

Adaptive T-spline Surface Fitting to Z-Map Models

Jianmin Zheng*

Yimin Wang[†]

Hock Soon Seah[‡]

School of Computer Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798

Abstract

Surface fitting refers to the process of constructing a smooth representation for an object surface from a fairly large number of measured 3D data points. This paper presents an automatic algorithm to construct smooth parametric surfaces using T-splines from z-map data. The algorithm begins with a rough surface approximation and then progressively refines it in the regions where the approximation accuracy does not meet the requirement. The topology of the resulting T-spline surface is determined adaptively based on the local geometric character of the input data and the geometry of the control points is obtained by a least squares procedure. The advantage of the approach is that the resulting surface is C^2 continuous and the refinement is essentially local, resulting in a small number of control points for the surface.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—curve, surface, solid and object representations;

Keywords: T-splines, z-map models, adaptive fitting, surface reconstruction

1 Introduction

With development of data imaging and acquisition devices such as coordinate measuring machines and laser range scanners, the surfaces of real objects or models can be easily measured or scanned, outputting highly precise and regular measurement data in large quantities [Blais 2004]. This naturally raises the problem of how to process such data for research or applications.

Spline surfaces are the major representation of freeform surfaces in CAD/CAM industry. Spline surfaces have parametric equations and can define the differential structure for shape analysis. Therefore in many applications such as manufacturing industry, it is desirable to reconstruct the spline surface representation from 3D discrete models [Krishnamurthy and Levoy 1996; Lee et al. 1997]. While the interpolation method might be used to create a representation that is inefficient in both computing and storage for this task, approximating or fitting methods are preferred because they create a more compact representation for the shape, which needs only a small number of points.

*e-mail: asjzheng@ntu.edu.sg

[†]e-mail: wang0066@ntu.edu.sg

[‡]e-mail: ashseah@ntu.edu.sg

This paper presents an automatic method to reconstruct a spline surface from z-map data. Z-map [Choi et al. 1997] is a discrete representation in which the height values at grid points on the domain plane are stored as a two-dimensional array (see Figure 1). It is a simple and yet versatile model for representing various data. For example, the z-map data is a typical representation of the range data arranged in the form of an image, which is obtained by image capturing devices and represents depth information of the real object along a direction. As a result, z-map is widely used in free-formed object styling, NC machining, GIS, and computer vision.

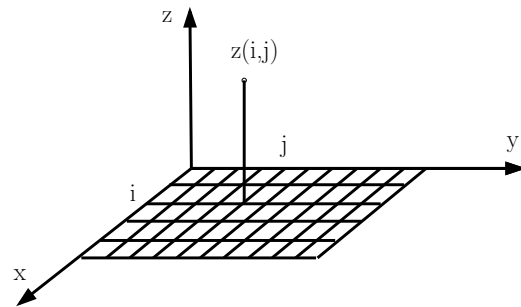


Figure 1: Z-map model

There is a vast amount of literature addressing the problem of reconstructing free-form surfaces from measured data [Schmitt et al. 1986; Solina and Bajcsy 1990; Franke and Nielson 1991; Sakar and Menq 1991; Muraki 1991; Hoppe et al. 1992; Chen and Schmitt 1994; Forsey and Bartels 1995; Park and Kim 1995; Chivate and Jablowok 1995; Jang et al. 1999; Carr et al. 2001]. The reader is referred to [Franke and Nielson 1991] and [Chivate and Jablowok 1995] for excellent surveys. Among the previous work where the surface description models used could be triangular meshes, tensor-product spline surfaces, triangular splines, superquadrics, or radial basis functions and the measured data could be range data, triangular meshes, irregular data or bivariate data, some focused on surface approximation to z-map data points. For example, Jang et al [1999] proposed an approximation method using Gregory patches. The Gregory patches were constructed with a constraint that the G^1 continuity condition was ensured. Lee et al [1997] introduced multilevel B-splines to compute a C^2 continuous interpolation or approximation surface to a set of irregularly spaced bivariate points. Their method could be used to fit z-map data. In this paper, we propose to use T-splines for fitting to z-map data. T-splines [Sederberg et al. 2003; Sederberg et al. 2004] are a new surface modelling technique that generalizes tensor-product non-uniform rational B-spline (NURBS) surfaces by permitting T-junction points in their control grid. We choose to use T-splines for two reasons: (1) T-spline surfaces allow local refinement, which makes them a suitable tool for adaptive fitting. It is thus possible to use a small number of points to represent the fitting surface. (2) T-spline surfaces automatically maintain C^2 continuity and are able to be converted into the NURBS representation. This enables the output of compact NURBS surfaces for use with existing commercial software systems.

