ABSTRACT

We present an algorithm for 3-D face modeling from a frontal and a profile view images of a person’s face. The algorithm starts by computing the 3D coordinates of automatically extracted facial feature points. The coordinates of the selected feature points are then used to deform a 3D generic face model to obtain a 3D face model for that person. Procrustes Analysis is used to globally minimize the distance between facial feature vertices in the model and the corresponding 3D points obtained from the images. Then, local deformation is performed on the facial feature vertices to obtain a more realistic 3D model for the person. Preliminary experiments to assess the applicability of the models for face recognition show encouraging results.

1. INTRODUCTION

3D face modeling has been an active area in computer graphics and machine vision. The need for fast head modeling for realistic facial animation and compression for video conferencing has led researchers to explore 3D model-based algorithms. The approaches in the literature for face modeling using 3D wired mesh differ in many aspects depending on the application, computing efficiency, type of sensors, cost, and required accuracy. In [1], a 3D generic face deformation is proposed for videophone applications, where depth is estimated from a single image. In [2], [3], and [4], they adapt a generic face to facial features extracted from images taken by an expensive 3D scanner. In [5] parallel stereo images were used to create 3D face models using differential geometry. The method requires a priori knowledge about the shape surfaces of the face and its differential geometry for accurate results.

In this paper, we propose to use the 3D coordinates of a set of facial feature points, calculated from images of a person’s face, to deform a generic face model to fit the calculated points. Our method uses stereo camera setup similar to the one in [3] and [6]. In this setup, two cameras with perpendicular optical axes take two orthogonal images of a person’s face. Corresponding landmark feature points in both images are automatically extracted and then depth is computed using perspective projection. The 3D generic model is initially deformed globally using procrustes analysis in which the rotation matrix is efficiently computed from orthogonal matrices of Singular Value Decomposition. This produces excellent initial results in centering and aligning the generic model. Local deformation is then accomplished by generating 3D spline curves for each facial component and adjusting corresponding vertices of the 3D model accordingly.

This paper is organized as follows: Section 2 describes the image acquisition setup. Section 3 describes our algorithm for automatic facial features extraction from two views. Section 4 introduces the generic 3D model used in this paper and describes our global and local deformation algorithms. Section 5 shows experimental results. Section 6 presents results of applying our models in face recognition. Conclusions and future work are given in section 7.

2. STEREO CAMERA SET-UP

We use the cameras setup of Fig.1 to simultaneously capture a frontal and a profile view images of the face. Assuming perspective projection, the 3D points are projected on the perspective rays passing through two corresponding projection centers C1 and C2, which are the camera centers separated by distance b from each other.

The focal length f is the distance of the image planes from each center of projection. Fixed at the center of each image are the points of intersection of the camera optical axis with the image plane. They are called the principal points of the frontal and profile views, (Xp, Yp) and (Zp, Yp). Finally, (Xf, Yf) and (Zf, Yf) are the projections of facial feature points on the frontal and profile views, respectively. From similar triangles P, C, P, and p, the projection coordinates for a 3D point (x,y,z) in the frontal image can be computed using:

\[ X_f = f \left( \frac{-x}{z} \right) + X_{p_o}, \quad Y_f = f \left( \frac{-y}{z} \right) + Y_{p_o} \]  

(1)
Similarly for the profile image
\[ Z_p = f \left( \frac{b-z}{b-x} \right) + Z_{p0}, \quad Y_p = f \left( \frac{-y}{b-x} \right) + Y_{p0} \] (2)
This system gives for every feature point four equations with three unknowns, i.e., x, y, and z. The solution to Eq.1 and Eq.2 can be expressed as:

\[ P = (A^T A)^{-1} A^T B \] (3)

where

\[ A = \begin{pmatrix} f & 0 & (X_f - X_{p0}) \\ 0 & f & 0 \\ (Z_f - Z_{p0}) & 0 & f \\ (Y_f - Y_{p0}) & f & 0 \end{pmatrix}, \quad B = \begin{pmatrix} z \\ 0 \\ (Z_f - Z_{p0}) b + bf \\ (Y_f - Y_{p0}) b \end{pmatrix} \]

