XX IMEKO World Congress
Metrology for Green Growth
September 9–14, 2012, Busan, Republic of Korea

NEW METHOD FOR FREE-FORM SURFACE FITTING IN PRECISION METROLOGY

Hoang D. Minh, and Alistair B. Forbes

National Physical Laboratory, UK
Minh.hoang@npl.co.uk and Alistair.forbes@npl.co.uk

Abstract: To assess if a given manufactured artifact conform to a template CAD model, the artifact is measured by a coordinate measuring system to obtain a point cloud. The point cloud is compared to the CAD model by fitting the point cloud to the CAD model. One of approaches is based on a method required to determine footpoint parameters.

The footpoints are the projection of the points in the point cloud onto the CAD surface. In this paper we introduce a new efficient method for estimating footpoint parameters.

Keywords: CAD model, point cloud fitting, footpoint parameter, free-form

1. INTRODUCTION

To determine if a given manufactured artifact conforms to its template CAD model, one needs to measure the manufactured artifact to obtain a set of points, e.g., point clouds data as by laser technique and compares with the CAD model. A least squares fitting approach is often used to compare the point cloud data with the CAD model [1]. One of approaches is required to determine footpoint parameters (u, v). The footpoints are the projection of the points in the point cloud to the CAD surface. They can be obtained by solving the least squares problem

$$\min_{u_i,v_i}\left(\sum_{i=1}^{m}(\boldsymbol{x}_i-\mathbf{f}(u_i,v_i))^{\mathrm{T}}(\boldsymbol{x}_i-\mathbf{f}(u_i,v_i))\right) \quad (1)$$

subject to $0 \le u_i \le 1$ and $0 \le v_i \le 1$,

where (u_i, v_i) is the footpoint corresponding to the point $\boldsymbol{x}_i = (x_i, y_i, z_i)$ of the point cloud, and $\mathbf{f}(u,v)$ is the NURBS representation of the CAD model [2]. The constraints on u and v arise from the NURB representation of the surface.

An iterative method is required to solve the nonlinear constrained least squares problem (1). It requires good initial estimate of u and v otherwise the algorithms may not converge.

In this paper we introduce a new approach to determine good initial estimate of u and v. We assume that the CAD model and the point cloud are provided in the same reference coordinate.

2. SURFACE FITTING

A general surface fitting, which fits the data points $\mathrm{X} = \{\boldsymbol{x}_i\}$ to the surface $\boldsymbol{S}$, is defined as follows:

$$\min \left\| d(\boldsymbol{S}, \boldsymbol{x}_i) \right\|_p, \quad (2)$$

where d is the distance measured in the norm p (for $p = 2$ we have Euclidean distance, for $p = \infty$, we have Chebyshev distance).

Let the surface $\boldsymbol{S}$ be defined by the parametric surface $\mathbf{f}(\mathbf{u},\mathbf{b})$: $\mathrm{R}^{k-1} \times \mathrm{R}^{q} \rightarrow \mathrm{R}^{k}$, where u are the footpoint parameters and b are shape, size and position parameters and k=3, and assuming that we are interested in 2-norm, then problem (2) become

$$\min_{\mathbf{u},\mathbf{b}}\left(\sum_{i=1}^{m}(\boldsymbol{x}_i-\mathbf{f}(\mathbf{u}_i,\mathbf{b}))^{\mathrm{T}}(\boldsymbol{x}_i-\mathbf{f}(\mathbf{u}_i,\mathbf{b}))\right) \quad (3)$$

There are two general approaches to solving problem (3): all at once approach and separation of variables

(a) All at once approach: the problem (3) is solved directly for footpoints **u** and the shape, size and position parameters **b** at the same time in one loop. The Jacobian of the least squares problem (3) has block-angular structure, which will be discussed in the following section.

