Automatic 3D Face Recognition Using Shapes of Facial Curves

Chafik Samir*, Anuj Srivastava†, and Mohamed Daoudi‡

Abstract

In addition to facial textures, manifested in facial images, shapes of facial surfaces are also important in face recognition. We study facial surfaces by approximating them with indexed sets of level curves of a continuous function, such as the depth function, defined on these surfaces. Level curves of depth function are extracted efficiently using standard techniques on range images. The shapes of facial surfaces are then analyzed implicitly using shapes of these level curves, called the facial curves, using a differential geometric approach that computes geodesic lengths between these closed curves on a shape manifold. The metric for comparing facial surfaces is a composition of the metric involving individual facial curves. These ideas are demonstrated in the context of face recognition using the nearest-neighbor classifier.

1 Introduction

Automatic face recognition has been actively researched in recent years, and various techniques using ideas from 2D image analysis have been presented. Although a significant progress has been made, the task of automated, robust face recognition is still a distant goal. 2D Image-based methods are inherently limited by variability in imaging factors such as illumination and pose. An emerging solution is to use laser scanners for capturing surfaces of human faces, and use this data in performing face recognition [3]. Such observations are relatively invariant to illumination and pose, although they do vary with facial expressions. As the technology for measuring facial surfaces becomes simpler and cheaper, the use of 3D facial scans will be increasingly prominent.

A measurement of a facial surface contains information about its shape and texture (more precisely, the reflectivity function). Although, in general, one should utilize both the pieces for recognition, we restrict to an analysis of shapes to study their contribution in recognition. Given 3D scans of facial surfaces, the goal now is to develop metrics and mechanisms for comparing their shapes. What are the desirable properties of such a metric? Firstly, it should be provide a robust way of recognizing faces. That is, under this metric, faces of the same people, despite shape variability resulting from different facial expressions, should be at smaller distances and faces of different people should be at larger distances. If the face separations between different people are large, the resulting recognition procedure will be immune to the observation noise. Secondly, this metric should be relatively easy to compute. 3D scans of human faces provide large amounts of data (approximately 20-30 thousand vertices, edges, and so on), and a successful method should be able to efficiently analyze this data and compute the metric quickly. In fact, our framework uses approximations that emphasize a large gain in efficiency with a small loss of accuracy.

Over the last few years, a number of approaches have emerged for comparing shapes of facial surfaces (see [1] and [4] for discussions of different approaches). The earliest idea was to detect a set of feature locations – nose, nose bridge, eyes, lips, etc - in the face, and use their relative locations to characterize a face. The next idea was to generate range images from 3D scans and to utilize techniques from image analysis to recognize people [8]. A more challenging problem is to study the shape explicitly using the geometry of surface deformations [6, 7, 2], similar to the techniques developed for comparing shapes of anatomical objects. However, this problem is difficult because no canonical representation exists for comparing surfaces. If a canonical coordinate system existed, and the two surfaces were naturally registered, then any standard algorithm based on statistical and/or geometrical feature matching could be applied. Otherwise, one is

*ENIC Telecom Lille1 INT/LIFL, Villeneuve d'Ascq 59650, France
†Department of Statistics, Florida State University, Tallahassee, FL 32306, USA
‡ENIC Telecom Lille1 INT/LIFL, Villeneuve d'Ascq 59650, France

forced to impose an artificial coordinate system on surfaces, perhaps using algorithms for face alignment, and the resulting shape analysis inherently depends on this choice.

Our approach is to derive *approximate* representations of facial surfaces, and to impose metrics that compare shapes of these representations. We exploit the fact that curves can be parameterized canonically, using the arc-length parameter, and thus can be compared naturally. One can represent a surface using an indexed family of curves; a subset of these indices provide an approximate representation of the surface. Our idea is to represent a facial surface using a small collection of closed, planar curves, called the **facial curves**, and to compare facial surfaces implicitly by comparing the corresponding facial curves. This strategy raises a few issues: (i) how to define curves on facial surfaces?, (ii) how to extract these curves from 3D scan data?, and (iii) how to compare shapes of facial curves?

We address these issues as follows. The most common idea for defining curves on surfaces is to use level sets of functions. For instance, one can analyze the curvature tensor on a facial surface to obtain parabolic and ridge curves [4], or use surface derivatives to extract crest lines [9]. However, the computation of derivatives and curvatures involves second derivatives, and is very susceptible to observation noise. We suggest the use of height function, i.e. the value of z-coordinate at each point, and study level sets of this function as facial curves of interest. We acknowledge that this choice has an important limitation in that its level curves vary with changes in the gaze direction of the face. The shapes of these curves are however invariant to the remaining transformations such as planar motion and scale, and that is a strength of this choice. (One can even obtain complete pose invariance of facial curves by using functions such as the geodesic length function, but that idea is not explored further in this paper.) The next issue is the extraction of these level sets from the 3D scan data. Instead of extracting curves from 3D meshes directly, we do so using their range images and some standard tools from image analysis. This approach is much more efficient and robust compared to extracting curves from the mesh data. Once the curves are extracted, the remaining problem is to compare their shapes. The paper [5] describes a geometric approach for comparing shapes of closed, planar curves, without the use of landmarks, PDEs, or diffeomorphisms. This method constructs a shape space for curves of interest, and compares their shapes by constructing geodesic paths between candidate shapes. The length of this path provides an intrinsic metric for comparing shapes of curves. We will demonstrate this approach using FSU database consisting of 300 scans of 50 people.