3. AUTOMATIC FEATURE EXTRACTION

Our feature extraction module automatically extracts 15 corresponding feature points from each view, shown in Fig. 3. We use the YCbCr color space, where Y is the luminance component and Cb, Cr are the chrominance components. The feature extraction starts by detecting the centers of the eyes and the mouth in the frontal view using our algorithm in [8]. This helps limit the search space during the process of detecting the facial features. Then, it detects the eyes and the mouth by building two likelihood maps; the skin likelihood map and the mouth likelihood map. The skin likelihood map assigns to each pixel, X, the probability \( P(Ch(X),Cr(X)|skin) \), where the probability density function \( P(Ch,Cr|skin) \) is a Gaussian distribution obtained through training on a large set of skin patches. Fig.2.b shows the skin likelihood map for the image in Fig.2.a. It is clear that the eye regions have low probabilities because there is no skin in these regions. Using the information about the eye centers locations and thresholding, the regions of the eyes can be separated from the image. Then the boundaries are enhanced by performing erosion followed by dilation. The shape of each eye is then approximated by an ellipse and the major and minor axes are computed. The two ends of the major and minor axes determine four feature points for each eye. This process is repeated for the mouth except that \( P(Ch,Cr|skin) \) is replaced with \( P(Ch,Cr|mouth) \) to obtain the mouth likelihood map as shown in Fig.2.c.

![Figure 2. Result of skin and mouth probability functions.](image)

The above procedure finds the locations of 12 feature points, \((X_f,Y_f)\) in the frontal view, which will then need to be located, if visible, in the profile view. We search for each corresponding feature point in profile view along the Y coordinate, which is the same for both views, as shown in Fig.3. This means we only search for the Z coordinate in the profile view within known areas. We use the fact that the Y coordinates of the feature points are the same, or close, in both views to restrict the search space in one view once the corresponding feature point is located in the other view. To locate the tip of the nose, it is easier to locate it first in the profile view. The algorithm detects edges in the profile view between the Y coordinate of the upper feature point in the mouth and the Y coordinate of the lower feature point in the eye. The location of the edge point that has the minimum Z distance to the origin determines the tip of the nose in the profile view. This gives the Y coordinate for the tip of the nose in the frontal view. The X coordinate is chosen to be in the middle of the two eye centers.

![Figure 3. Facial feature selection.](image)

In the profile view, the upper and lower mouth points are found by searching for edges with minimum Z along the Y coordinate for the upper and lower mouth points in the frontal view. The eye feature points are located in the profile view by applying intensity correlation with a 3 x 3 average mask, along each of the Y coordinates obtained from the frontal view. The correlation results with minimum values give the locations of the eye feature points in the profile view. The location of the eye corner is further restricted to have larger Z than the lower eye feature point. The mouth corner point in the profile view is computed similarly. The Y coordinate of the nose corner is not known in both views. We locate the nose corners by using a correlation method similar to the one used for detecting the eye and mouth corners. The search is only performed around the Y coordinate of the tip of the nose and above the mouth upper feature point. The hidden feature points in the profile view are determined from the visible ones based on symmetry.

4. 3D GENERIC MODEL DEFORMATION

There are several 3D generic face models available in the literature. In this paper, we use the Candide model shown in Fig.4.a [7]. Our algorithm starts with the global deformation of the 15 selected vertices shown in Fig.4.b. Then, it applies local deformation to these 15 vertices and 14 additional vertices that are shown with the 15 vertices in Fig.4.c. The 3D model feature vertices are highlighted with black dots.