The rest of the paper is organized as follows. Section 2 gives a

brief overview of T-splines. Section 3 describes the adaptive fitting algorithm. Section 4 provides some implementation results. The conclusion with future work is drawn in Section 5.

2 T-splines

A T-spline surface is a NURBS surface with T-junctions and is defined by a control grid called T-mesh. The T-mesh is similar to a NURBS control mesh except that in a T-mesh a row or column of control points is permitted to terminate. The final control point in a partial row or column is a T-junction. The T-junctions enable T-spline surfaces to be refined locally. That is, it is possible to add a single control point to a T-spline control grid without propagating an entire row or column of control points and without altering the surface. The T-mesh is also associated with knot information of the T-spline, which is expressed using knot intervals indicating the difference between two knots and assigned to the edges of the T-mesh. Refer to Figure 2 for an example of a T-spline. The left shows the pre-image of the T-mesh in the parameter domain, and the right shows the T-mesh and the surface. The equation for a T-spline surface is

$$P(s, t) = \frac{\sum_{i=1}^k P_i w_i B_i(s, t)}{\sum_{i=1}^k w_i B_i(s, t)} \quad (1)$$

where the P_i are control points and the w_i are control point weights. The T-spline basis function corresponding to control point P_i is $B_i(s, t)$:

$$B_i(s, t) = N[\mathbf{s}_i](s)N[\mathbf{t}_i](t) \quad (2)$$

where $N[\mathbf{s}_i](s), N[\mathbf{t}_i](t)$ are the cubic B-spline basis functions associated with the knot vectors

$$\mathbf{s}_i = [s_{i0}, s_{i1}, s_{i2}, s_{i3}, s_{i4}] \quad (3)$$

and

$$\mathbf{t}_i = [t_{i0}, t_{i1}, t_{i2}, t_{i3}, t_{i4}] \quad (4)$$

respectively. For example,

$$N[\mathbf{s}_i](s) = \begin{cases} \frac{(s-s_{i0})^3}{(s_{i1}-s_{i0})(s_{i3}-s_{i0})(s_{i2}-s_{i0})}, & s_{i0} < s \leq s_{i1} \\ \frac{(s-s_{i0})^2(s_{i2}-s)}{(s_{i2}-s_{i1})(s_{i3}-s_{i0})(s_{i2}-s_{i0})} + \frac{(s_{i3}-s)(s-s_{i0})(s-s_{i1})}{(s_{i2}-s_{i1})(s_{i3}-s_{i1})(s_{i3}-s_{i0})} + \frac{(s_{i4}-s)(s-s_{i1})}{(s_{i2}-s_{i1})(s_{i4}-s_{i1})(s_{i3}-s_{i1})}, & s_{i1} < s \leq s_{i2} \\ \frac{(s-s_{i0})(s_{i3}-s)^2}{(s_{i3}-s_{i2})(s_{i3}-s_{i1})(s_{i3}-s_{i0})} + \frac{(s_{i4}-s)(s_{i3}-s)(s-s_{i1})}{(s_{i3}-s_{i2})(s_{i4}-s_{i1})(s_{i3}-s_{i1})} + \frac{(s_{i4}-s)^2(s-s_{i2})}{(s_{i3}-s_{i2})(s_{i4}-s_{i2})(s_{i4}-s_{i1})}, & s_{i2} < s \leq s_{i3} \\ \frac{(s_{i4}-s)^3}{(s_{i4}-s_{i3})(s_{i4}-s_{i2})(s_{i4}-s_{i1})}, & s_{i3} < s \leq s_{i4} \\ 0, & s < s_{i0} \text{ or } s > s_{i4} \end{cases}$$

The knot vectors \mathbf{s}_i and \mathbf{t}_i are extracted from the T-mesh neighborhood of P_i . The details on T-splines can be found in [Sederberg et al. 2003; Sederberg et al. 2004].

2.1 Local refinement

One of the most fundamental operations in T-splines is local refinement or local knot insertion, which means to insert one or more control points into a T-mesh without changing the shape of the T-spline surface. The algorithm of inserting new control point(s) can be described as follows:

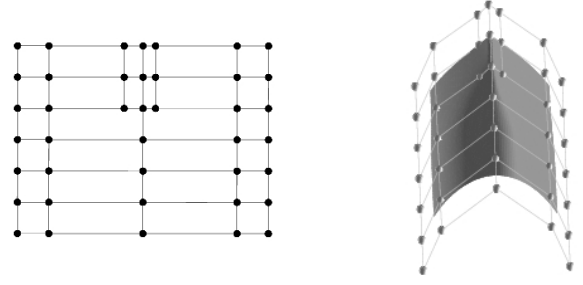


Figure 2: An example of a T-spline: the pre-image, the T-mesh and the surface.

