Shape Matching of 3-D Contours using Normalized Fourier Descriptors

Hao Zhang Eugene Fiume

Department of Computer Science
University of Toronto
10 King's College Road
Toronto, Ontario Canada M5S 3G4
E-mail: richard, elf@dgp.utoronto.ca

Abstract

In this paper, we develop a simple, eigen-space matching algorithm for closed 3-D contours. Our algorithm relies on a novel method which normalizes the Fourier descriptors (FDs) of a 3-D contour with respect to two of its FD coefficients corresponding to the lowest non-zero frequencies. The remaining matching task only involves vertex shift and rotation about the z-axis. Our approach is inspired by the observation that the traditional Fourier transform of a 1-D signal is equivalent to the decomposition of the signal into a linear combination of the eigenvectors of a smoothing operator. It turns out that our FD normalization is equivalent to aligning the limit plane approached by the sequence of progressively smoothed 3-D contour with the xy-plane.

1. Introduction

Shape matching and the measurement of shape similarity are essential tasks in several computer vision problems, including model-based object recognition, classification, and retrieval. Earlier work on the topic focus primarily on planar shapes [7, 12, 13, 14, 17, 19, 21]. For an extended introduction of these techniques, please refer to any of the several survey papers [6, 9, 18]. Matching of 3-D surface models is considerably more difficult [1, 15], and is just starting to receive more attention [4, 8, 20].

Planar shapes are usually characterized by, and compared with respect to, their 2-D contours [7, 12, 13, 19, 21]. One of the main reasons for the popularity of contour-based shape analysis techniques is that edge detection constitutes an important aspect of shape recognition by the human visual system and by psycho-visual experiments studying eye movements [9].

Contour-based methods also tend to be more efficient and intuitive than those based on moment integrals [17] or structural information [14]. In particular, it is difficult to correlate high-order moments with shape features [6], and structural methods, especially those using graph-like representations [14], usually lead to variants of the computation-intensive graph isomorphism algorithm [9].

In this paper, we study the problem of matching *3-D contours*, which are closed curves embedded in $\mathbf{R}^3$. This is more difficult than the 2-D problem, but more manageable than general surface matching, since generalization of known 2-D techniques is possible. On the other hand, efficient matching techniques for 3-D contours are quite useful in the surface case, if appropriate characteristic contours can be extracted from the surface models [1].

1.1. Related work and motivation

The first notable matching technique for 2-D contours is due to Zahn and Roskies [21], where a contour is represented by a 1-D, arc-length parameterized signal of cumulative turning angles. The Fourier transform, or the *Fourier descriptors* (*FDs*), of the signal are used to represent the curve. Other measures, such as curvature [13] and moments [7], can also be used to define the signal.

A problem with such descriptors is that a closed curve is not always closed when it is reconstructed from a truncated set of FDs. Persoon and Fu [11] remedies this by treating a 2-D contour as a *complex-valued* signal, and the FDs are still defined as its Fourier transform.

In both cases, the similarity distance between two contours is defined as the *smallest* possible sum of componentwise differences between the corresponding FDs; the contours are free to undergo any rigid transformation, and the choice of the starting point on the contour can be arbitrary. Finding the best way to match two contours is possible through non-linear optimization [11], but the computational cost is formidable when interactivity is desired.

Wallace and Wintz [19], and more recently, Rui et al. [12], propose the use of *normalized Fourier descriptors* (*NFDs*) to match 2-D contours more efficiently. The average position, i.e., the centroid, of the contour vertices are aligned with the origin. Scale normalization is accomplished by dividing all vertices by the magnitude of the largest FD coefficient. Finally, the *orientation of the major axes* of the shapes are aligned exactly before a much simplified shape matching step is performed.

The price to pay for efficiency is that these approaches using NFDs are not guaranteed to find the optimal matching. But if the two contours are sufficiently close to each other, which is often the case in applications such as image classification and retrieval, the distance found will generally be very close to the optimum [19].

Let us now turn our attention to the matching of 3-D contours. We first argue that current 1-D or 2-D techniques do not work for 3-D contours as is. For example, the transformation of a 3-D contour into a 1-D signal of curvature would lose too much information, as shown in Figure 1.

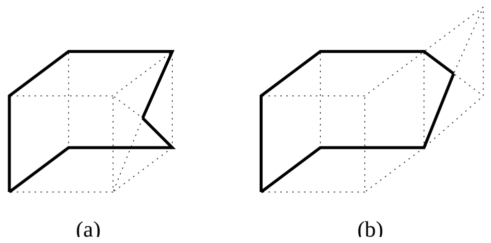


Figure 1. Two very different contours can have the same arc-length parameterized curvature distribution.

To represent a 3-D contour parametrically and uniquely, higher-order differential invariants, such as torsion [2, 5] or Gaussian curvature, need to be used. Accurate measurements of these differential quantities usually require the construction of a fourth-order B-spline approximation of the sampled contour. Although B-splines offer local control, they are sensitive to quantization or measurement noise, and a pre-smoothing step using explicit energy minimization is often necessary [2].

