Best-Fit of Sculptured Surfaces

Walter Gander (gander@inf.ethz.ch)
ETH Zürich, Switzerland

David Sourlier (delta-3d@bluewin.ch)
Delta 3D GmbH, Hinwil, Switzerland

Abstract: The parametric representation of surfaces is often avoided in best-fit computations because of the large number of unknowns. By exploiting the matrix structure, D. Sourlier has developed an effective best-fit software 'FUNKE', which is based on parametric representation. FUNKE is perfectly applicable for sculptured surfaces. We report on the best-fit of a turbine blade.

In the second part we propose a new *shape distance measure* to compare two CAD-described surfaces. We give algorithms in Matlab to compute this shape distance and the corresponding transformation.

1. Introduction and Notations

In this paper we will represent a patch of a sculptured surface (i.e. a polynomial surface) as a tensor product surface

$$\mathbf{x}(u,v)=\begin{pmatrix}x(u,v)\\y(u,v)\\z(u,v)\end{pmatrix}=\begin{pmatrix}\mathbf{f}^T(u)A\,\mathbf{g}(v)\\\mathbf{f}^T(u)B\,\mathbf{g}(v)\\\mathbf{f}^T(u)C\,\mathbf{g}(v)\end{pmatrix},\quad\begin{array}{l}0\le u\le 1\\0\le v\le 1\end{array}\tag{1}$$

$\mathbf{f}(u)=[f_1(u),\dots,f_n(u)]^T$ and $\mathbf{g}(v)=[g_1(v),\dots,g_m(v)]^T$ are given basis functions of the surface parameters u and v. These basis functions could be NURBS, B-splines, Bezier-polynomials or simply monomes i.e. $f_i(u)=u^{i-1}$. The coefficients specifying the surface are given by the three matrices A, B, $C\in\mathbb{R}^{m\times n}$. It will be convenient to write the surface (1) sometimes as

$$\begin{pmatrix}x(u,v)\\y(u,v)\\z(u,v)\end{pmatrix}=\begin{pmatrix}\sum_{i=1}^n\sum_{j=1}^m a_{ij}f_i(u)g_j(v)\\\sum_{i=1}^n\sum_{j=1}^m b_{ij}f_i(u)g_j(v)\\\sum_{i=1}^n\sum_{j=1}^m c_{ij}f_i(u)g_j(v)\end{pmatrix}=\begin{pmatrix}\sum_{k=1}^{mn}\alpha_kF_k(u,v)\\\sum_{k=1}^{mn}\beta_kF_k(u,v)\\\sum_{k=1}^{mn}\gamma_kF_k(u,v)\end{pmatrix}.\tag{2}$$

The coefficients α_k, β_k and γ_k and the basis functions $F_k(u,v)$ are obtained by rearranging the data as follows: $\alpha_{i+n(j-1)}=a_{ij}$, $\beta_{i+n(j-1)}=b_{ij}$, $\gamma_{i+n(j-1)}=c_{ij}$ and $F_{i+n(j-1)}(u,v)=f_i(u)g_j(v)$.

2. Least Squares Fit of sculptured surface

A problem in coordinate metrology is the best-fit of a geometric element into measuring points. Best-fit means that the sum of squares of the geometric distances of the measuring points to the geometric element with optimally fitted parameters is to be minimized.

Although the parametric surface representation is a standard in all fields of CAD/ CAM/CAQ (cf. B-splines/NURBS) as well as in related fields like e.g. surface reconstruction, this representation is not widely introduced and used so far in the commercial software for the practical needs of coordinate measuring techniques. The best-fit calculation is still based there on implicit surface representation $f(\mathbf{x})=0$. The main reason for that is the advantage of not having the two surface coordinates u and

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35392-0_40

v involved as additional unknowns in the least squares fit. The problem with implicit surface representation is, however, that only for standard surfaces like plane, sphere, cylinder, cone an explicit distance function is known. For more complex surfaces like ellipsoids, with no explicit expression for the distance, an approximation for the geometric distance is often used. Of course by using such an approximation the computed fit is not the exact best-fit and this may not be acceptable.

With the parametric approach the number of unknowns is dramatically enlarged: each measuring point generates an additional unknown when fitting a 3D-curve, or even two unknowns if we wish to fit a surface (cf. (Gander, Golub and Strebel, 1994) or (Sourlier, 1995)). However, Sourlier has developed a software FUNKE (Sourlier, 1995) based on the parametric representation which allows to compute best-fits of surfaces. He shows that by exploiting the structure of the Jacobian matrix even with the additional unknowns the computational complexity is comparable to a fit using an implicit surface representation. Interesting features of FUNKE are

- best-fit of sculptured surfaces, complex features (e. g. involute, helical toroid), combined features (e. g. quadrant to be fitted by position/orientation, length, width and height),
- best-fit with frozen degrees of freedom (e. g. fix-radius cylinder fit),
- best-fit available for any new defined surface, standardization of best-fit-implementation,
- function-independent best-fit procedure by separating geometry from position/orientation description),
- improved possibilities for probe radius correction and deflection compensation.