Rest of this paper is organized as follows: Section 2 describes a representation of a facial surface using a collection of facial curves, and present metrics for comparing facial shapes under this representation. Section 3 describes the process of extracting facial curves from 3D face meshes, and Section 4 presents some experimental results. We finish the paper with a brief summary in Section 5.

2 Representation of Facial Shapes

Let S be a facial surface denoting a scanned face. Although in practice S is a triangulated mesh with a collection of edges and vertices, we start the discussion by assuming that it is a continuous surface. More precisely, it is an embedding of the upper unit hemisphere $\mathbb{S}^2_+$ in $\mathbb{R}^3$. In this definition, we have assumed that holes in S associated with eyes, nose, and mouth, are already patched. Some pictorial examples of S are shown in Figure 1 (top row) where facial surfaces associated with six facial expressions of the same person are displayed.

Let $F : S \mapsto \mathbb{R}$ be a continuous map on S. Let C_λ denote the level set of F, also called a **facial curve**, for the value $\lambda \in F(S)$, i.e. $C_\lambda = \{p \in S | F(p) = \lambda\} \quad \subset S$. We can reconstruct S through these level curves according to $S = \cup_\lambda C_\lambda$. Figure 1 (bottom left) shows some examples of facial curves along with the corresponding surface S. In principle, the collection $\{C_\lambda | \lambda \in \mathbb{R}_+\}$ contains all the information about S and one should be able to analyze shape of S via shapes of C_λs. In practice, however, a finite sampling of λ restricts our knowledge to a coarse approximation of the shape of S.

In this paper we choose F to be the **depth function**. Accordingly $F(p) = p_z$, the z-component of the point $p \in \mathbb{R}^3$. Our goal is to analyze shape of S invariant to action of the group of rigid motion $\mathbb{SE}(3) \equiv \mathbb{SO}(3) \ltimes \mathbb{R}^3$ on the surface S ($\ltimes$ implies a semi-direct product, that is the rotation is always applied before the translation). Let us investigate the variability of level sets of F with respect to these transformations. Rewrite $\mathbb{SE}(3)$ as $(\mathbb{SO}(2) \times \mathbb{S}^2) \ltimes (\mathbb{R}^2 \times \mathbb{R}^1)$, where we can interpret $\mathbb{SO}(2) \ltimes \mathbb{R}^2$ as a rigid motion in $x-y$ plane, i.e. perpendicular to the z axis, $\mathbb{S}^2$ as the direction of the z axis, and $\mathbb{R}$ as translation

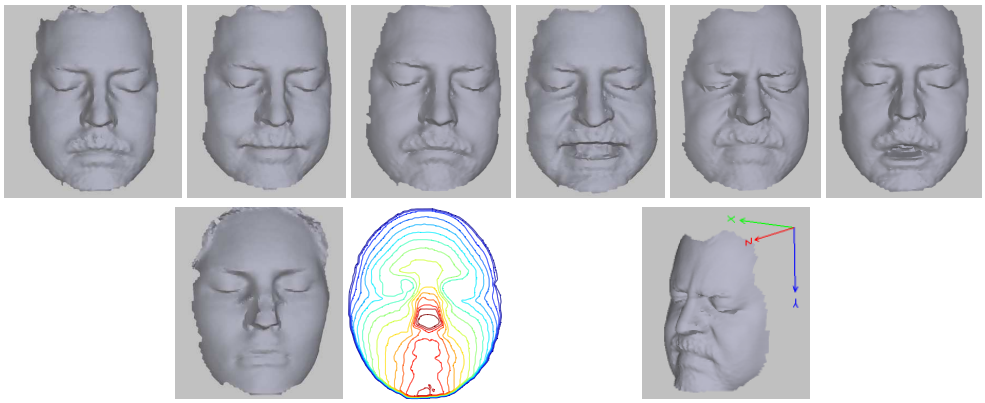


Figure 1: Top: Examples of facial surfaces of a person under different facial expressions. Bottom left: Examples of facial curves C_λ for a surface S. Bottom right: A coordinate system attached to the face.

in z direction. We are assuming that $x-y-z$ axes form a *body centered Cartesian coordinate system*, so that z axis is aligned with the gaze direction, as shown in Figure 1 (bottom right). Our technique for comparing shapes of closed curves will be automatically invariant to planar transformations in $\mathbb{SO}(2) \ltimes \mathbb{R}^2$ and the z-translations in $\mathbb{R}$. However, we have no simple way of removing variability due to the change in z direction (or gaze direction) that varies over the $\mathbb{S}^2$; in general, one has to search over all rotations of a face in $\mathbb{S}^2$ to best align it with another face. In this paper, we avoid this search by using facial scans that are collected while the subjects were staring at the camera. The advantages and disadvantages associated with this choice of depth function are as follows:

