the weight function introduced in [5], which specifies the contribution of the \( k \)th image to the texture at each mesh vertex \( \vec{P}_n \). With the synthesized global texture map \( T(\vec{P}) \), the textured face model is finally constructed.

### 3. Experimental results

We test our method on two sets of scanning data: one set comes from a real person’s face, and the other from a human skull. As the skull is rigid and hence easy to be rescanned, we use it mainly to test the effectiveness of our method.

The experimental data are listed in Table 1, and corresponding experimental results of the optimization algorithm are listed in Table 2. In addition, the variation of the average error during the optimization process is plotted in Fig.5. Here the average error is defined as the divided value of \( E(\vec{P}, M^k) \) by the sum of all the correspondences.

<table>
<thead>
<tr>
<th>Mesh Model</th>
<th>Number of Images</th>
<th>Number of Selected Vertices</th>
<th>Number of Correspondences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Face</td>
<td>8</td>
<td>11</td>
<td>66</td>
</tr>
<tr>
<td>Skull</td>
<td>5</td>
<td>12</td>
<td>48</td>
</tr>
</tbody>
</table>

Table 1. Experimental data

<table>
<thead>
<tr>
<th>Mesh Model</th>
<th>Iteration Cycles</th>
<th>Initial average Error (pixel)</th>
<th>Final average Error (pixel)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Face</td>
<td>9</td>
<td>5.15</td>
<td>1.02</td>
</tr>
<tr>
<td>Skull</td>
<td>5</td>
<td>67.30</td>
<td>1.01</td>
</tr>
</tbody>
</table>

Table 2. Experimental results of the optimization algorithm

Some figures for face modeling are shown in Fig.6. As a majority of the correspondences are located on the front face where obvious facial features (such as the corners of the mouth and eyes) exist, the visual effects on the front face of the textured model are better than that on the side faces, where feature misalignment (such as on the ears) exists.

Some figures for skull modeling are shown in Fig.7. Due to the inaccuracy of initially located correspondences for the 5th image of the skull, the initial error sum is amazingly big. However, after parameter adjustment in the refining loop, the error sum satisfactorily converges to a small value finally (corresponding data are shown in Table 2).

Comparison of the textured face model before and after the iteration process is shown in Fig.8. The obvious distortion of the textures has been greatly improved after the iteration process. In addition, comparison of the projections (in black color) of the skull mesh model before and after the iteration process is given in Fig.9. The projections of the mesh vertices are scattered in the whole image before the iterations. They are reorganized, however, in a regular region (the shape of which resembles that of the skull contour in the image) after 3 iteration cycles.

### 4. Conclusions

We proposed a camera-calibration based method to register mesh models with multi-view color images. A key feature of our method is that an iterative optimization algorithm is designed to overcome the sensitivity of the estimated projection parameters to the initially located correspondences. Experimental results have shown that the algorithm is robust and has good convergence properties.

However, there still exists a problem not addressed: due to the lack of obvious features in some facial areas, e.g. the chin of a person’s face, valid correspondences can hardly be located there. Thus, constraints are insufficient for the parameter estimation, resulting in distortion of textures outside the bounding region of the selected correspondences. Considering this problem, we plan to compare the renderings of the textured models with the input images, and define a new error function based on the difference of the colors for further adjustment of the
projection parameters. In this way, we taken into consideration the overlapping texture patches to reduce discrepancies in appearance due to mapping different input images.

5. References


Figure 6. Face modeling. (a) 3-D geometrical model (35927 vertices, 69999 triangles) (b) Images (c) 3-D textured model

Figure 7. Skull modeling. (a) 3-D geometrical model (50926 vertices, 100000 triangles) (b) Images (c) 3-D textured model

Figure 8. Comparison of the textured face model before (left) and after (right) the iterations

Figure 9. Comparison of the projection of the skull mesh model (black color) before (left) and after (right) the iterations