Data noise can be conveniently filtered out using truncated sets of FDs. Since noise contributes only to FD coefficients corresponding to high frequencies, they can simply be ignored in shape matching. But to be able to match 3-D contours using FDs, current 2-D techniques [11, 12, 19, 21] have to be generalized to handle the extra degrees of freedom in the rigid transformations allowed.

A crucial difference caused by the increase in dimension lies in FD normalization. Existing 2-D methods [12, 19] rely on the fact that the major axis of a planar shape is characterized by a *single* 2-D vector. After proper normalization with respect to centroid and scale, the major axes of two planar shapes can be aligned *exactly* by a rotation. However, the major axis of a 3-D contour is characterized by *two* 3-D vectors, and there rarely is a set of rotations which would align the major axes of two 3-D contours exactly.

Previous matching methods for 3-D curves [2, 5] combine polygonal pre-smoothing, B-spline approximation, and geometric hashing [20] to locate maximum matchable *curve portions* between a test curve and a model curve. Matching is based on estimated differential quantities invariant under rigid transformation. Since a curve segment is defined using a *sequence* of control points, there is no need to handle different choices of the starting point for the parameterization. To achieve better robustness, Pajdla and van Gool [10] propose the use of semi-differential invariants.

In this paper, we are concerned with matching the shape of 3-D contours *globally*. For many practical applications, the matching of overall shapes is more essential than that of local features. This, along with the inevitable presence of noise in the raw data, warrants the use of FDs. We generalize the notion of *NFDs* [19, 12] to handle 3-D contours, and develop a matching algorithm that is simple, efficient, and easy to implement. We demonstrate the effectiveness of our approach both theoretically and empirically.

1.2. Overview of approach and contribution

We formulate the discrete Fourier transform of a 3-D contour as a decomposition with respect to the eigenvectors of a *smoothing operator*. Each FD coefficient obtained is a complex 3-D vector. Separating the real and imaginary parts of the complex numbers, we can view an FD coefficient as being characterized by two real 3-D vectors.

To match two contours, we obtain their FDs after centroid and scale normalization. We generalize the normalization of the major axes of planar shapes to the alignment of the two FD coefficients corresponding to the lowest non-zero frequencies. Indeed, these coefficients capture the most dominant component of the overall shape.

We show that our normalization problem reduces to the alignment of two pairs of 3-D vectors, (u_1, v_1) and (u_2, v_2), under vertex shift and rotation. We make the key observation that the plane spanned by u_i and v_i, $i = 1, 2$, is invariant under vertex shift. Aligning these two planes first has a nice geometric interpretation, and it reduces the computational cost greatly. The remaining matching task only involves vertex shift and rotation about the z-axis; we show how this can be carried out efficiently.

The complexity of our algorithm is $O(kn)$, where n is the number of point samples along the contour, and $k \leq n$ is the number of FD coefficients needed to measure the similarity distance. Note that in practice, not many FD coefficients are need for shape discrimination; this is especially true if the number of samples is large. We also point out the possibility of reducing the complexity of our algorithm to an optimal $O(n \log n)$.

To the extent of our knowledge, our approach is the first generalization of NFDs to handle 3-D contours. The choice of eigen-decomposition over traditional Fourier transform formulations not only offers a great deal of geometric intuition, it also lays some groundwork for the generalization of this kind of techniques to handle 3-D meshes.

Without an appropriate functional representation for a mesh with irregular connectivity, a Fourier transform cannot be constructed. But various discrete smoothing operators have natural 3-D generalizations. Eigen-decompositions of a mesh with respect to such operators can be quite useful in a number of mesh processing tasks. The best example of this work in computer graphics is Taubin's signal processing approach for mesh smoothing [16].

Finally, as we have mentioned, an efficient matching technique for 3-D curves alone can be used in the analysis and recognition of surface models, provided that a set of characteristic curves have been identified.

1.3. Organization of the paper

In the next section, we briefly recall the traditional formulation of FDs, and define the similarity measure we use. In Section 3, we present the smoothing operator, its eigenproperties, and a theorem concerning its limit behavior. In Section 4, we describe our matching algorithm using the NFDs we develop. Experimental results are shown in Section 5 to support our approach. Finally, we conclude in Section 6 with some suggestions for future work.

2. Traditional FDs and our similarity distance

In this paper, each 3-D contour a is characterized by a sequence of sample points along a, which are its vertices and are denoted by $a_1, a_2, \ldots, a_n$ in order. We represent a by an $n \times 3$ *coordinate matrix* $\mathbf{a}$, consisting of an ordered list of vertex coordinates. Note that a is simply a closed 3-D polygon, and for convenience, we often address it using its coordinate matrix.

Let a and b be two 3-D contours. The traditional *FDs* A (and B) of a (and b) are defined using the usual discrete Fourier transform. Specifically, each A_k is a vector in $\mathbf{C}^3$ given by $A_k = \sum_{j=1}^{n} a_j e^{-i\frac{2\pi jk}{n}}$, where $k = 1, \ldots, n$. The set of A_k's are referred to as the *FD coefficients* of a.