The software FUNKE computes the best-fit (in the 2-norm) to given measuring points for every parameterized surface function. FUNKE is based on a generic surface function $\mathbf{x}(u,v)$ which is replaced in each function call by the current surface that we wish to fit. There is a strict separation of *geometric* and *position/orientation* parameters. The surface is thus described in its simplest position. We call this the *special parametric description* in which only the parameters $\mathbf{p}$ occur which describe the *geometry* of the surface. Here are some examples:

$$\begin{aligned}
\mathbf{x}'_{\text{plane}}(u,v,\mathbf{p}) &= (u,v,0)\\
\mathbf{x}'_{\text{sphere}}(u,v,\mathbf{p}) &= (p_1\cos u\cos v, p_1\sin u\cos v, \sin v)\\
\mathbf{x}'_{\text{cylinder}}(u,v,\mathbf{p}) &= (p_1\cos u, p_1\sin u, v)\\
\mathbf{x}'\text{screw}(u,v,\mathbf{p}) &= (v\cos u, v\sin u, p_1 u)
\end{aligned}$$

The *general description* of a surface which includes the position/ orientation information

$$\mathbf{t} = (t_x, t_y, t_z) \quad \text{and the angles} \quad \mathbf{a} = (a,b,c)$$

is given by the transformation ($R(\mathbf{a})$ is an orthogonal matrix):

$$\mathbf{x}(u,v,\mathbf{p},\mathbf{a},\mathbf{t}) = R(\mathbf{a})\mathbf{x}'(u,v,\mathbf{p}) + \bar{\mathbf{t}} = R(\mathbf{a})(\mathbf{x}'(u,v,\mathbf{p}) + \mathbf{t}).$$

If we wish to compute the distance $d_{\hat{\mathbf{x}}}$ of a measuring point $\hat{\mathbf{x}}$ to the surface, we have to solve a minimization problem in order to find the foot-point coordinates

$$d_{\hat{\mathbf{x}}} = \min_{u,v} ||\hat{\mathbf{x}} - \mathbf{x}(u,v,\mathbf{p},\mathbf{a},\mathbf{t})||.$$

To compute the best-fit of a surface to n measuring points we need to minimize the sum of squares of the distances and thus we have to introduce $2n$ more unknowns for the foot-point coordinates $\mathbf{U} = (u_1, v_1, \ldots, u_n, v_n)$:

$$Q_2(\mathbf{U},\mathbf{p},\mathbf{a},\mathbf{t}) = \sum_{i=1}^{n} ||\hat{\mathbf{x}}_i - \mathbf{x}(u_i,v_i,\mathbf{p},\mathbf{a},\mathbf{t})||^2 = \min$$

At first it seems that these new unknowns would make the problem much more difficult to solve. However, as shown in (Sourlier, 1995) by exploiting the structure of the Jacobian, the complexity grows only linearly with the number of measuring points.

Because we separate geometry from position/orientation it is possible to fit a set of N geometrically different surfaces with common position/orientation as a *compound object.* For the implementation we have to define a relation

$$i \mapsto K(i); \quad 1 \leq i \leq n \quad \text{and} \quad A \leq K \leq N$$

which assigns the i-th point to a specific surface K. In this way we obtain for each pair (u_i, v_i) of surface coordinates the values of the corresponding surface point from the generic function

$$\mathbf{x}_{(i)} = \mathbf{x}'^{(K(i))}(u_i, v_i, \mathbf{p}).$$

If, for example, we have N different surfaces of the same type (e.g. planes) but in different positions/orientations, we can start from the same nominal surface function, e.g. $\mathbf{x}'(u,v) = (u,v,0)$, and construct the N surface functions by attributing appropriate values to their position/orientation offsets:

$$\mathbf{x}'^{(A)}(u,v,\mathbf{p}) = R(\mathbf{a}^{(A)}) \cdot (\mathbf{x}'(u,v,\mathbf{p}) + \mathbf{t}^{(A)})$$

$$\vdots$$

$$\mathbf{x}'^{(N)}(u,v,\mathbf{p}) = R(\mathbf{a}^{(N)}) \cdot (\mathbf{x}'(u,v,\mathbf{p}) + \mathbf{t}^{(N)}) \ .$$

The position/orientation offsets $\mathbf{a}^{(A)}, \mathbf{t}^{(A)}, \ldots, \mathbf{a}^{(N)}, \mathbf{t}^{(N)}$ which define the (known and fixed) relative position/orientations of the N surfaces are stored *individually* and are *not* changed during the best-fit procedure, in contrast to the position/orientation parameters $\mathbf{a}$ and $\mathbf{t}$ which are *common* for all actually fitted surfaces and which are subject to *best-fit.*

With this technique it is possible to best-fit a set of surfaces patches as a whole. We can even mix standard surfaces with sculptured surfaces. Sculptured surfaces are a special case. It is important to maintain the C_0 or C_1 continuity of the patches as defined by the CAD dates. Therefore a best-fit respecting the given relative position/orientation is very important. For this purpose FUNKE is able to read and process CAD data of a sculptured surface.

Imagine a turbine blade as shown in Figure 1. What can we do to completely separate form deviations from position/orientation deviations? It is not enough to compare the CAD-defined surface against the measuring points in a coordinate system e.g. defined by some reference points or determined by the turbine base (on which we are able to measure and best-fit standard surfaces). Doing this we would get a larger form deviation than effectively possible (with the theoretically optimal alignment)! In order to obtain the pure form deviation (free from any position/orientation deviation) we have to determine the "intrinsic" coordinate system of the sculptured surface of the turbine blade itself. In other words: form deviations can be shown in that workpiece coordinate system where the sum of squares of all surface scan points attains its minimum.

To show this we perform the evaluation in two ways. Figure 1 shows the form deviations computed by FUNKE in the theoretical optimal way: We obtain here for a sculptured surface the same good results as we would also obtain (with commercial software) for standard surfaces. The mathematical description (in polynomial form) is given in the "VDA-FS"-Format (well-known in Europa), but as well we could use also other sculptured surface formats, like IGES or STEP.