1. **Advantage**: For a smooth surface S, the depth function F is smooth and easy to compute. Furthermore, each level set C_λ of F is a planar curve. Given our past work on analyzing shapes of planar curves [5], we can directly apply those tools in this situation. This comparison is invariant to the rigid transformations of facial surfaces in the plane perpendicular to the body-centered z axis. In other words we do not worry about translation, rotation, and scaling of facial surfaces in $x-y$ plane. Furthermore, the translations along the z axis are easily accounted for by setting $z=0$ at the nose tip of each facial surface. Also, note that global scaling of S does not change the shapes of facial curves. In summary, aside from gaze direction, this representation automatically registers two facial surfaces with respect to other rotation, translation, and scale variables.

2. **Disadvantage**: Shapes of C_λs vary with changes in gaze direction. One can resolve this by either selecting another function for F that is invariant to this rotation, or by including an alignment step. However, the first option may lead to level sets that no longer planar, thus requiring a theory for analyzing shapes of curves in $\mathbb{R}^3$. In the dataset used in this paper, this problem is resolved by having all subjects stare directly at the camera while been scanned.

2.1 Comparing Shapes of Facial Curves

Consider facial curves C_λ as closed, arc-length parameterized, planar curves. Coordinate function $\alpha(s)$ of C_λ relates to the direction function $\theta(s)$ according to $\dot{\alpha}(s) = e^{j\,\theta(s)}$, $j=\sqrt{-1}$. To make shapes invariant to planar rotation, restrict to angle functions such that, $\frac{1}{2\pi}\int_0^{2\pi}\theta(s)ds = \pi$. Also, for a closed curve, θ must satisfy the *closure condition*: $\int_0^{2\pi}\exp(j\ \theta(s))ds = 0$. Summarizing, one restricts to the set $\mathcal{C} = \{\theta|\ \frac{1}{2\pi}\int_0^{2\pi}\theta(s)ds = \pi,\ \int_0^{2\pi}e^{j\theta(s)}ds = 0\}$. To remove the re-parameterization group $\mathbb{S}^1$ (relating to different placements of origin, point with $s=0$, on the same curve), define the quotient space $\mathcal{D} \equiv \mathcal{C}/\mathbb{S}^1$ as the shape space.

Let C_λ^1 and C_λ^2 be two facial curves associated with two different faces but at the same level λ. Let θ_1 and θ_2 be the angle functions associated with the two curves, respectively. An important tool in a Riemannian analysis of shapes is to construct geodesic paths between shapes and to use geodesic lengths as shape metric. Klassen et al. [5] provide a numerical procedure for computing geodesics between arbitrary points in $\mathcal{D}$. For

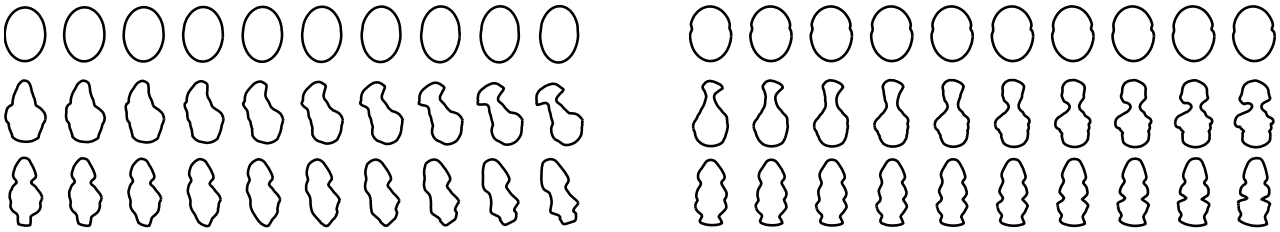


Figure 2: Geodesic paths between different facial curves.

any two shapes $\theta_1, \theta_2 \in \mathcal{D}$, they use a *shooting method* to construct the geodesic between them. The basic idea is search for a tangent direction g at the first shape θ_1, such that a geodesic in that direction reaches the second shape θ_2 in unit time. This search is performed by minimizing a "miss function", defined as a $\mathbb{L}^2$ distance between the shape reached and θ_2, using a gradient process. The geodesic is with respect to the metric $\langle g_1, g_2 \rangle = \int_0^{2\pi} g_1(s) g_2(s) ds$. This choice implies that a geodesic between two shapes is the path that uses **minimum energy to bend** one shape into the other. Shown in Figure 2 are some examples of geodesic paths between corresponding facial curves of two different facial surfaces. Let $d(C_\lambda^1, C_\lambda^2)$ denote the length of geodesic connecting their representatives, θ_1 and θ_2, in the shape space $\mathcal{D}$. This distance is independent of rotation, translation, and scale of the facial surfaces in the $x - y$ plane.