In this paper, we define the *similarity distance* between a and b as the *minimum* of the following L_2 measure:

$$\mu(A, B) = \sqrt{\sum_{k=1}^{n} w_k \left[\sum_{j=1}^{3} |A_{k,j} - B_{k,j}|^2\right]}, \qquad (1)$$

under arbitrary translation, scale, rotation of the contours, and vertex shifts, where $|\cdot|$ denotes the norm of a complex number. Of course, the FD coefficients will be derived in a different way using our NFD approach.

The weights w_k can be chosen in a variety of ways. For example, noise tolerance can be achieved by setting $w_k = 0$ whenever the k-th FD coefficient captures a high frequency contribution. For simplicity, we use uniform weighing in our experiments, i.e., $w_1 = \ldots = w_n = 1$. In either case, our similarity distance is indeed a *metric*, since it is equivalent to an Euclidean distance measure between points in $\mathbf{R}^m$ for some positive integer m.

Our similarity distance is a trivial generalization of the one used by Wallace and Wintz [19] to 3-D, and with the addition of the square root to obtain a metric. Using an L_∞ measure instead of L_2 appears to make little difference [19]. On the other hand, the variance measure of Rui et al. [12] does not appear to have an easy 3-D generalization.

Finally, a measure $\hat{\mu}(A, B) = \sum_{k=1}^{n}(|A_k|^2 - |B_k|^2)^2$ that is invariant under both rigid transformation and vertex shift has been used for the recognition of hand-written characters [9]. But observe that for general contours in 3-D, this measure would produce many false positives. For example, a contour a can look quite different from one that is obtained from a by shifting only the y coordinates, while the distance reported by $\hat{\mu}$ would be zero.

3. Eigen-decomposition and associated FDs

In this section, we prepare the necessary theory for our matching algorithm. We derive the FDs of a 3-D contour through a decomposition using the eigenvectors of a *midpoint smoothing* operator. We analyze the eigenproperties and the limit behavior of midpoint smoothing, with emphasis on the underlying geometric intuition. We show that with a uniform dilation, repeated applications of the midpoint smoothing operator force a 3-D contour to converge to a limit plane. In particular, two spanning vectors of this limit plane correspond to the dominant FD coefficient of the original contour.

In what follows, the letter i is reserved for $\sqrt{-1}$, the imaginary unit. The n-th roots of unity are denoted by $z_1, \ldots, z_n$, where $z_j = e^{i\frac{2j\pi}{n}}$. Clearly, we have $z_n = 1$ and $z_j = z_1^j$, $j = 1, \ldots, n$. Finally, we denote the transpose of a matrix M by M^T, the conjugate by $\overline{M}$, and the conjugate transpose by M^*.

3.1. The midpoint smoothing operator

Let $\mathbf{a}$ be the original 3-D contour, i.e., a closed polygon, with n vertices. The *midpoint polygon* $\mathbf{a}'$ of $\mathbf{a}$ is the polygon obtained by connecting the midpoints of all the line segments of $\mathbf{a}$ in order. The midpoint smoothing operation can be captured by the *n-th order midpoint smoothing operator* S_n, with $\mathbf{a}' = S_n\mathbf{a}$. The operator S_n is an $n \times n$ matrix whose ij-th entry is 1/2 if $j = i$ or $j = i + 1$ mod n, and 0 otherwise.

It is interesting to note that S_n is invertible if and only if n is odd. For any odd number n, S_n^{-1} has 1's all along its main diagonal, and all along the diagonal below. In all other diagonals, the entries alternate between 1 and -1. So it makes sense to call these midpoint polygons the *midpoint signatures*, at different smoothness levels, of the original polygon.

Midpoint smoothing is closely related to *uniform Laplacian smoothing* [16]. After one step of Laplacian smoothing of the polygon $\mathbf{a}$, we yield the vertex transformation

$$\mathbf{a}'_j = \frac{1}{4}\mathbf{a}_{j-1} + \frac{1}{2}\mathbf{a}_j + \frac{1}{4}\mathbf{a}_{j+1}. \qquad (2)$$

It is easily seen that two steps of midpoint smoothing corresponds to one step of Laplacian smoothing. This connection helps us obtain a geometric interpretation of the frequencies associated with the FD coefficients we derive later.

We choose to use the midpoint smoothing operator to construct an eigen-decomposition since, as we shall see in the following sections, the eigenvectors of our smoothing operator has a simpler form, and it corresponds exactly to the traditional discrete Fourier transform.

3.2. Eigenproperties of the smoothing operator

The behavior of midpoint smoothing can be analyzed according to the eigenproperties of the smoothing operator. We now state some of the properties we need for later development, and leave out their proofs, since they fairly straightforward.