For 981 measuring points the computation took about 0.3 seconds on a PC equipped with a Pentium processor. The average deviation for this best-fit is 3.8μ. After 5 steps of Gauss-Newton iterations we obtained convergence. The average

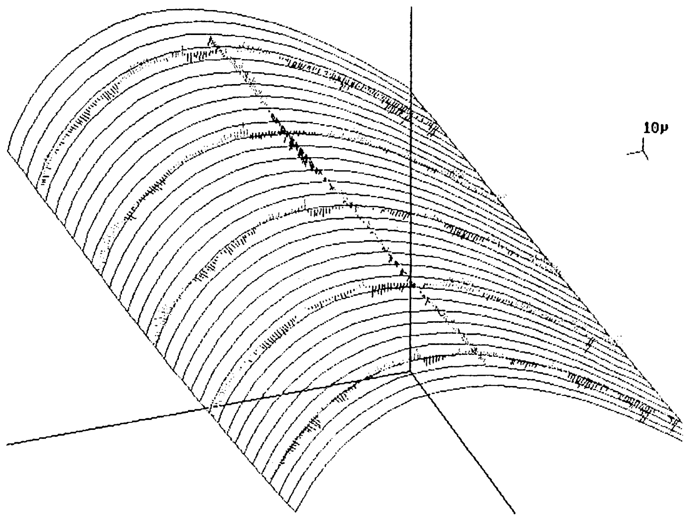

Figure 1. Turbine blade: pure form deviation $\sigma = 3.83\mu$

deviation declined as

$$2363.62912\mu \to 205.70313\mu \to 5.65472\mu \to 3.8342\mu \to 3.83208\mu \to 3.83208\mu$$

Figure 2 shows the same deviations in a coordinate system defined by the turbine base. This coordinate system – which can easily be determined also by commercial CMM software – delivers too large form deviations due to the relative position/orientation deviation between turbine base and blade. This can clearly be seen as systematic deviation in Figure 2 and 3.

3. A Shape Distance Measure for Sculptured Surfaces

3.1. Norm of a Surface

In the preceding chapter we have discussed a best-fit method for sculptured surfaces. The best-fit problem is defined as the fit of a point cloud (measuring points) against a given surface (CAD-description). But what about comparing two surfaces (i.e. two CAD-descriptions) to each other? This is what we are concerned in this chapter: we first propose a *shape distance measure* D which tells how close two surfaces described parametrically by polynomials (monomials, Bezier-polynomials or B-splines) are, with respect of their geometrical shape. Then we will show a method which minimizes this shape measure D as function of the relative position and orientation of these two surfaces. This allows us to compare two slightly different geometries, regarding their "similarity of shape" independently of their actual position/orientation and thus their description with rather different coefficients.

We will assume in the following that the parameterization (u, v) of both surfaces is compatible and comparable (e.g. using a chordal parameterization), so that in some sense points with the same u, v-coordinates correspond to each other. We will call such surfaces *similarly parameterized.* This assumption may be questionable from a mathematical viewpoint. However, we think that for practical purposes the shape

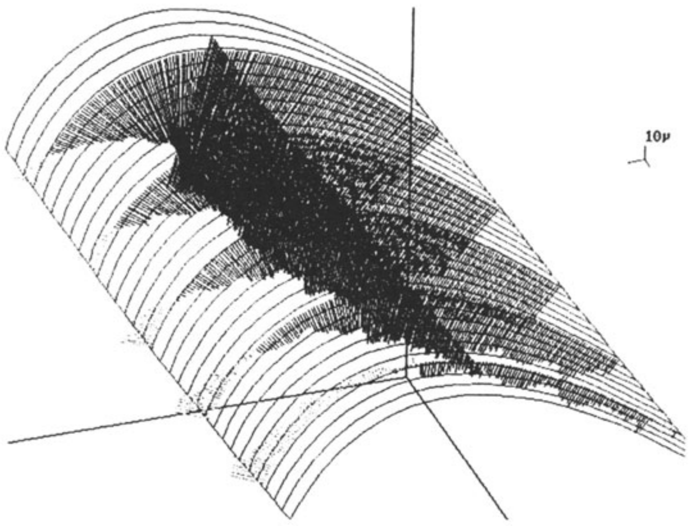

Figure 2. Turbine blade: deviation evaluated in the coordinate system defined by the turbine base (combined form and position/orientation deviations $\sigma = 56\mu$)

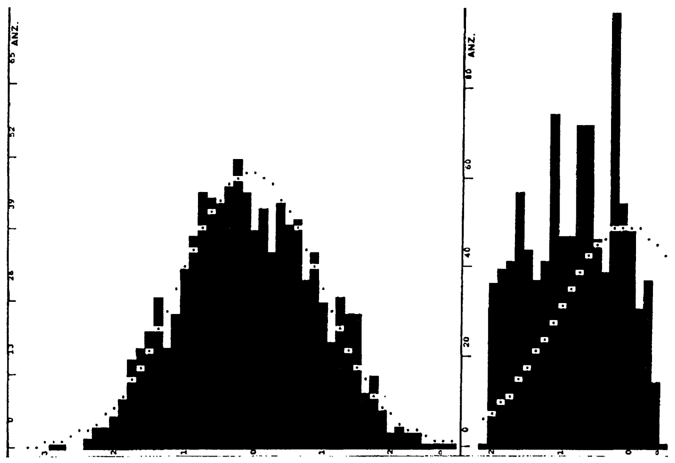

Figure 3. Left: distribution of form deviation from Figure 1. Right: same for Figure 2. The theoretical Gaussian distribution is marked by the dots.

distance measure based on this assumption and introduced in the following may be useful anyway. We will show examples where this measure proves to be valuable.