2.2 Metric for Comparing Facial Shapes

Now that we have defined a metric for comparing shapes of facial curves, it can be easily extended to compare shapes of facial surfaces. Assuming that $\{C_\lambda^1 | \lambda \in \Lambda\}$ and $\{C_\lambda^2 | \lambda \in \Lambda\}$ be the collections of facial curves associated with the two surfaces, two possible metrics between them are:

$$d_e(S^1, S^2) = \left(\sum_{\lambda \in \Lambda} d(C_\lambda^1, C_\lambda^2)^2 \right)^{1/2} \quad \text{and} \quad d_g(S^1, S^2) = \left(\prod_{\lambda \in \Lambda} d(C_\lambda^1, C_\lambda^2) \right)^{1/|\Lambda|}. \tag{1}$$

d_e denotes the Euclidean length and d_g denotes the geometric mean. Here Λ is a finite set of values used in approximating a facial surface by facial curves.

The choice of Λ is also important in the resulting performance. Of course, the accuracy of d_e and d_g will improve with increase in the size of Λ, but the question is how to choose the elements of Λ. In this paper, we have sampled the range of depth values uniformly to obtain Λ.

3 Data collection and Curve Extraction

In this section we describe the process of extracting facial curves from the scanned 3D meshes. Our general approach is to convert 3D scanned meshes into range images, and then use techniques from standard image analysis to extract level curves.

Data Collection: The dataset used in these experiments was collected using a Minolta vivid 700 scanner, in a controlled imaging environment shown in the left panel of Fig. 3. Subjects were imaged in a closed room with fixed uniform illumination and using markers to align their gaze directions. Each subject was scanned under six different facial expressions, as shown in the right panels of Fig. 3. To avoid missing data, the subjects were asked to close their eyes during scanning. The scanner outputs a triangulated mesh of points sampled from the facial surfaces.

Triangulated Surfaces: Using Gnu Triangulated Surface (GTS open source library), we first converted all observed meshes into GTS format triangulation. This format allows us to: (i) fills holes resulting from missed triangles on the mesh, and (ii) obtain coarse (or refined) representations of a surface by easily decreasing (or increasing) the number of edges. This preprocessing allows us to obtain high-resolution range images, and to ensure smoothness of extracted level curves. It also avoids holes in the resulting range images.

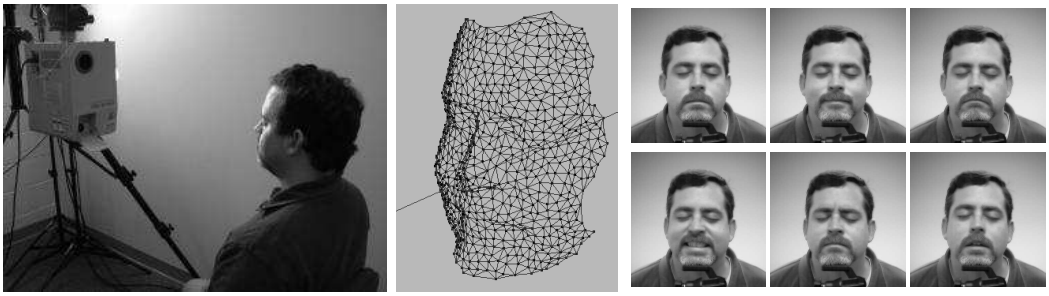


Figure 3: Data capture: subjects gaze at the scanner to obtain similar gaze direction. Each subject was scanned for six different facial expressions.

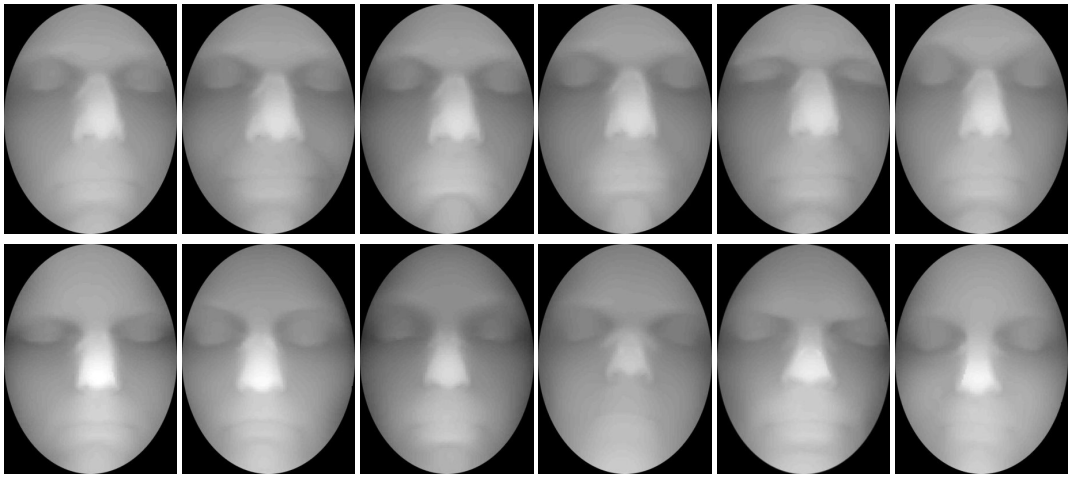
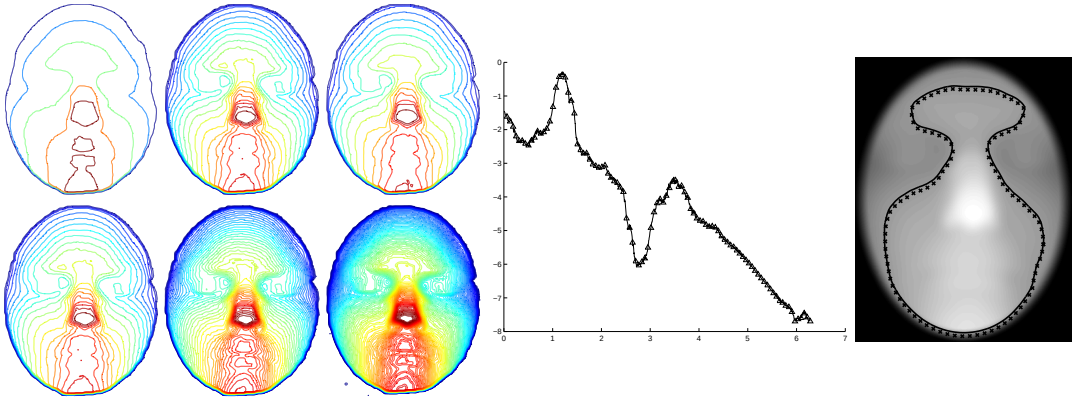