Let us first assume that n, the number of vertices in the polygon $\mathbf{a}$, is *odd*, and recall that $z_1, \ldots, z_n$ denote the n-th roots of unity. Then the eigenvalues of the n-th order midpoint smoothing operator S_n are

$$\lambda_j = \frac{1+z_j}{2} = \cos^2\frac{j\pi}{n} + i\sin\frac{j\pi}{n}\cos\frac{j\pi}{n} = \cos\frac{j\pi}{n}e^{i\frac{j\pi}{n}},$$

for $j = 1, \ldots, n$. For each j, an eigenvector of S_n corresponding to λ_j is

$$\mathbf{z}_j = [z_j \;\; z_j^2 \;\; \ldots \;\; z_j^n]^T = [z_1^j \;\; z_1^{2j} \;\; \ldots \;\; z_1^{nj}]^T.$$

For all $1 \le j, k \le n$, the eigenvectors $\mathbf{z}_1, \ldots, \mathbf{z}_n$ satisfy

$$\begin{aligned} \mathbf{z}_j^*\mathbf{z}_k &= 0, \quad \text{if } j \neq k \\ &= n, \quad \text{otherwise .} \end{aligned}$$

Therefore, they form a basis of the complex n-dimensional space. Furthermore, if we use Z_n to denote the matrix of eigenvectors, $Z_n = [\mathbf{z}_1 \;\; \mathbf{z}_2 \;\ldots\; \mathbf{z}_n]$, then the inverse of Z_n is given by $Z_n^{-1} = Z^*/n$. If the eigenvectors are *normalized*, then Z_n becomes *unitary*.

If n is even, then we have $\lambda_j = (1-z_j)/2$ and the corresponding eigenvector $\mathbf{z}_j = [z_j \; -z_j^2 \; z_j^3 \; \ldots \; -z_j^n]^T$, with the signs of the vector entries alternating. It turns out that the parity of n does not affect any of our results or the way our matching algorithm works. In particular, the unitarity of the matrix Z_n still holds if n is even. So in the rest of our discussions, let us assume that n is odd.

3.3. FDs with respect to our smoothing operator

Since the eigenvectors $\mathbf{z}_1, \ldots, \mathbf{z}_n$ form a basis, any polygon $\mathbf{a}$ has an *eigen-decomposition*

$$\mathbf{a} = \sum_{j=1}^{n} \mathbf{z}_j A_j \qquad (3)$$

with respect to our midpoint smoothing operator, where each A_j, an FD coefficient, is a uniquely determined 3-D vector in $\mathbf{C}^3$. To evaluate A_k, for $k = 1, 2, \ldots, n$, multiply both sides of (3) by $\mathbf{z}_k^*/n$. We have

$$A_k = \frac{1}{n}\,\mathbf{z}_k^*\mathbf{a} = \frac{1}{n}\sum_{j=1}^{n}\overline{z}_1^{\,kj}a_j = \frac{1}{n}\sum_{j=1}^{n}a_j e^{-i\frac{2\pi kj}{n}}, \qquad (4)$$

which is precisely the traditional discrete Fourier transform of $\mathbf{a}$. Note that if the Laplacian smoothing operator [16] is used instead, we would arrive at a different form for the transform.

When $k = n$, we have $A_n = \frac{1}{n}\sum_{j=1}^{n} a_j \in \mathbf{R}^3$, giving the centroid of all the vertices. For $k < n$, we can view an FD coefficient $A_k = [r_x + g_x\, i,\; r_y + g_y\, i,\; r_z + g_z\, i]$, as a pair of 3-D vectors (r_x, r_y, r_z) and (g_x, g_y, g_z). Later on, we shall reduce the problem of normalizing FDs to the problem of aligning these 3-D vectors.

3.4. Frequencies associated with FD coefficients

Consider a non-degenerate polygon $\mathbf{a}$ for which there is a scalar λ such that $(I - S_n^2)\mathbf{a} = \lambda\mathbf{a}$, where I is the identity matrix. Such a polygon is called an *eigenpolygon*, and the scalar λ is said to be its frequency. By (2), we have $\lambda\mathbf{a}_k = \mathbf{a}_k - S_n^2\mathbf{a}_k = \mathbf{a}_k - \mathbf{a}'_k$. It follows that $|\mathbf{a}_k - \mathbf{a}'_k| = |\lambda| \cdot |\mathbf{a}_k|$. The distance between $\mathbf{a}_k$ and $\mathbf{a}'_k$ is a discrete approximation of the Laplacian at $\mathbf{a}_k$, as shown in Figure 2.

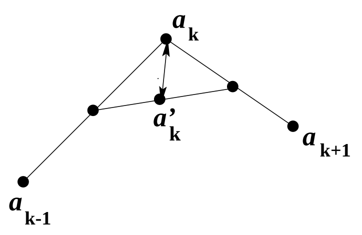


Figure 2. Discrete Laplacian at a vertex $\mathbf{a}_k$.

Geometrically, the higher the frequency λ, the larger the discrete Laplacians at the vertices, and the larger the oscillation along the polygon. If $\mathbf{a}$ is a polygon with the eigen-decomposition (3), then $(I - S_n^2)\mathbf{z}_j A_j = (1 - \lambda_j^2)\mathbf{z}_j A_j$, for each j. It follows that the real part of $\mathbf{z}_j A_j$ gives an eigenpolygon, and the larger the magnitude of λ_j, the lower its corresponding frequency. Therefore, the lowest frequency zero corresponds to the eigenvalue $\lambda_n = 1$, and the next two lowest frequencies correspond to the eigenvalues $\lambda_1 = \cos\frac{\pi}{n}e^{i\frac{\pi}{n}}$ and $\lambda_{n-1} = \cos\frac{\pi}{n}e^{-i\frac{\pi}{n}}$.