DEFINITION 1. *The* 2-norm of a surface *is*

$$||\mathbf{x}|| := \sqrt{\int_0^1 \int_0^1 ||\mathbf{x}(u,v)||_2^2 \, dudv}, \tag{3}$$

where $||\mathbf{x}(u,v)||_2^2 = x(u,v)^2 + y(u,v)^2 + z(u,v)^2$.

Using the representation (1) we obtain the expression

$$||\mathbf{x}||^2 = \int_0^1 \int_0^1 (\mathbf{f}^T(u)A\,\mathbf{g}(v))^2 + (\mathbf{f}^T(u)B\,\mathbf{g}(v))^2 + (\mathbf{f}^T(u)C\,\mathbf{g}(v))^2\,dudv. \quad (4)$$

Considering only the first term in (4) we obtain

$$\begin{aligned}\int_0^1 \int_0^1 (\mathbf{f}^T A\,\mathbf{g})^2\,dudv &= \int_0^1 \int_0^1 \left(\sum_{i,j} a_{ij} f_i g_j\right)\left(\sum_{k,l} a_{kl} f_k g_l\right) dudv \quad (5)\\ &= \sum_{i,j,k,l} a_{ij}a_{kl} \int_0^1 f_i(u)f_k(u)\,du \int_0^1 g_j(v)g_l(v)\,dv.\end{aligned}$$

Now, *if the basis functions are orthogonal* i.e. if

$$\int_0^1 f_i(u)f_k(u)\,du = \int_0^1 g_i(u)g_k(u)\,du = \begin{cases} 1, & i=k \\ 0, & i\neq k\end{cases}$$

then (5) simplifies to

$$\int_0^1 \int_0^1 (\mathbf{f}^T A\,\mathbf{g})^2\,dudv = \sum_{i,j} a_{ij}^2 = ||A||_F^2.$$

The same simplifications hold for the other terms. Thus we obtain the theorem:

THEOREM 1. *Let* $\mathbf{x}(u,v)$ *be defined by (1) with orthonormal basis functions* $f_i(u)$ *and* $g_j(v)$. *Then the 2-norm can be computed by the expression:*

$$||\mathbf{x}||^2 = \int_0^1 \int_0^1 ||\mathbf{x}(u,v)||_2^2\,dudv = ||A||_F^2 + ||B||_F^2 + ||C||_F^2, \quad (6)$$

where $||\ ||_F$ *denotes the Frobenius norm.*

3.2. Mean Square Distance of Surfaces

DEFINITION 2. *The* mean square distance D *of two surfaces* $\mathbf{x}(u,v)$ *and* $\mathbf{x}'(u,v)$ *is given by*

$$D^2 = \int_0^1 \int_0^1 ||\mathbf{x}(u,v) - \mathbf{x}'(u,v)||_2^2\,dudv. \quad (7)$$

If we perform a change of basis and represent the two surfaces by the same set of orthogonal basis functions then by Theorem 1 the mean square distance (*the shape distance*) can be computed by

$$D^2 = ||\Delta A||_F^2 + ||\Delta B||_F^2 + ||\Delta C||_F^2, \quad (8)$$

where $\Delta A = A - A'$ etc.

Let us consider now the following problem: Given two similarly parameterized surfaces $\mathbf{x}(u,v)$ and $\mathbf{x}'(u,v)$, find an orthogonal matrix R (a product of three rotations) and a translation vector $\mathbf{t}$ such that the shape distance between $\mathbf{x}(u,v)$ and $R\mathbf{x}'(u,v) + \mathbf{t}$ is minimized:

$$D^2 = \int_0^1 \int_0^1 ||R\mathbf{x}'(u,v) + \mathbf{t} - \mathbf{x}(u,v))||_2^2\,dudv = \min$$

Using the representation (2)

$$\mathbf{x}(u,v) = \begin{pmatrix} \sum_{k=1}^{mn} \alpha_k F_k(u,v) \\ \sum_{k=1}^{mn} \beta_k F_k(u,v) \\ \sum_{k=1}^{mn} \gamma_k F_k(u,v) \end{pmatrix} = \Gamma \mathbf{F},$$

where Γ denotes the matrix

$$\Gamma = \begin{bmatrix} \alpha_1 & \dots & \alpha_{nm} \\ \beta_1 & \dots & \beta_{nm} \\ \gamma_1 & \dots & \gamma_{nm} \end{bmatrix}$$

and

$$\mathbf{F} = [F_1(u,v), \dots, F_{mn}(u,v)]^T$$

the problem becomes

$$D^2 = \int_0^1 \int_0^1 ||(R\Gamma' - \Gamma)\mathbf{F} + \mathbf{t}||_2^2 \, dudv = \min$$

The problem simplifies if we make the assumption that $F_1(u,v) \equiv 1$. This assumption is not very restrictive since it is very often the case that the constant 1 belongs to the set of orthogonal functions. If $F_1(u,v) \equiv 1$ the translation vector $\mathbf{t}$ can be subtracted from the first column of Γ, giving the matrix Γ^*. Now the problem becomes

$$D^2 = \int_0^1 \int_0^1 ||(R\Gamma' - \Gamma^*)\mathbf{F}||_2^2 \, dudv = ||R\Gamma' - \Gamma^*||_F^2 = \min . \tag{9}$$

We have now reduced the problem to compute the shape distance of two similarly parameterized surfaces to a standard problem, namely to an *orthogonal Procrustes Problem* (Golub and Van Loan, 1996): find an orthogonal matrix R such that

$$Q(R,\mathbf{t}) = ||R\Gamma' - \Gamma^*||_F^2 = ||\Gamma^{*T} - \Gamma'^T R^T||_F^2 = \min . \tag{10}$$

A problem of type (10) also occurs when computing a least squares fit of two point clouds; cf. (Hanson and Norris, 1981) and (Gander, 1997).