Figure 4: Top row: Range images of a subject's face under six different facial expressions. Bottom row: Range images of six different subjects under the same facial expression.

Range Image Generation: A range image is a rectangular array of pixels in an image plane, with pixels values being proportional to the distance (or depth) of the nearest point on the surface, in direction perpendicular to the image plane. Although these depth values, or z-coordinates, are given for vertices present in the mesh, they may not project uniformly to pixel locations in the image plane, thus leaving holes in a range image. For each triangle in the mesh, we find pixels in the image plane that are interior to the projection of that triangle using the line crossing algorithm. Then, for each pixel inside the projected triangle, we compute the z-coordinate using a linear interpolation of the z-coordinates of three vertices of that triangle. If multiple triangles assign range values to the same pixel, then the smallest pixel value is selected. We also crop the range images near the extremities of the image, using a standard masking image. Shown in Figure 4 are some examples of the resulting range images.

Level Curve Extraction: Extraction of level sets, of functions defined on 3D meshes, is not straightforward. Sparse data points and undersampled regions make this process difficult. One often uses the geometry of meshes to accomplish this task [9, 4]. However, we exploit the fact that the level sets of the depth function are planar. Therefore, one can utilize 2D image analysis to perform curve extraction, and thus avoid dealing with 3D issues. One of the reasons for selecting the depth function is that using range images, it is relatively straightforward to extract level curves. We remark that in general image processing, level curves are not popular because images often consist of "flat" regions, i.e., regions of constant brightness, where the level curves are not well defined. However, we expect range images of facial surfaces to be mostly smooth and non flat, with well-defined level curves. Only the regions containing eyes and lips may contain range discontinuity but we prevent it by using interpolation techniques while filling holes. Also, note that there are plenty of



(a) (b)

Figure 5: (a): Level sets of depth function for several levels. (b): Angle functions, observed (marked) and fitted (solid), for a level curve of the depth function in the range image shown in right.

level curves, i.e., one can extract almost as many as needed by the application. Following steps summarize the extraction of level curves from range images: (i) Smooth the range image using a Gaussian filter, to help improve the extraction performance. Such smoothing does not significantly change the shape of resulting facial curves, and therefore is a valid step in pre-processing data. (ii) Extract pixel locations at a certain range, say λ_0. (iii) Interpolate between the extracted points to build a continuous curve, closed curve. The last two steps can be implemented using *imcontour* in matlab.

Some examples of extracted level curves are shown in Figure 5(a). For the same facial surface S, these panels shown an increasing number of level curves extracted using the procedure described above. It must be noted that for certain levels, we can get multiple connected components, i.e. there can be multiple closed curves at the same level. In this paper, we have avoided using those λs that result in multiple components, although one can easily extend the algorithm to take this into account.

Angle Function Representation: For an observed contour, denoted by an ordered set of non-uniformly sampled points in $\mathbb{R}^2$, one can generate a representative element $\theta \in \mathcal{D}$ as follows. For each neighboring pair of points, compute the chord angle θ_i and the Euclidean distance s_i between them. Then, fit a smooth θ function, e.g. using splines, to the graph formed by $\{(\sum_{j=1}^{i} s_j, \theta_i)\}$. Finally, re-sample θ uniformly (using arc-length parametrization) and project onto $\mathcal{D}$ using techniques described in [5]. Shown in Figure 5(b) right panels is an example. In the left panel, we show the original graph $\{(\sum_{j=1}^{i} s_j, \theta_i)\}$ in marked line, and a smooth fitted function in solid line. The corresponding facial curves are shown in the right panel, superimposed on the original range image. Figure 6 some more examples of the fitted smooth curves.