3.5. Convergence to planarity

It can be shown that even with a uniform dilation, the vertices of the midpoint polygons reach planarity. A precise statement of this is given by the following theorem. Due to a lack of space, we are not able to include the proof [22] of the theorem in this paper. For simplicity, let us assume that $A_n = 0$ in (3); that is, the centroid of the polygon vertices remains at the origin.

Theorem: *If, after each iteration of midpoint smoothing, the new polygon vertices are dilated away from the origin by a factor of* $1/\cos\frac{\pi}{n}$*, then the vertices of the midpoint polygons approach the plane spanned by the 3-D vectors*

$$\mathbf{p} = \sum_{j=1}^{n} a_j \cos\frac{2j\pi}{n} \quad \text{and} \quad \mathbf{q} = \sum_{j=1}^{n} a_j \sin\frac{2j\pi}{n}.$$

As we can see, the limit plane is completely determined by the original polygon. The key observation we make is that the directions of $\mathbf{p}$ and $\mathbf{q}$ coincide with that of the two 3-D vectors corresponding to the dominant FD coefficient

$$\begin{aligned} A_1 &= \frac{1}{n}\sum_{j=1}^{n} z_1^{-j} a_j \\ &= \left[\frac{1}{n}\sum_{j=1}^{n} a_j \cos\frac{2j\pi}{n}\right] - \left[\frac{1}{n}\sum_{j=1}^{n} a_j \sin\frac{2j\pi}{n}\right] i \\ &= \frac{\mathbf{p} - \mathbf{q}i}{n}. \end{aligned}$$

The same holds for A_{n-1}, with $A_{n-1} = (\mathbf{p}+\mathbf{q}i)/n$. Recall that A_1 and A_{n-1} are associated with the lowest non-zero frequency, and together, they quantify the dominant shape characteristic of the polygon. We shall use this fact for our FD normalization.

4. Matching 3D contours using NFDs

In this section, we first describe our notion of normalized Fourier descriptors, or NFDs, for 3-D contours. We perform normalization with respect to centroid, scale, and more crucially, the two dominant FD coefficients. Then we present our shape matching algorithm, analyze its complexity, and reason about its effectiveness.

4.1. Normalized FDs for 3-D contours

Normalization with respect to centroid and scale is straightforward [19]. This is accomplished by first translating the polygon so that the centroid of the vertices is at the origin, i.e., transform A_n to 0. Next, the coordinates of the vertices are divided by the magnitude of A_1, the dominant FD coefficient, to normalize scale.

Then, we normalize with respect to the next dominant shape characteristics, captured by A_1 and A_{n-1}. Since $A_1 = (\mathbf{p} - \mathbf{q}i)/n$ and $A_{n-1} = (\mathbf{p} + \mathbf{q}i)/n$, the normalization problem can be reduced to the alignment of the two vectors $\mathbf{p}$ and $\mathbf{q}$ to some reference frame. In the context of shape matching, we are given two contours a and b having the spanning vectors $\{\mathbf{p}_a, \mathbf{q}_a\}$ and $\{\mathbf{p}_b, \mathbf{q}_b\}$ of their respective limit plane, we need to align the two pairs of vectors *optimally* under vertex shift and rotation.

An obvious optimality criterion is to minimize the similarity distance (1). But even with the vertex orders fixed, finding the optimal rotation is a complex non-linear problem. To solve the alignment problem efficiently, we observe that although the spanning vectors are not preserved under vertex shift, the limit plane is! Regardless of the vertex ordering, midpoint smoothing produces the same sequence of midpoint polygons. Therefore, we complete the normalization by aligning both limit planes with the xy-plane.

Since the limit plane represents an “average plane” for the contour vertices, our approach also has a nice geometric interpretation. Limit plane alignment is accomplished by finding a rotation which aligns the plane normal with the z-axis exactly. This is a standard practice in vector algebra, and can be done in constant time [22].

4.2. The matching algorithm

The input to our algorithm consists of two closed 3-D polygons with the same number of vertices. In practice, if two curves to be matched do not have the same number of sample points, or the noise in the samples is too great, appropriate preprocessing steps may be applied. We shall discuss this in Section 4.2.2.

4.2.1 Overview of the algorithm

The matching algorithm can be summarized as follows, where a and b are the input polygons, each having n vertices, and k is the number of FD coefficients used for measuring similarity.

1. Compute A and B, the FDs of a and b. [$O(kn)$, or $O(n\log n)$ with Fast Fourier Transform (FFT).]

2. Centroid and scale normalization. [$O(n)$]

3. Apply rotations to align the limit planes with the xy-plane. [$O(1)$; note that the spanning vectors are available from A_1 and B_1, which are already computed.]

4. For each shift j of the vertices in b, $0 \le j \le n-1$, compute B_j, the FD of b after the shift and a rotation which aligns the major axes of a and b optimally in the xy-plane. [$O(kn)$; we will explain this step in Section 4.2.3.]

5. The similarity distance returned is the minimum of the L_2 metric $\mu(A, B_j)$, $j = 0, \ldots, n-1$, defined in (1) with $w_1 = w_2 = \ldots = w_n = 1$.