(b) Separation of variables approach: in this approach the solution for **u** and **b** are separated into two loops: suppose $\mathbf{u}^*_i$ solves the ith foot-point problem

$$\min_{\mathbf{u}_i}(\mathbf{x}_i-\mathbf{f}(\mathbf{u}_i,\mathbf{b}))^{\mathrm{T}}(\mathbf{x}_i-\mathbf{f}(\mathbf{u}_i,\mathbf{b})), \quad (4)$$

defining $\mathbf{u}^*_i = \mathbf{u}^*_i(\mathbf{b})$ as a function of **b.** Setting

$$d_i^2(\mathbf{b}) = \left(\mathbf{x}_i - f_i^*(\mathbf{b})\right)^T\left(\mathbf{x}_i - f_i^*(\mathbf{b})\right),$$
$$f_i^*(\mathbf{b}) = \mathbf{f}\left(u_i^*(\mathbf{b}), \mathbf{b}\right),$$

also a function of **b.** Problem (3) becomes

$$\min_{\mathbf{b}} \sum_{i=1}^{m} d_i^2(\mathbf{b}). \quad (5)$$

The quantity $d_i(\mathbf{b})$ is the orthogonal distance from the data point $\mathbf{x}_i$ to the surface $\mathbf{f}(\mathbf{u},\mathbf{b})$. The solution to problem (3) requires two loops: (2) the outer loop is the solution of a standard nonlinear least squares problem and (5) the inner loop is the solution of the footpoints problem (4). The footpoints problem can be solved as collections of individual points or all foot points at once. To solve problem (5) using the Gauss-Newton algorithm, it is required to calculate the partial derivatives $\partial d_i/\partial b_k$ as follows.

Let $\mathbf{x}$ be a data point reasonably close to the surface $\mathbf{u}\rightarrow \mathbf{f}(\mathbf{u},\mathbf{b})$ and let $\mathbf{u}^*$ solve the *foot point problem* [3]

$$\min_{\mathbf{u}} (x - \mathbf{f}(\mathbf{u},\mathbf{b}))^{\mathrm{T}} (\mathbf{x} - \mathbf{f}(\mathbf{u},\mathbf{b})), \qquad (6)$$

so that $\mathbf{u}^* = \mathbf{u}^*(\mathbf{b})$ specifies the point $\mathbf{f}^* = \mathbf{f}(\mathbf{u}^*,\mathbf{b})$ on $\mathbf{f}(\mathbf{u},\mathbf{b})$ closest to $\mathbf{x}$. The optimality conditions associated with problem (3) implicitly define $\mathbf{u}^*$ as a function of $\mathbf{b}$ through the equations

$$(\mathbf{x} - \mathbf{f}(\mathbf{u},\mathbf{b}))^{\mathrm{T}} \mathbf{f}_u = 0, \qquad (\mathbf{x} - \mathbf{f}(\mathbf{u},\mathbf{b}))^{\mathrm{T}} \mathbf{f}_v = 0 ,$$

where $\mathbf{f}_u = \partial\mathbf{f}/\partial u$ and $\mathrm{f}_v = \partial\mathbf{f}/\partial v$. Let $\mathbf{n} = \mathbf{n}(\mathbf{b})$ be the normal to the surface at $\mathbf{f}^*$, likewise a function of $\mathbf{b}$, and set

$$d(\mathbf{x},\mathbf{b}) = (\mathbf{x} - \mathbf{f}^*)^{\mathrm{T}} \mathbf{n} = (\mathbf{x} - \mathbf{f}(\mathbf{u}^*(\mathbf{b}),\mathbf{b}))^{\mathrm{T}} \mathbf{n}(\mathbf{b}). \quad (7)$$

Then $d(\mathbf{x},\mathbf{b})$ is the signed distance of $\mathbf{x}$ from $\mathbf{f}(\mathbf{x}, \mathbf{b})$, where the sign is consistent with the convention for choosing the surface normal. Furthermore,

$$\frac{\partial d}{\partial b_k} = -\left(\frac{\partial \mathbf{f}}{\partial b_k}\right)^{\mathrm{T}} \mathbf{n}, k = 1, \ldots, n,$$

where all terms on the left-hand side are evaluated at $\mathbf{u}^*$. Although $\mathbf{u}^* = \mathbf{u}^*(\mathbf{b})$ and $\mathbf{n} = \mathbf{n}(\mathbf{b})$ are both functions of $\mathbf{b}$, there is no need to calculate their derivatives with respect to $\mathbf{b}$ in order to evaluate the derivatives of the distance function. This is because their contribution is always as linear combinations of vectors orthogonal to $\mathbf{n}$ or $\mathbf{x}$-$\mathbf{f}$.