By minimizing Q with respect to $\mathbf{t}$ we obtain the necessary equation (note that $A(:,1)$ indicates the first column of the matrix A):

$$\mathbf{t} = \Gamma(:,1) - R\,\Gamma'(:,1) \tag{11}$$

which means, that after computing R, we can always choose $\mathbf{t}$ such that the first column of the matrix $R\Gamma' - \Gamma^*$ is zero. Thus we first determine R such that

$$||R\Gamma'(:,2:mn) - \Gamma(:,2:mn)||_F^2 = \min \tag{12}$$

and then compute $\mathbf{t}$ using(11).

3.3. Orthogonal Basis Functions

The surface $\mathbf{x}(u,v)$ represented by (1) makes use of the basis functions $\mathbf{f}^T(u) = [f_1(u), \dots, [f_n(u)]$ and $\mathbf{g}^T(v) = [g_1(v), \dots, [g_m(v)]$.

Surfaces are often given in the monomial basis i.e. $f_i(t) = g_i(t) = t^{i-1}$. For our purpose we need to transform this basis in an orthogonal basis. For the interval $(0,1)$, the orthogonal polynomials are the scaled and normalized *Legendre polynomials*:

$$p_0(t) = 1 \tag{13}$$

$$p_1(t) = \sqrt{3} - 2\sqrt{3}t \tag{14}$$

$$p_2(t) = \sqrt{5} - 6\sqrt{5}t + 6\sqrt{5}t^2 \tag{15}$$

If we denote the monomial basis functions by $\mathbf{m}^T(t) = [1, t, \ldots, t^{n-1}]$ and with $\mathbf{p}$ the scaled and normalized Legendre polynomials then there exists a linear transformation $\mathbf{p} = T\mathbf{m}$. The lower triangular transformation matrix T is

$$T = \begin{pmatrix} 1 & 0 & 0 & \cdots \\ \sqrt{3} & -2\sqrt{3} & 0 & \cdots \\ \sqrt{5} & -6\sqrt{5} & 6\sqrt{5} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}.$$

T can be generated by the following Matlab function:

```
function T = ng (n)
%  p = T m
i = 1:n-1;
F = diag(i./(4*i-2),-1) + 1/2*eye(n) + diag(i./(4*i+2),1);
T = eye(n);
for k = 2:n,
  T(:,k) = F*T(:,k-1);
end;
for i = 1:n,
  T(i,:) = T(i,:)/sqrt(2*i-1);
end;
T = T';
```

The matrix $U = T^{-1}$ of the inverse transformation $\mathbf{m} = U\mathbf{p}$ is also easy to compute:

```
function U = gn (n)
%  m = U p
U = eye(n);
for k = 2:n,
  U(:,k) = (2-1/(k-1))*(2*[0; U(1:end-1,k-1)] - U(:,k-1));
  if (k>2), U(:,k) = U(:,k) - (1-1/(k-1))*U(:,k-2); end;
end;
for k = 1:n,
  U(:,k) = U(:,k)*sqrt(2*k-1);
end;
 U =U';
```

Thus if a surface is given in the monomial basis

$$\mathbf{x}(u,v) = \begin{pmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{pmatrix} = \begin{pmatrix} \mathbf{m}^T(u)A\,\mathbf{m}(v) \\ \mathbf{m}^T(u)B\,\mathbf{m}(v) \\ \mathbf{m}^T(u)C\,\mathbf{m}(v) \end{pmatrix} \tag{16}$$

then using $\mathbf{m} = U\mathbf{p}$ we obtain the representation in the orthogonal basis

$$\mathbf{x}(u,v) = \begin{pmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{pmatrix} = \begin{pmatrix} \mathbf{p}^T(u)\tilde{A}\,\mathbf{p}(v) \\ \mathbf{p}^T(u)\tilde{B}\,\mathbf{p}(v) \\ \mathbf{p}^T(u)\tilde{C}\,\mathbf{p}(v) \end{pmatrix} \tag{17}$$

with $\tilde{A} = U^T A U$, $\tilde{B} = U^T B U$ and $\tilde{C} = U^T C U$.

3.4. Shape Distance Algorithm

The following Matlab program computes the minimal mean square distance (shape distance) as function of R and $\mathbf{t}$ between two surfaces $\mathbf{x}$ and $R\mathbf{x}' + \mathbf{t}$ that are given according to (1) by the three matrices A,B and C respectively A', B' and C'. We assume that the monomes $\mathbf{m}$ are used as basis functions and that all matrices are $n \times m$.

```
function [D, R, t] = distance(A,B,C,Ap,Bp,Cp);
% [D, R, t] = distance(A,B,C,Ap,Bp,Cp) computes an
% orthogonal matrix R and a translation vector t,
% such that the mean square distance D between the
% sculptured surfaces x(u,v) and Rx'(u,v)+t is
% minimized.
% The basis functions used for representing x and
% x' are supposed to be the monomes.

 [n,m] = size(A);

% transform  coefficients for orthogonal basis functions
 Um = ng(m); Un = ng(n);
 At = Un'*A*Um;    Apt = Un'*Ap*Um;
 Bt = Un'*B*Um;    Bpt = Un'*Bp*Um;
 Ct = Un'*C*Um;    Cpt = Un'*Cp*Um;

% compute Gamma Matrices
 Gamma  = reshape([At,Bt, Ct],[m*n,3]')'
 Gammap = reshape([Apt,Bpt,Cpt],[m*n,3]')';

% solve Procrustes problem: find R such that
% norm(R*Gammap(:,2:m*n)-Gamma(:,2:m*n),'fro') = min
 [u s v] = svd(Gammap(:,2:m*n)*Gamma(:,2:m*n)');
 R = v*u';

% compute the translation vector t
t = Gamma(:,1) - R*Gammap(:,1);

% compute shape distance
D = sqrt(norm(R*Gammap(:,2:m*n)-Gamma(:,2:m*n),'fro'));
```