4.2.2 Preprocessing

Like all other FD-based approaches for similarity computations [12, 19, 21], the result of our matching algorithm could be sensitive to the *sampling* of the input curves. A similarity distance greater than zero may be reported for two identical curves that are only sampled differently.

To remedy this problem, we propose the use of *uniform resampling* before running the matching algorithm. We can not only ensure that the input polygons have the same number of vertices, this number can also be chosen as a power of 2, so that FFT is applicable. Making the sampling uniform should have an added advantage since midpoint smoothing does assume a uniform parameterization.

Besides uniform resampling, a polygonal curve simplification algorithm can also be applied to reduce quantization or measurement noise. Obviously, the effectiveness of the resampling or simplification algorithm should have a great influence on the overall performance of our shape matching algorithm. But discussions on choosing the appropriate sampling rate or simplification criteria are beyond the scope of this paper. We simply assume that after proper preprocessing, the input polygons have the same number of vertices, and they capture the shapes of the original input curves sufficiently well.

4.2.3 The optimal shift and rotation combination

Given two closed polygons a and b with their FDs A_0 and B_0 normalized, we show how to find the optimal shift and rotation combination for b to match its major axes with a in time $O(kn)$, where n and k are as defined in the algorithm in Section 4.2.1. Note that both major axes lie in the xy-plane after normalization.

Suppose that we fix the vertex shift j. Then it is sufficient to show that computing B_j, the FD of b after the shift and a rotation which aligns the major axes optimally in the xy-plane, takes time $O(k)$.

By (4), the m-th FD coefficient of the polygon b is $B_{0,m} = \sum_{l=1}^{n} b_l e^{-i\frac{2\pi ml}{n}}$. After a vertex shift of j, we have the following with arithmetic modulo n,

$$\begin{aligned} B_{j,m} &= \sum_{l=1}^{n} b_{l+j} e^{-i\frac{2\pi ml}{n}} = e^{i\frac{2\pi jm}{n}} \sum_{l=1}^{n} b_{l+j} e^{-i\frac{2\pi m(l+j)}{n}} \\ &= e^{i\frac{2\pi jm}{n}} \sum_{l=1}^{n} b_l e^{-i\frac{2\pi ml}{n}} = B_{0,m} e^{i\frac{2\pi jm}{n}}. \end{aligned}$$

Therefore, computing the m-th FD coefficient of a shifted polygon b is equivalent to scaling the m-th FD coefficient of the original b, which takes time $O(1)$. Since there are k FD coefficients to compute, the total time is $O(k)$.

Let us denote the shifted version of b by $b^{(j)}$. To align the major axes of a and $b^{(j)}$ optimally in the xy-plane, we first obtain the major axis $\{\mathbf{p}_b^{(j)}, \mathbf{q}_b^{(j)}\}$ for $b^{(j)}$. Since $\mathbf{p}_b^{(j)}$ and $\mathbf{q}_b^{(j)}$ are embedded in $B_{j,1}$, no extra computation is required. Similarly, the major axis $\{\mathbf{p}_a, \mathbf{q}_a\}$ of a is already computed in $A_{0,1}$. We then find the vector h_a which bisects the angle between $\mathbf{p}_a$ and $\mathbf{q}_a$ in the xy-plane. The angle bisector $h_b^{(j)}$ is defined in the same way. It should be intuitive to see that the optimal rotation angle is simply the angle between h_a and $h_b^{(j)}$, as shown in Figure 3. All these operations can be performed in $O(1)$ time.

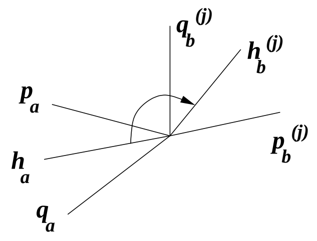


Figure 3. Aligning major axes in the xy-plane.

Let R denote the rotation matrix associated with the optimal rotation angle. By linearity of the Fourier transforms, the FDs for a rotated polygon can be obtained from a rotated version of the FDs of the original polygon. Since R is 3×3, computing the final B_j takes time $O(k)$.

4.3. Summary

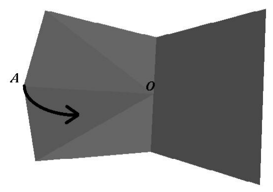
Our shape matching algorithm is invariant under translation, scale, rotation, and vertex shift; it returns zero if and only if one input polygon is a transform of the other. Our algorithm is efficient, simple to describe, and easy to implement, and the similarity distance computed is a metric.

The best time complexity for an FD-type matching algorithm is $O(n \log n)$, since computing the FDs using FFT alone takes that much time. To achieve this, it is sufficient to find an appropriate measure of the “aligned-ness” between two pairs of 3-D vectors, where the length of the spanning vectors and the subtended angles are both considered. Also, the measure should be correlated with the similarity distance between the input contours, but much easier to minimize. We are still working towards this goal.