The main computational steps of our procedure can be summarized as follows: (i) Mesh refinement, generation of range images, and masking of the range images. For a mesh approximately 20K points, this step takes less than a second. (ii) Level curves extractions from range images. Using *imcontour* in matlab, this step is instantaneous. (iii) Computation of an angle function for each of the extracted curve, fitting a spline through its graph and re-sampling for arc-length parametrization. It takes less than 0.5 second in matlab to generate facial curves for one face scan. And finally, (iv) Computation of geodesic lengths between respective facial curves, and calculation of a distance between facial surfaces. It takes less than 0.1 second to compute d_e and d_g.

4 Experimental Results

In this section we present some experimental results to demonstrate effectiveness of our approach. First, we demonstrate the success d_e and d_g by presenting a matrix of pairwise distances between a small set of faces. This matrix shows that faces for the same people are closer than faces of different people. We further emphasize that idea using a simple clustering example. We cluster a small number of facial surfaces using

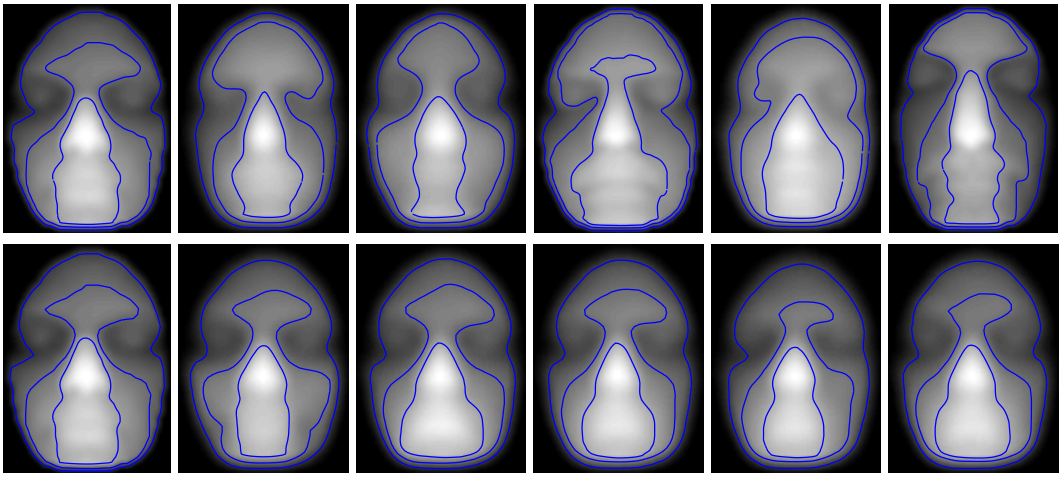


Figure 6: Three level sets in each surface. Top: same facial expressions, six different subjects. Bottom: six facial expressions, same subject.

dendrogram clustering, and demonstrate that facial surfaces of sample people are clustered together, despite having different facial expressions. Secondly, we setup a recognition experiment where we use a fraction of faces as training and the remaining as test, and use the nearest neighbor classifier to classify the test faces. These results are compared to image-based PCA of range and texture images.

Each of the faces used here were scanned at a high resolution, containing approximately 200,000 vertices after mesh refinement. The original range images generated were of size 1201×900, but we down-sampled them to 376×449. From each range image we extract 1,2,..., 5 curves, depending upon the experiment, and used the aforementioned geodesic program to compute pairwise distances d_e and d_g.