For standard geometric elements, the distance function $d(\mathbf{x},\mathbf{b})$ can be defined as an explicit function of the parameters but for free-form surfaces, the optimal footpoint parameters $\mathbf{u}^*$ have to be determined using numerical techniques

3. BLOCK-ANGULAR AND BLOCK DIAGONAL STRUCTURE

The Jacobian matrix J associated with the problem (2) has a block-angular structure

$$J = \begin{pmatrix} J_1 & & J_{0,1} \\ & \ddots & \vdots \\ & & J_{0,m} \end{pmatrix},$$

where J_i have dimension of 3×2, and $J_{0,i}$ have dimension of $3 \times n$, n is the number of parameters of the vector $\mathbf{b}$.

The Jacobian matrix J_f associated with least squares problem (4) for the all at once approach has a block diagonal structure

$$J_f = \begin{pmatrix} J_1 & & \\ & \ddots & \\ & & J_m \end{pmatrix}.$$

4. ORDINARY DISTANCE REGRESSION FOR FREE-FORM SURFACES

Unlike ordinary distance regression (ODR) for standard geometries, which can usually be formulated as an unconstrained least squares problem, ODR for free-form surfaces is usually a *box constrained* one due to the parameters u and v being constrained to lie in a given interval, usually [0, 1]. The constraints on the parametric variables depend on the parametric representations of the surface. One of the most popular parametric representation of surfaces used in CAD/CAM industry is NURBS representation [4]. The constraints on the parametric variables $\mathbf{u} = (u,v)$ are $0 \le u \le 1$ and $0 \le v \le 1$. The constrained optimisation problem (3) with box constraints on $\mathbf{u}$ has special block-angular structure [5,6]. We exploit this structure in our MATLAB implementation of the solution algorithm.

5. ALGORITHM FOR ESTIMATING INITIAL FOOTPOINT PARAMETERS

The following algorithm is based on the result of computational geometry, which can determine nearest neighbours points efficiently, based on Delaunay Triangulations or Voronoi diagram [7-14] . This task involves two steps: the first one is to build Delaunay Triangulations or Voronoi diagram, and the second one is to locate a point in the Delaunay Triangulations or Voronoi diagram in the first step. There are practical efficient algorithms for these tasks, for example, the complexity of the second task, the point location problem, was reported in [10] and is of $O(n^{1/4})$ for n random points, and for the first task the complexity bound is $O(n)$ under some mild sampling condition of the points [1].

In this paper, we use the approach of Delaunay Triangulations implemented in MATLAB. However, our idea can be applied to the other approach. To find initial values for parameters (u,v) for each point belonging to the point cloud $\mathbf{P}$, we do as follows:

Step 1. The reference surface is sampled along the u and v directions obtaining a set of points, called *sampling points,* $\mathbf{P}_s$ with corresponding values of (u,v).

Step 2. Delaunay tessellation of the sampling points.

Step 3. Search for closest points between the point cloud $\mathbf{P}$ and $\mathbf{P}_s$.

Step 4. Assign initial value of (u,v) for each point in $\mathbf{P}$ using (u, v) of the closest point in $\mathbf{P}_s$.

To obtain a better estimate of (u,v) one needs to sample as many points as possible.

6. EXAMPLES

For the purpose of illustration, we have create 3 free form NURBS surfaces as follows:

(a) The NURBS surface is of order of 6 by 6 and has 81 control points. Using this NURBS

surface as a reference CAD model, we generate a test dataset using the null space method [1]. The test dataset consists of 5151 points. The test dataset points are close and deviated from the reference surface. Figure 1 shows the NURBS surface and its deviation from the test dataset.