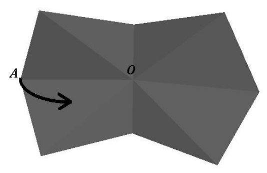
5. Experimental results

In the first experiment, let us consider the shapes shown in Figure 4. The polygons we are concerned with enclose the respective shape, and the triangle faces are added only to help with the visualization. In each case, we gradually deform the polygon by rotating the vertex A about the center O, as shown in the figure.

During each rotation, we record 100 uniformly spaced samples. We apply a random translation, vertex shift, and rotation to each sample polygon obtained, before comparing it with the original polygon using our shape matching algorithm. In Figure 5, we plot the similarity distances reported. As we can see, the output of our algorithm does match our intuition, regardless of the parity of the number of vertices in the polygons.

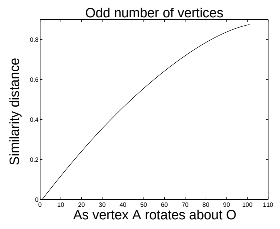


(a)

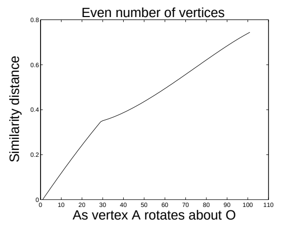


(b)

Figure 4. (a) A polygon with seven vertices. (b) A polygon with eight vertices.



(a)



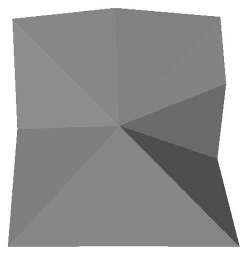
(b)

Figure 5. Plots of similarity distances.

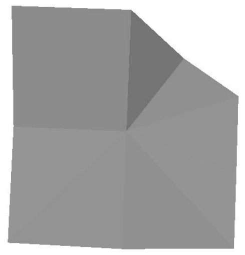
In our second experiment, we consider the test polygons shown in Figure 6. In 6(a), we have a perturbed rectangular polygon. From 6(b) to 6(f), the polygon is gradually changing into more of a triangular shape, with random perturbation applied. In 6(g), we have an arrow shape. The polygon in 6(h) is obtained from 6(g) after a significant "pulling" of two of the vertices.

Our intuition tells us from 6(b) to 6(h), the polygons are more and more different from 6(a), and the last polygon is significantly different from all the other ones. Moreover, the two pairs of polygons, 6(e) & 6(f) and 6(f) & 6(g), are closer to each other than any other pairs.

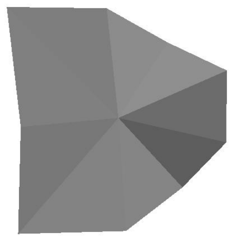
In Figure 7, we show a shading plot of the matrix of similarity distances between the eight polygons. Note that the degree of darkness indicates the degree of *similarity* between the polygons, and a similarity distance of 0 is shown in perfect black. We observe again that the output of our algorithm matches our intuition perfectly. Furthermore, these good results are obtained despite of the highly nonuniform samplings with some of the test contours, e.g., 6(f).



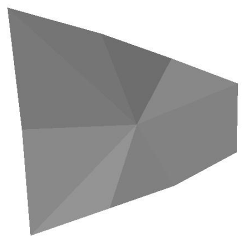
(a)



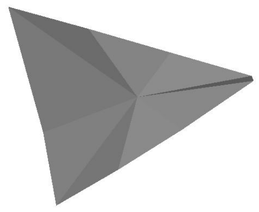
(b)



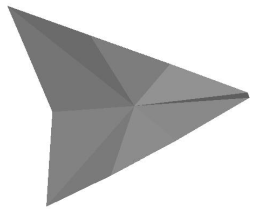
(c)



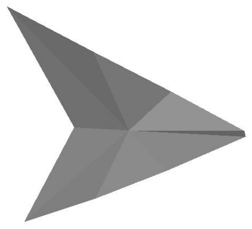
(d)



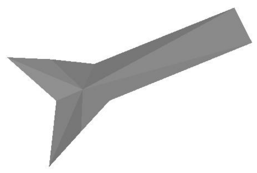
(e)



(f)



(g)



(h)

Figure 6. Test polygons for experiment 2.

6. Conclusion and future work

In this paper, we formulate a generalized notion of NFDs for arbitrary 3-D contours. We develop an $O(kn)$ shape matching algorithm, which is invariant under rigid transformations and vertex shift. The algorithm is simple, efficient, and easy to implement; it performs well in all our experiments. The similarity distance computed is a metric, and it is seen to match our intuition.

Like all other FD approaches, our matching algorithm does not guarantee that an optimal matching is found for any pair of 3-D polygons. We do believe that if two con-

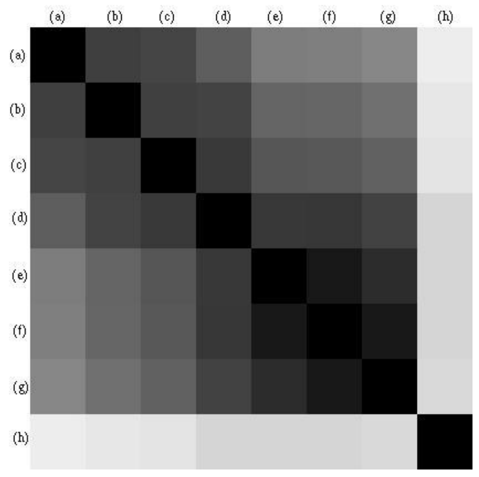


Figure 7. Shading plot of similarity distances for experiment 2.

tours are sufficiently close, the distance computed will be very close to the optimum. On the other hand, let us note that the use of geometric hashing [2, 10] is not guaranteed to produce optimal matchings either. Several factors including approximation errors, resulting from the use of the iterative closest point algorithm [10], and hash table quantization influence the performance of the matching algorithms.