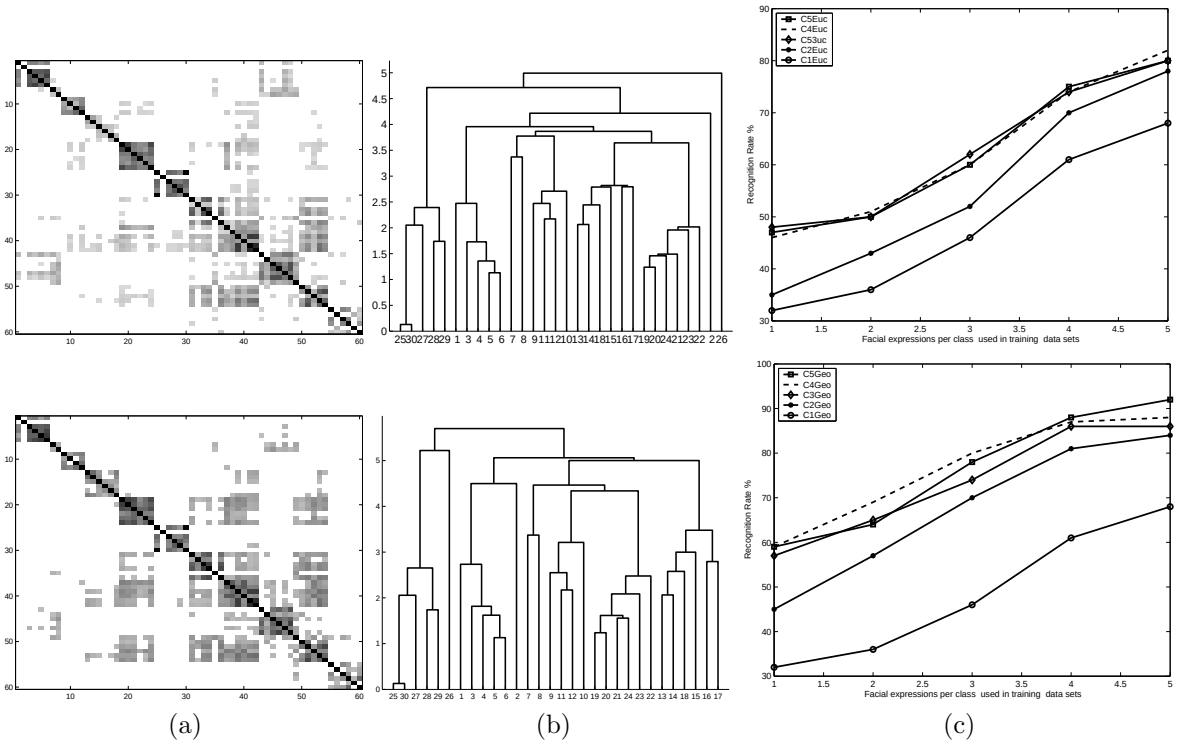
4.1 Distances Matrices

In the first experiment, we considered 60 faces (six facial expression each for ten persons). These faces are labelled in order, i.e. 1-6 for person 1, 7-12 for person 2, etc. We computed the distance $d(S^1, S^2)$ for each pair and the results are shown in Figure 7(a). The top panel uses d_e while the bottom panel uses d_g. To improve the display we have truncated the values above a certain threshold. (If the pairwise distance exceeds a certain value, we have set that pixel to be white, while smaller distances are denoted by darker pixels.) It is easy to see that both d_e (top) and d_g (bottom) are successful in imposing smaller distances between different face scans of the same person.

To further demonstrate these metrics, we have performed a dendrogram clustering of faces using pairwise distances. In this experiment we used restricted to 30 facial surfaces associated with five people (as dendrogram becomes crowded with more data points). Shown in Figure 7(b) are the two dendrogram resulting from d_e (top) and d_g (bottom), respectively. As these pictures illustrate, the clustering is quite successful in both cases. The six facial surfaces for a person are mostly clustered together, away from the clusters for other persons. There are only two exceptions (erroneous clusterings) in each case: Faces 2 and 26 are not clustered with their classes in the top panel, and Faces 7 and 8 are not clustered with their class in the bottom panel.

4.2 Nearest Neighbor Recognition

In this experiment we used a total of 300 facial surfaces (six facial expressions each for 50 persons). We divide this set into training and test sets by taking r ($r = 1, 2, \ldots, 5$) faces per person as labeled training data, and the remaining $6 - r$ faces/person as test data. Then, we use nearest neighbor classifier and the distance $d(S^1, S^2)$ to classify each test face. Since we know the true class of the test face, we can compute



(a) (b) (c)

Figure 7: (a): Pairwise distances between 60 facial surfaces. (b): Dendrogram clustering between facial surfaces, indexed 1 to 30. (c): Recognition rate plotted versus training data size per person.

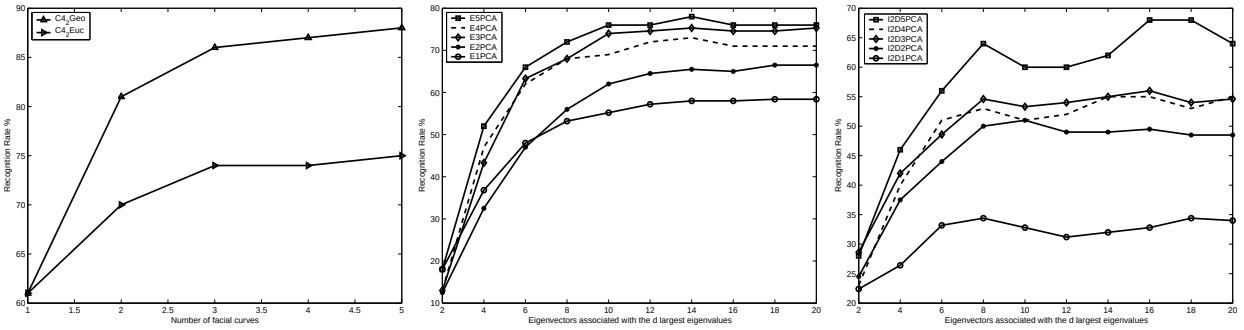
the percentage of correctly classified test faces. This recognition performance studied by varying tje setup as follows.

First, we computed the recognition performance by changing r, the number of training faces. Shown in Figure 7(c) are results of this experiment. Each curve denotes a different number of curves used (in coarsely approximating a facial surface). For example, $C1$ denotes the recognition performance obtained when only one curve was to represent a facial surface ($|\Lambda = 1|$). Similarly, $C2$ uses two curves, $C3$ uses three curves and so on. The top figure is for d_e and the bottom figure is for d_g. $CnEuc$ is the performance for Euclidean average, while $CnGeo$ is that for the geometric mean.