Local knot insertion algorithm

- 1) Insert new control point(s) into the T-mesh.
- 2) If any basis function is missing a knot inferred from the current T-mesh, perform the necessary knot insertions into that basis function.
- 3) If any basis function has a knot that is not indicated in the current T-mesh or can not find a corresponding point in the current T-mesh, add an appropriate control point into the T-mesh.
- 4) Repeat steps 2) and 3) until there is no new operation. Now every basis function properly associates with a control point in the T-mesh.

The basic idea of the above algorithm is to maintain the validity of the T-mesh and to ensure that the basis functions and the control points are properly associated. The formulae for the basis function refinement can be found in [Sederberg et al. 2004].

2.2 Standard T-splines and semi-standard T-splines

In general, a T-spline surface is a rational surface except it is a standard T-spline or semi-standard T-spline. A standard T-spline is one for which, if all weights $w_i = 1$, then $\sum_{i=1}^k w_i B_i(s, t) = 1$. A semi-standard T-spline is one for which $\sum_{i=1}^k w_i B_i(s, t) = 1$ and not all $w_i = 1$. An important property of T-splines is that if we perform a local refinement on a semi-standard or standard T-spline, the result will always be a standard or semi-standard T-spline. Both standard and semi-standard T-splines define piecewise polynomial surfaces.

3 Adaptive surface fitting

Assume the z-map data is given in the form of $z(i, j)$, $i = 1, \dots, m$, and $j = 1, \dots, n$. Together, $(i, j, z(i, j))$ gives the bivariate data point. The independent data (i, j) is 2D and the dependent data $z(i, j)$ is a simple scalar. Thus the fitting problem is to find a T-spline function $f(x, y)$ such that the surface $(x, y, f(x, y))$ approximates the bivariate data, where

$$f(x, y) = \frac{\sum_{i=1}^k f_i w_i B_i(x, y)}{\sum_{i=1}^k w_i B_i(x, y)}, \quad x \in [1, m], j \in [1, n] \quad (5)$$

This involves determining the topology of the T-mesh and finding the values of f_i and w_i .

Our method begins with a bicubic B-spline surface grid as the initial setting of the T-mesh topology. This implies that the surface is actually a standard T-spline. Then an optimization is performed

to compute the best T-spline surface corresponding to the specified T-mesh topology for approximating the data. The surface is tested to decide if it is sufficiently close to the underlying data. If the testing is passed, the surface is accepted. Otherwise, it undergoes the next phase of local refinement, which refines the T-mesh to generate a new topology, and goes back to the optimization phase. The process continues until the final surface is accepted. Figure 3 shows the flowchart of the procedure. We explain the details below.

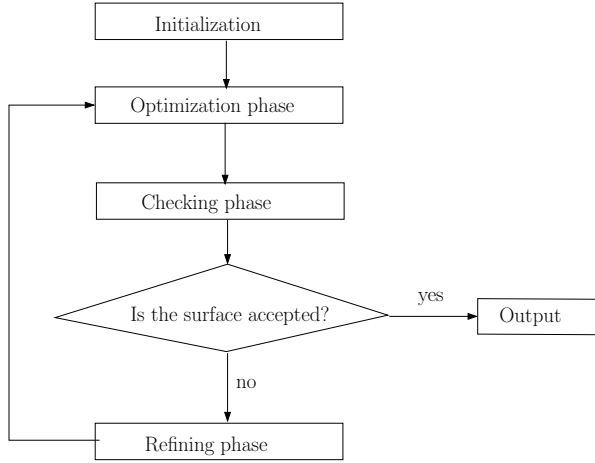


Figure 3: The flow chart of the adaptive fitting procedure

3.1 Initialization

The initial surface is chosen to be a bicubic B-spline patch over the rectangular domain $[1, m] \times [1, n]$. Choose the knot vectors along the x - and y -directions to be $\{-1, 0, 1, m, m+1, m+2\}$ and $\{-1, 0, 1, n, n+1, n+2\}$, respectively. In the parameter domain these knots form a pre-image of the control grid of the surface. Let us denote the pre-image by D . We also let all control point weights be one. Denote by W the set of all weights corresponding to the control points. Then D and W define a class of T-splines that have the same pre-image D and the same weights W . We denote this class by S . Each surface in S is a standard T-spline no matter what values the control points are.