If we wish to decompose the orthogonal matrix R into the product of three rotations $R = R_3R_2R_1$ where

$$R_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_1 & s_1 \\ 0 & -s_1 & c_1 \end{pmatrix},\ R_2 = \begin{pmatrix} c_2 & 0 & s_2 \\ 0 & 1 & 0 \\ -s_2 & 0 & c_2 \end{pmatrix} \text{and } R_3 = \begin{pmatrix} c_3 & s_3 & 0 \\ -s_3 & c_3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad (18)$$

are plane rotation matrices specified by $c_k = \cos\theta_k$ and $s_k = \sin\theta_k$, $k = 1,2,3$, defining rotations about the $x-$, $y-$ and $z-$axes, respectively, we can use the Matlab function

```
function [theta] = rotangle(H)
%
if det(H)<0,
  error('The matrix is not a product of rotations')
end
n = size(H,1);
theta = [];
for i = 1 : n - 1
  for j = i + 1 : n
      theta_k = atan2(-H(j,i),H(i,i));
      theta = [theta_k, theta];
      c = cos(theta_k); s = sin(theta_k);
      R = eye(n);
      R(i,i) = c; R(j,j) = c;
      R(i,j) = -s; R(j,i) = s;
      H = R * H;
  end
end
```

which is described in (Gander, 1997).

We have presented the method for a "one-patch" surface. But the same method method works just as well when comparing "multi-patch" surfaces.

In the case of a Bezier or a B-spline polynomial description we could alternatively apply the algorithm above directly to the two sets of the $m \times n$ Bezier-points (or De Boor points respectively) belonging to the two surfaces and fit them against each other. Doing so we would likely get even then also a useful result. Of course we wouldn't achieve the theoretically best result in the sense of minimizing the measure D. This we can only achieve by changing to the proposed orthogonal base.

From a practical point of view, of course it would also be useful to minimize the distance of the two surface with respect to a "Chebychev-criterion" i.e. minimizing their maximal difference in shape (regarding all possible u, v surface coordinates). But because we are dealing with parametric polynomials, where we have a vector instead of a scalar function, the explicit expression (8) which considers the three different coordinate-directions x, y, z individually, unfortunately works only for the 2-norm.

3.5. Examples

3.5.1. *First example*

We consider the surface $\mathbf{x}$ given by the following three matrices:

$$A = \begin{pmatrix} 20.00 & 65.55 & 0 & 76.08 & -76.08 \\ 65.55 & 0 & -45194.21 & 89475.40 & -44281.19 \\ 0 & -34694.54 & 375248.86 & -654859.47 & 314305.16 \\ 0 & 69389.08 & -638425.22 & 1088769.50 & -519733.37 \\ 0 & -34694.54 & 308199.37 & -523119.13 & 249614.29 \end{pmatrix}$$

$$B = \begin{pmatrix} 30.00 & 118.78 & 0 & 9.46 & -9.46 \\ -273.19 & 0 & -5619.51 & 11125.49 & -5505.98 \\ 0 & -4313.97 & 46658.96 & -81426.13 & 39081.14 \\ 0 & 8627.93 & -79382.68 & 135379.11 & -64624.37 \\ 0 & -4313.97 & 38321.94 & -65045.36 & 31037.39 \end{pmatrix}$$

$$C = \begin{pmatrix} 40.00 & 267.57 & 0 & -22.84 & 22.84 \\ 105.21 & 0 & 13566.70 & -26859.32 & 13292.62 \\ 0 & 10414.84 & -112644.70 & 196580.08 & -94350.21 \\ 0 & -20829.68 & 191646.73 & -326834.08 & 156017.02 \\ 0 & 10414.84 & -92517.34 & 157033.38 & -74930.88 \end{pmatrix}$$