As this figure illustrates, the recognition performance steadily increases with increase in r. This result is intuitive as more training data generally implies a better classification performance. Another interesting point is that for a fixed r, the recognition performance initially increases with the number of curves and then saturates as we reach four and five curves. This idea is better illustrated in Figure 8(a), where the recognition performance is plotted against the number of curves for a fixed $r = 4$; the two curves correspond to d_e and d_g, respectively. In this experiment involving 50 people, and 300 face scans, five facial curves per surface seem sufficient to reach the peak performance.

The performance of nearest neighbor recognition strongly supports the idea of using geometries facial curves to recognize people. Even with one curve per face, and one training data, we achieve more than 30% recognition rate. Remember that uniform sampling will result in only $1/50 = 2\%$ recognition rate. Using five training faces per person, and five facial curves per face, we can achieve a recognition rate of almost 92%. Considering that we perform a coarse sampling of a facial surface and that we completely ignore the surface textures, this rate is quite significant and points to the possibility of practical 3D shape-based face-recognition systems.

To compare our results, we have used the traditional image-based PCA for face recognition in two different ways. In the first case, we use the PCA of range images, of size 300×400 to perform recognition, and in the second we use the PCA of color (texture) images of size 130×130. The PCA procedure is standard: reshape



(a) (b) (c)

Figure 8: Recognition rates using (a) facial curves, (b) PCA of range images, (c) PCA of texture images.

images into column vectors, perform singular value decomposition of data matrix, and utilize the first d left singular vectors for dimension reduction. The Euclidean distance in $\mathbb{R}^d$ is used to define a nearest neighbor classifier for the test images. Figure 8(b) shows the recognition rate for range images and (c) shows that for the texture images. The performance is plotted versus d for $r = 1, 2, \ldots, 5$ number of training images and $6 - r$ test images per person. We clarify that the range images happened to be aligned in translation and rotation, while the texture images are aligned only in rotation and not in translation.

5 Summary

We have described a geometric approach for comparing shapes of facial surfaces via the shapes of facial curves. The basic idea is to coarsely approximate a facial surface S with a finite set of level curves, called the facial curves, of the height function on S. Curve extraction is accomplished using range images, and metric between facial curves are computed using a method described in [5]. A metric on shapes of facial surfaces is derived by accumulating distances between corresponding facial curves. Results are presented from clustering and recognition of facial surfaces according to this metric.

The strengths of this approach are: (i) Other than alignment of the gaze directions, this method does not require additional face alignment. (ii) In contrast to the methods that utilize curvature based analysis of surfaces, this method is quite robust to observation noise. (iii) The utilization of range image allows standard techniques from image analysis to extract level sets. Also, the technique used here for comparing shapes of facial curves does not require any landmarks. The whole procedure is automated in that regard. (iv) In principle, this approach can handle holes in the observed surfaces. If the level curves do not pass through these holes, then there is no impact on the results. In case, the curves pass through the holes, once can use standard curve fitting algorithms to complete them smoothly. Just like shape analysis, completion of curves is much simpler than completion of surfaces.

References

[1] K. W. Bowyer, K. I. Chang, and P. J. Flynn. A survey of approaches to 3D and multi-modal 3D+2D face recognition. In *Proc. 17th International Conference on Pattern Recognition*, Cambridge, UK. poser.

[2] A. M. Bronstein, M. M. Bronstein, A. Spira, and R. Kimmel. Face recognition from facial surface metric. In *Proc. 8th European Conference on Computer Vision*, volume 3022 of *Lecture Notes in Computer Science*, pages 225–237, Prague, Czech Republic, 2004. Springer-Verlag.

[3] K. I. Chang, K. W. Bowyer, and P. J. Flynn. An evaluation of multi-modal 2D+3D face biometrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):619–624, 2005.

[4] P. W. Hallinan, G. G. Gordon, A. L. Yuille, P. Giblin, and D. Mumford. *Two- and Three-Dimensional Patterns of Face*. A. K. Peters, 1999.

[5] E. Klassen, A. Srivastava, W. Mio, and S. Joshi. Analysis of planar shapes using geodesic paths on shape spaces. *IEEE Pattern Analysis and Machine Intelligence*, 26(3):372–383, March, 2004.

[6] X. Lu and A. K. Jain. Deformation analysis for 3D face matching. In *Proc. 7th IEEE Workshop on Applications of Computer Vision*, pages 99–104, Breckenridge, CO, 2005.

[7] M. I. Miller and L. Younes. Group actions, homeomorphisms, and matching: A general framework. *International Journal of Computer Vision*, 41(1/2):61–84, 2002.

[8] A. Srivastava, X. Liu, and C. Hesher. Face recognition using optimal linear components of range images. *Journal of Image and Vision Computing*, to appear, 2005.

[9] S. Yoshizawa, A. G. Belyaev, and H.-P. Seidel. Fast and robust detection of crest lines on meshes. In *Proc. of ACM Symposium on Solid and Physical Modeling*, pages 227–232, 2005.