(b) The NURBS surface is of order of 3 by 3 and has 25 control points. Figure 2 shows the NURBS surface and its deviation from the test dataset.

(c) The NURBS surface is a blade, consisting of two NURBS surfaces, each of order 6 by 6 and consisting of 126 control points. Figure 3 shows the NURBS surface and its deviation of the test dataset.

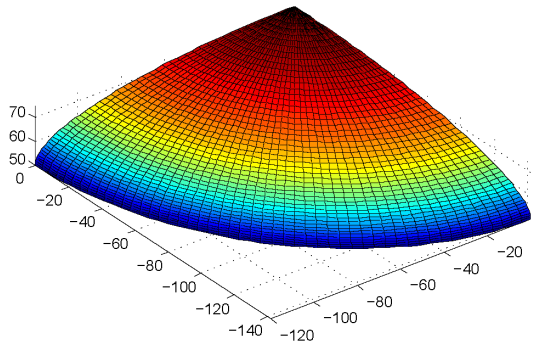

Figure 1 NURBS surface of order 6 by 6 and its deviation from a test dataset consisting of 5151 points. Colour represents the deviation at the point

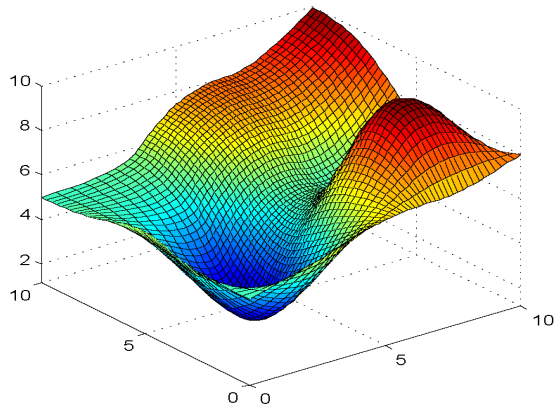

Figure 2 NURBS surface of order 3 by 3 and its deviation from a test dataset consisting of 2601 points. Colour represents the deviation at the point

Problem	#Iterations	CPU time (s)	$\|\mathbf{u}-\mathbf{u}^*\|_2$	$\|\mathbf{u}-\mathbf{u}^*\|_\infty$	Notes
(a)	6	12.4	0.2970	0.0060	*
(a)	24	30.5			
(b)	35	17.3	0.2198	0.0077	*
(b)	Fail to converge				
(c)	18	14	0.1677	0.0059	*
(c)	Fail to converge				
*: with our estimate for initial values					

Table 1: summary of computational results

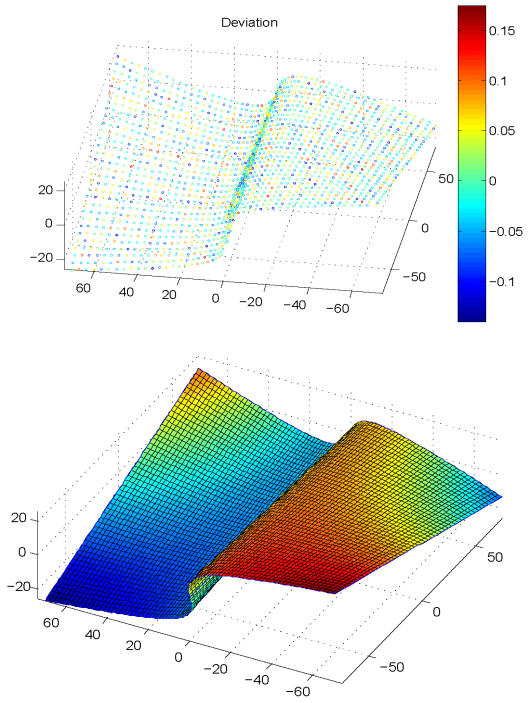

Figure 3 NURBS surface consist of two NURBS surfaces, each surface is of order 6 by 6 and its deviation from a test dataset consisting of 1692 points. Colour represents the deviation at the point

The timings below refer to computations carried out on a PC with Intel Core 2 Duo CPU E4500, 2.2GHz, 3GB of RAM running Windows XP SP2 and MATLAB version R2010a.