![Figure 4. Generic 3D model.](image)

4.1. Global deformation
The global deformation brings the entire 3D model vertices as close as possible to the corresponding 3D coordinates of the feature points calculated from the images. To achieve this goal, the 3D model must be rotated, translated, and scaled to match the calculated 3D points. Eq.4 gives the sum squared error between the calculated points and the corresponding transformed model points in terms of scale, rotation, and translation.

$$\text{Min } E(S,R,T) = \sum (P_j - P_{Mj})^2$$

where $P_M = [x_M \ y_M \ z_M]^T = S \cdot R \cdot [x_{M0} \ y_{M0} \ z_{M0}]^T + T$

Subscripts $M$, $I$, and $M0$ correspond to model points, calculated image points, and initial model points respectively. $S$ is the scale factor, $R$ is the rotation matrix, and $T$ is a translation vector. We use procrustes analysis [9] to obtain the parameters of $S$, $R$, and $T$ that minimize Eq.4. Given two sets of 3D points, namely the 15 calculated, $(x_i, y_i, z_i)$, and the 15 corresponding, $(x_{M0}, y_{M0}, z_{M0})$, model points, procrustes analysis finds the translation and rotation matrices that best match the corresponding data points. We summarize the deformation problem as follows. Let $P_I$ be an $N \times 3$ matrix containing the calculated 3D points from the images and let $P_M$ be the corresponding 3D model vertices, where $N$ is the total number of feature points or vertices.

a) Compute the mean of both $P_I$ and $P_M$

b) Center each set at its origin (i.e. $P_{M0} = (P_I - \text{mean}(P_I))$

c) Compute the norm of each set $||P_I||^2$ and $||P_{M0}||^2$

d) Normalize each set to equal unit norm (i.e. $P_{M0} = P_{M0} / ||P_I||^2$

e) Let $A = P_{M0}^T \cdot P_{M0}$

f) Compute the Singular Value Decomposition, SVD, of $A$, which results in the matrices $L$, $D$, and $M$ (i.e. $LDM = SVD(A)$)

g) Compute the rotation matrix $R = M L^T$

h) Compute scaling factor $S = \sum_{\text{diagonal}(D)} \frac{|P_I|^2}{|P_{M0}|^2}$

i) Compute the translating vector $T = \text{mean}(P_{M0}) - (S) \cdot \text{mean}(P_{M0}) \cdot (R)$

j) Transform $P_{M0}$ to $P_I$ using $P_I = S \cdot P_{M0} \cdot R + T$

k) Finally, transform the entire 3D model vertices using $S$, $R$, and $T$.

4.2. Local deformation

The global rigid deformation successfully scales and aligns the generic model to the 3D feature points calculated from the face images. Local deformation is then needed to treat the facial features as separate non-rigid components to bring the vertices of the generic model, in a more realistic manner, closer to the calculated 3D facial feature points. Fig.4.b shows the Candide model with 15 highlighted vertices. These vertices correspond to the 15 feature points that are calculated from the images. Fig.4.c shows the model with the vertices in Fig.4.b and 14 additional vertices, $(x_i, y_i, z_i)$. The local deformation is done in two steps. In the first step, the 15 feature vertices, shown in Fig.4.b, of the globally deformed model are replaced with the 15 feature points, $(x_i, y_i, z_i)$, calculated from the images. In the second step, the other 14 vertices are replaced with coordinates obtained from the interpolation of the calculated feature points for each facial component. For example using the four calculated 3D points of one eye, we use 3D interpolation to generate an approximate smooth curve with additional points connecting them, denoted by $(x_i, y_i, z_i)$. Those points are the result of interpolation between the two eye corner points and the upper point to generate the upper eye curve, as well as the interpolation between the two corner points and the lower point to generate the lower eye curve. The results are shown in Fig.5 for all the selected facial components. The coordinates of the 14 vertices are obtained from the interpolated points of the curves based on the minimum Euclidian distance between the 14 vertices, $(x_i, y_i, z_i)$, of the model and the interpolated points of the curve, $(x_i, y_i, z_i)$.