Surface **x** is represented in Figure 4 using the commands

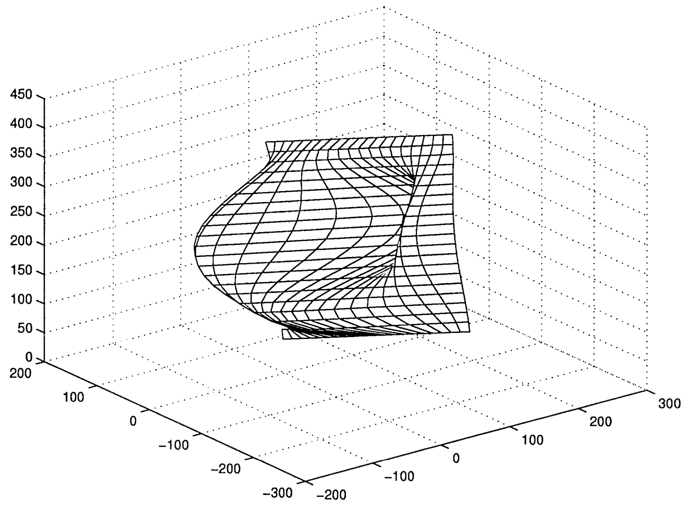

Figure 4. Surface x

```
[X,Y,Z] = evalpoints(A,B,C,0.05)
 mesh(X,Y,Z)
```

where `evalpoints` is defined as

```
function [X,Y,Z] = evalpoints(A,B,C,delta)
% EVALPOINTS(A,B,C,delta) evaluates for parameter values
% u,v in the range  0:delta:1 the surface coordinates
% X(u,v), Y(u,v), Z(u,v) that are defined as quadratic
% forms in the monomial basis i.e.  X(u,v) = U*A*V',
% Y(u,v) = U*B*V' and Z(u,v)=  U*C*V' where
% U = [1 u ... u^(n-1)] and V = [1, v, ..., v^(m-1)].

[n, m] = size(A);
disk = 0:delta:1;
X=[]; Y=[]; Z=[];
i=0;
for u = disk
  i=i+1;
  j=0;
  for v = disk
   j=j+1;
   U=[];
   for k = 0:1:n-1
     U=[U u^k];
   end
   V=[];
   for k = 0:1:m-1
     V=[V v^k];
   end
   X(j,i)= U*A*V';
   Y(j,i)= U*B*V';
   Z(j,i)= U*C*V';
  end
end
```

The second surface $\mathbf{x}'$ is obtained by slightly modifying the geometry of surface $\mathbf{x}$.

$$A1 = \begin{pmatrix} 14.64 & 82.48 & 8.17 & 11.25 & -27.88 \\ 67.69 & 41.12 & -45599.89 & 90232.25 & -44691.13 \\ 54.33 & -35102.16 & 376766.79 & -656919.19 & 315196.27 \\ -94.96 & 70074.02 & -640445.19 & 1091043.92 & -520521.84 \\ 46.24 & -35060.86 & 309170.71 & -524055.07 & 249856.09 \end{pmatrix}$$

$$B1 = \begin{pmatrix} 30.06 & 82.76 & 62.26 & -13.31 & -15.52 \\ -318.31 & 391.45 & -6265.88 & 11458.94 & -5548.97 \\ 155.04 & -5269.93 & 48209.97 & -82303.03 & 39263.89 \\ -189.24 & 9597.59 & -80850.86 & 136143.29 & -64756.29 \\ 83.75 & -4717.12 & 38869.78 & -65247.23 & 31018.27 \end{pmatrix}$$

$$C1 = \begin{pmatrix} 36.01 & 299.19 & -75.52 & 56.40 & -10.76 \\ 163.26 & -305.86 & 14116.39 & -27134.67 & 13314.67 \\ -140.68 & 11276.88 & -114341.68 & 197596.04 & -94561.39 \\ 128.19 & -21677.98 & 193520.48 & -328303.55 & 156561.11 \\ -34.77 & 10682.92 & -93251.32 & 157817.07 & -75324.40 \end{pmatrix}$$

Using `[D R T] = distance(A,B,C,A1,B1,C1)` to compute the shape distance we obtain

```
D =  1.4615

R =
    1.0000    0.0072   -0.0060
   -0.0073    1.0000   -0.0053
    0.0059    0.0053    1.0000

T =
    2.0909
    2.1280
   -2.3780
```

Because we did not modify the position/orientation we get nearly the identity matrix R and a small translation vector.

3.5.2. *Second example*

We consider the same surfaces as in Example 1. But now the second surface is modified in its position/orientation. The coefficients of this second surface are:

$$A1 = \begin{pmatrix} 84.89 & -225.20 & 89.90 & -55.72 & 9.39 \\ -313.00 & 431.87 & -646.10 & 159.53 & 85.64 \\ 175.22 & -1053.91 & 1689.54 & -634.18 & -9.25 \\ -169.97 & 965.06 & -1637.39 & 878.26 & -265.45 \\ 56.19 & -307.99 & 572.68 & -449.29 & 234.52 \end{pmatrix}$$

$$B1 = \begin{pmatrix} 9.88 & -64.37 & 30.77 & -12.00 & -18.26 \\ 35.15 & 112.66 & -45944.39 & 90704.60 & -44889.54 \\ 86.97 & -35479.89 & 379130.18 & -660532.19 & 316856.12 \\ -113.34 & 70601.41 & -644256.40 & 1097173.65 & -523444.36 \\ 44.62 & -35267.12 & 310960.78 & -527072.28 & 251341.42 \end{pmatrix}$$

$$C1 = \begin{pmatrix} 23.32 & -219.79 & -24.83 & -15.38 & 26.69 \\ 182.63 & -221.97 & 14372.87 & -27966.17 & 13794.57 \\ -92.27 & 11276.90 & -116657.13 & 202626.82 & -97130.84 \\ 139.74 & -22066.42 & 197707.10 & -336012.30 & 160167.65 \\ -72.20 & 11035.18 & -95374.33 & 161219.57 & -76755.43 \end{pmatrix}$$

The surface $\mathbf{x}'$ is plotted by the commands

```
[X1,Y1,Z1] = evalpoints(A1,B1,C1,0.05)
mesh(X1,Y1,Z1)
```

and shown in Figure 5. If we now compute again the shape distance using `[D R T] = distance(A,B,C,A1,B1,C1)`, we obtain

```
D =    1.4609

R =
   -0.2967    0.8967   -0.3284
    0.5104   -0.1417   -0.8482
   -0.8071   -0.4193   -0.4157


T =
   40.7155
    9.7456
  116.2242
```