To determine footpoints (u,v) corresponding to the points in the test dataset, we use the algorithm in Section 5 to generate initial estimates of (u,v). The reference surface is sampled with $N = 100$ in the u direction and $M = 100$ in the v direction. The obtained u,v values were used as initial estimate for the ODR algorithm. We then use MATLAB function `lsqnonlin` to solve the problem.

For surface (a), with our initial estimate the ODR algorithm required 6 iterations to converge to a solution, taking 12.4 seconds. Without our initial estimate, the algorithm required 24 iterations (30.5 seconds) to converge.

For surface (b), with our initial estimate values it took 35 iterations and 17.3 seconds to converge to the solution, without our initial estimate values it was trapped in a point far from the solution.

For surface (c), with our initial estimate values it took 18 iterations and 14 seconds to converge to the solution, without our initial estimate values the `lsqnonlin` was trapped in a point far from the solution.

Table 1 summarizes the computational results.

5. CONCLUSION

We have described a new, efficient method to obtain good estimate values for fitting with free-form surfaces. The method uses the result from the computational geometry which can determine nearest points efficiently.

6. ACKNOWLEDGEMENTS

This work has been supported by the UK's National Measurement Office Mathematics and Modelling for Metrology programme and with funding by European Union through the European Metrology Research Programme (EMRP) Joint Research Project NEW06, Traceability in Computationally-Intensive Metrology. The EMRP is jointly funded by the EMRP participating countries within EURAMET and the European Union.

7. REFERENCES

[1] A.B. Forbes, H.D. Minh., Form assessment in coordinate metrology, Approximation Algorithms for Complex Systems, 2009.

[2] L. Piegl and W. Tiller. The NURBS Book, 2nd ed., 1997.

[3] A. B. Forbes. Structured nonlinear Gauss-Markov problems. In A. Iske and J. Levesley, editors, Algorithms for Approximation V, pages 167–186, Berlin, Springer, 2006.

[4] The initial graphics exchange specification (IGES) version 5.x, 2006.

[5] M.G. Cox, The least-squares solution of linear equations with blockangular observation matrix, Advances in Reliable Numerical Computation, (1989) 227-240.

[6] A.B. Forbes, Least squares best fit geometry elements, Algorithms for Approximation II, (1990) 311-319.

[7] J.L. Bentley, Multidimensional divide-and-conquer, Commun. ACM, 23 (1980) 15.

[8] J.L. Bentley, M.I. Shamos, Divide-and-conquer in multidimensional space, in: 8th Annu. ACM Sympos. Theory Comput., 1976, pp. 220-230.

[9] M.T. Dickerson, R.L. Drysdale, J.R. Sack., Simple algorithms for enumerating interpoint distances and finding *k* nearest neighbors., Internat. J. Comput. Geom. Appl., 2 (1992) 221-239.

[10] I.S. Ernst P. Mucke, and Binhai Zhu, Fast randomized point location without preprocessing in two- and three-dimensional delaunay triangulations, in: Proceedings of the twelfth annual symposium on Computational geometry, SCG '96,, ACM, New York, NY, USA, 1996, pp. 274–283.

[11] D. Attali, J.-D. Boissonnat, A linear bound on the complexity of the delaunay triangulation of points on polyhedral surfaces, in: SMA '02 Proceedings of the seventh ACM symposium on Solid modeling and applications, 2002.

[12] M.T. Dickerson, R.L. Drysdale., Enumerating *k* distances for *n* points in the plane, in: Proc. 7th Annu. ACM Sympos. Comput. Geom., 1991, pp. 234–238.

[13] M.T. Dickerson, R.S. Drysdale., Fixed-radius near neighbor search algorithms for points and segments, Inform. Process. Lett., 35 (1990) 269–273.

[14] P.M. Vaidya, An $O(n \log n)$ algorithm for the all-nearest-neighbors problem, Discrete Comput. Geom., 4 (1989) 101–115.