Besides the possible improvement mentioned in Section 4.3, we are also looking at the possibility of extending our technique to handle open curves, e.g., when there is occlusion. Finally, many challenges still lay ahead when we wish to apply the FD techniques to the matching of large mesh models. We anticipate that the ultimate matching algorithm for meshes will need to put together an intelligent resampling technique, an efficient multiresolution method for mesh spectrum construction, and an appropriate formulation of NFDs in the irregular grid setting.

References

[1] N. Ayache, et al., "Steps Toward the Automatic Interpretation of 3-D Images," In *3D Imaging in Medicine,* by H. Fuchs et al., Springer-Verlag, pp. 107-120, 1990.

[2] A. Guéziec and N. Ayache, "Smoothing and Matching of 3-D Spatial Curves," *Int. J. Compu. Vis.,* Vol. 12, No. 1, 79-104, 1994.

[3] P. Heckbert and M. Garland, "Survey of Polygonal Surface Simplification Algorithms," *SIGGRAPH 97 Course Notes*.

[4] M. Hilaga et al., "Topology Matching for Fully Automatic Similarity Estimation of 3D Shapes," *SIGGRAPH 2001,* pp. 250-259.

[5] E. Kishon, E. Hastie, and H. Wolfson, "3-D Curve Matching using Splines," *Proc. 1st Europ. Conf. Compu. Vis.,* Antibes, France, 1990.

[6] S. Loncaric, "A Survey of Shape Analysis Techniques," *Pattern Recognition,* Vol. 31, No. 8, pp. 983-1001, 1998.

[7] D. Mumford, "Mathematical Theories of Shape: Do They Model Perception?," *SPIE Vol. 1570 Geometric Methods in Computer Vision,* pp. 2-10, 1991.

[8] R. Osada, et al., "Matching 3D Models with Shape Distributions," *SMI 2001,* Genova, Italy.

[9] P. J. van Otterloo, *A Contour-Based Approach to Shape Analysis,* Prentice Hall, 1991.

[10] T. Pajdla and L. Van Gool, "Efficient Matching of Space Curves," In *6-th Int. Conf. Computer Analysis of Images and Patterns,* pp. 25-32, September. 1995.

[11] E. Persoon and K. S. Fu, "Shape Discrimination using Fourier Descriptors," *IEEE Trans. Sys. Man, Cyb.*, 1977.

[12] Y. Rui, A. She, and T. S. Huang, "A Modified Fourier Descriptor for Shape Matching in MARS," *Image Databases and Multimedia Search, Series on Software Engineering and Knowledge Engineering,* Vol. 8, pp .165-180, 1998

[13] J. Schwartz and M. Sharir, "Identification of Partially Obscured Objects in Two and Three Dimensions by Matching Noisy Charateristics Curves," *The International Journal on Robotics Research,* Vol. 6, No. 2, pp. 29-44, 1987.

[14] L. G. Shapiro, "A Structural Model of Shape," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* pp. 111-126, 1980.

[15] H. Y. Shum, M. Hebert, and K. Ikeuchi, *On 3D Shape Similarity,* CMU Technical Report 95-212, School of Computer Science, Carnegie Mellon University, 1995.

[16] G. Taubin, "A Signal Processing Approach to Fair Surface Design," *Computer Graphics Proceedings, SIGGRAPH 95,* pp. 351-358, 1995.

[17] M. R. Teague, "Image Analysis via the General Theory of Moments," *. J. Opt. Soc. Amer.,* Vol. 70, pp. 920-930, 1980.

[18] R. C. Veltkamp, "Shape Matching: Similarity Measures and Algorithms," *Proc. Shape Modelling International,* Genova, Italy, pp. 188-197, May 2001.

[19] T. P. Wallace and P. A. Wintz, "An Efficient 3-D Aircraft Recognition Algorithm Using Normalized Fourier Descriptors," *Computer Graphics and Image Processing,* Vol. 13, pp. 99-126, 1980.

[20] H. J. Wolfson and I. Rigoutsos, "Geometric Hashing: An Overview," *IEEE Computational Science and Engineering,* Vol. 4, No. 4, pp. 10-21, October, 1997.

[21] C. T. Zahn and R. Z. Roskies, "Fourier Descriptors for Plane Closed Curves," *IEEE Transactions on Computers,* Vol. 21, No. 3, pp. 269-281, March 1972.

[22] H. Zhang, *Eigenvalue Decomposition of Polygonal Shapes and Applications,* Ph.D. Thesis, Department of Computer Science, University of Toronto, expected June 2002.