To decompose R as a product of three rotation matrices (18) we use the command `theta = rotangle(R)` and get

```
theta =    2.3518    0.9392   -2.0974
```

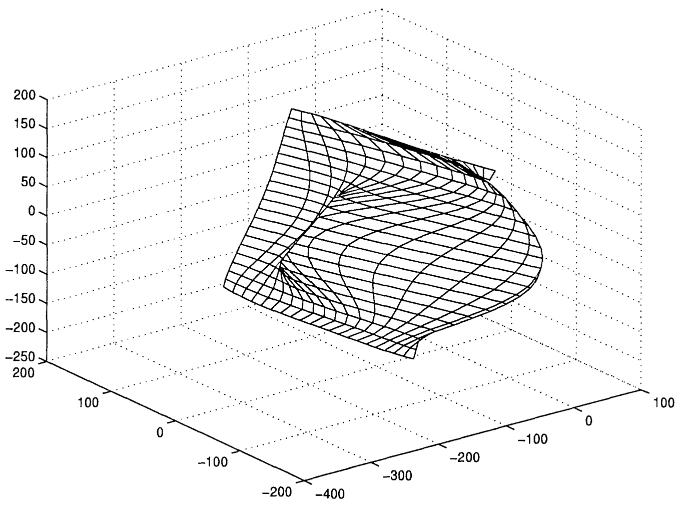

Figure 5. Surface $\mathbf{x}'$

We obtain the best-fit position where the mean square distance takes its minimum. We realize that the shape distance is again the same as before. (The small difference is due to the rounding of the data matrices. In order to make the results more legible and reproducible for the reader, we used the rounded numbers of the coefficients printed in this paper).

In order to check the result, we transform the point set used to plot surface $\mathbf{x}'$ and draw the transformed surface:

```
 [X2,Y2,Z2] = transf(X1,Y1,Z1,R,T)
 mesh(X2,Y2,Z2)
```

We make use of the function `transf`:

```
function [X1,Y1,Z1] = transf(X,Y,Z,R,t)
% TRANSF computes an affine transformation of the point
% set defined by the matrices X,Y,Z.
% [X1,Y1,Z1] = transf(X,Y,Z,R,t) produces a new point set
% such that
% [X1(i,j);Y1(i,j);Z1(i,j)] = R*[X(i,j);Y(i,j);Z(i,j)] + t
% holds for all indices i and j.

[n, m] = size(X);
 H = R*reshape([X,Y,Z],[m*n,3])' + t*ones(1,m*n);
 X1 = reshape(H(1,:),n,m);
 Y1 = reshape(H(2,:),n,m);
 Z1 = reshape(H(3,:),n,m);
```

The plot we obtain can be seen in Figure 6. The surface looks like a good approximation to $\mathbf{x}$.

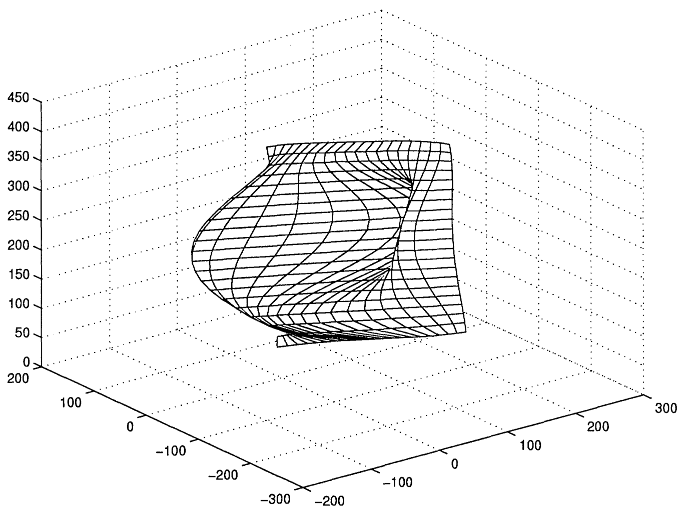

Figure 6. Back-transformed Surface

References

Gander, W.: 1990, 'Algorithms for the Polar Decomposition', *SIAM J. on Sci. and Stat. Comp.*, **Vol. 11, No. 6**, pp. 1102–1115

Gander, W.: 1997, 'Least Square Fit of Point Clouds', in: W. Gander and J. Hřebíček, ed.: *Solving Problems in Scientific Computing Using Maple and Matlab*, ch. 23, pp. 339–349. Springer, Berlin etc., 3rd edition.

Gander, Walter, Golub, Gene H. and Strebel, Rolf: 1994, 'Least-Squares Fitting of Circles and Ellipses', *BIT*, **Vol. 34**, pp. 558-578.

Golub, Gene H. and Van Loan, Charles F.: 1996, 'Matrix Computations'. 3rd ed. Baltimore Johns Hopkins University Press,

Hanson, R. and Norris, M.: 1981, 'Analysis of Measurements Based on the Singular Value Decomposition', *SIAM J. on Sci. and Stat. Comp.*, **Vol. 2, No. 3**, pp.

Sourlier, D.: 1995, 'Case Study 3: Exact Measurement of a Sculptured Surface (Aircraft Wing', in: John A Bosch ed., *Coordinate Measuring Machines and Systems*, in: W. Knapp: Chapter 12 , Marcel Dekker, pp. 327 - 332

Sourlier D.: 1995, 'Three Dimensional Feature Independent Best-fit in Coordinate Metrology', *PhD thesis*, **No. 11319**, Eidgenössische Technische Hochschule Zürich

Address for Offprints: Prof. Walter Gander, Wissenschaftliches Rechnen, ETH Zentrum, CH-8092 Zürich, Switzerland