5. EXPERIMENTS

In this section we apply the deformation algorithms to images of 26 different people. Our accuracy measure for the global deformation is based on the mean square error between the 15 feature points computed from the images and the corresponding feature vertices from the model. In case of local deformation, we subjectively compare the projection of the model with the corresponding images. Fig.6 shows the results for one of the 26 cases that have been considered. The figure shows frontal and profile views of a given face followed by 3D global and local deformation of the generic model. Comparing the front view of...
the person’s face with local and global deformation we see that the model is adjusted to match the human face. At the profile view, it’s obvious that after the local deformation, the model has been adjusted at the nose to match the nose of the person. Fig.7 shows the near frontal view of the deformed model.

![Figure 7. The near frontal view and the updated model.](image)

6. APPLICATION TO FACE RECOGNITION

This section presents a simple application of our algorithm to human face recognition. The goal is to demonstrate the potential of the constructed 3D models for face recognition. We start by creating a database that contains a 3D face model for each person. Each model is a modified version of the generic face model based on a total of 29 person-specific 3D feature points, as described earlier in this paper. We make use of these features in recognition. Currently we have 3D models for 26 persons of various ethnic backgrounds and ages. For each person we captured two pairs of frontal and profile views. One pair is used for building the database model and the second is used for testing.

Given the two views of a test face, taken by the cameras setup of Fig.1, we obtain the coordinates of 29 feature points, as explained in the paper, through the global and local deformations. We then rigidly align the features of the test face with each face in the database using the global deformation explained section 4.1. Given the calculated model of a test face, we calculate the distance between each of the 29 vertices in the test model and the corresponding vertices in the database models. The database model that has the largest number of vertices close to the test model is selected as the recognized face. We tested the algorithm with all of the 26 faces in our database and found that 25 people were classified correctly, i.e. a recognition rate of about 96.2%. Table 1 shows the results of our experiments for the best case, the worst case, and the case of misclassification. From the table, under test face 23, we have eight numbers in the row labeled “right eye,” each indicates the face number in the database to which the corresponding feature point has been classified. In case of test face 23 all features of the right eye are classified correctly to belong to face number 23. At the “nose” row of test face 23, we see that the last feature is classified to belong to face number 16 in the database. Some feature points of the mouth were classified to belong to face 24 and face 16. The majority vote, which is 26 in this case, happens to belong correctly to face number 23 in the database. For test face 4, a total of 11 features were correctly classified to belong to face 4 in the database. Inspecting the table at test face 4, we can see that the second candidate for recognition is face number 21 appearing eight times in the column. From the experiments, we found that a good threshold for correct recognition is more than 10 feature points classified correctly. Test face 15 is the only case of misclassification with majority vote of seven features classified wrongly to belong to face number 18.

7. CONCLUSIONS AND FUTURE WORK

An algorithm for 3D face modeling is presented. It works by deforming a generic 3D model to fit a set of automatically extracted 3D facial feature points. The 3D feature points are calculated from orthogonal images of the face. The model is globally deformed to the person’s face using procrustes analysis followed by local deformation through vertices displacement according to the calculated and interpolated 3D points. Application to face recognition is presented. The quality of the results is encouraging. We are in the process of building a database of 3D models for more people and use them in 2D face recognition from different views.

### Table 1. Results of face recognition experiments.

<table>
<thead>
<tr>
<th>Test face 23</th>
<th>Test face 4</th>
<th>Test face 15</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Correct</strong></td>
<td><strong>Correct</strong></td>
<td><strong>Wrong</strong></td>
</tr>
<tr>
<td>Right eye</td>
<td>23, 23, 23</td>
<td>23, 23, 23</td>
</tr>
<tr>
<td>Left eye</td>
<td>23, 23, 23</td>
<td>23, 23, 23</td>
</tr>
<tr>
<td>Nose</td>
<td>23, 23, 23</td>
<td>23, 23, 23</td>
</tr>
<tr>
<td>Mouth</td>
<td>23, 23, 23</td>
<td>23, 23, 23</td>
</tr>
<tr>
<td>Maj. vote</td>
<td>26 for face 23</td>
<td>11 for face 4</td>
</tr>
</tbody>
</table>

8. REFERENCES