3.2 Optimization phase

Consider the class S of T-splines that are determined by pre-image D and weight set W . The pre-image D and the weight set W are the ones either given in the initialization or produced by local refinement of the initial D and W . We seek to find the T-spline in S , which best fits the data under a certain criterion. It is important to note that any T-spline in S is a standard or semi-standard one. This is because the initial class S given in initialization consists of only standard T-splines and our local refinement algorithm converts a standard T-spline into either a standard or a semi-standard T-spline. Therefore the denominator of any T-spline in S is identically one and we only need to focus on the numerator:

$$f(x, y) = \sum_{i=1}^k f_i w_i B_i(x, y), \quad x \in [1, m], \quad y \in [1, n] \quad (6)$$

We thus try to find f_i such that $\sum_{i=1}^m \sum_{j=1}^n (f(i, j) - z(i, j))^2$ is minimized. This is a least-squares approximation. More generally,

we would like to include a fairness functional $J_{fair}(f)$ into the approximation to generate a fair surface. Thus the optimization problem becomes the minimizing of the following objective function:

$$G(f_1, \dots, f_k) = \sum_{i=1}^m \sum_{j=1}^n (f(i, j) - z(i, j))^2 + \sigma J_{fair}(f) \quad (7)$$

where σ is a non-zero constant somehow measuring the tradeoff between fairing and approximating. A larger σ will lead to a more fair surface and a smaller σ will give a more accurate approximation. Usually σ is chosen to be between 0.01 and 0.1.

There are a few choices for the fairness functionals [Celniker and Gossard 1991]. Considering the low computational complexity, we choose the simple (discrete) thin plate energy functional:

$$J_{fair}(f) = \sum_{i=1}^m \sum_{j=1}^n (f_{xx}^2(i, j) + 2f_{xy}^2(i, j) + f_{yy}^2(i, j)) \quad (8)$$

To solve the optimization problem, we differentiate the objective function with respect to f_g and equate the partial derivative to zero. This leads to

$$\begin{aligned} & \sum_{l=1}^k w_l \sum_{i=1}^m \sum_{j=1}^n (B_l(i, j) B_g(i, j) + \sigma (B_{lxx}(i, j) B_{gxx}(i, j) \\ & + 2B_{lxy}(i, j) B_{gxy}(i, j) + B_{lyy}(i, j) B_{gyy}(i, j))) f_l \\ & = \sum_{i=1}^m \sum_{j=1}^n z(i, j) B_g(i, j) \end{aligned}$$

for $g = 1, \dots, k$, where $B_{lxx} = \frac{d^2 N[\mathbf{x}_l](x)}{dx^2} N[\mathbf{y}_l](y)$ is the second order partial derivative of B_l with respect to x , $B_{lxy} = \frac{dN[\mathbf{x}_l](x)}{dx} \frac{dN[\mathbf{y}_l](y)}{dy}$ is the mixed partial derivative of B_l with respect to x and y , and the others are similarly defined. If we introduce a $k \times k$ matrix $A = (a_{gl})$, two $k \times 1$ matrices $C = (c_g)$ and $F = (f_g)$ with

$$\begin{aligned} a_{gl} &= w_l \sum_{i=1}^m \sum_{j=1}^n (B_l(i, j) B_g(i, j) + \sigma (B_{lxx}(i, j) B_{gxx}(i, j) \\ & + 2B_{lxy}(i, j) B_{gxy}(i, j) + B_{lyy}(i, j) B_{gyy}(i, j))) \end{aligned} \quad (9)$$

and

$$c_g = \sum_{i=1}^m \sum_{j=1}^n z(i, j) B_g(i, j), \quad (10)$$

then the above linear equations can be written as $AF = C$. Solving the linear system using standard numerical methods such as Gaussian elimination gives the solution $F = A^{-1}C$, which defines the T-spline surface.

3.3 Checking phase

Once the surface is computed, it should be checked if the result is satisfactory. A simple way is to test whether the vertical distance from each point in the z -map database to the surface is below a given tolerance. If all these distances $|f(i, j) - z(i, j)|$ pass the test, the surface is accepted. Otherwise, the regions with poor approximation are marked for further processing.

The phase also compares the maximum distance against the one obtained in the previous iteration. In some situations, though the objective function becomes smaller, the maximum distance does not improve substantially due to complicated distribution of data. We need to adjust the parameter σ to make the error term in the objective function contribute more. Specifically, if the maximum distance is not decreased by a factor μ , then we decrease the parameter σ by a factor η . Both μ and η are constants falling between 0 and 1. In our implementation, we set $\mu = 0.9$ and $\eta = 0.2$.

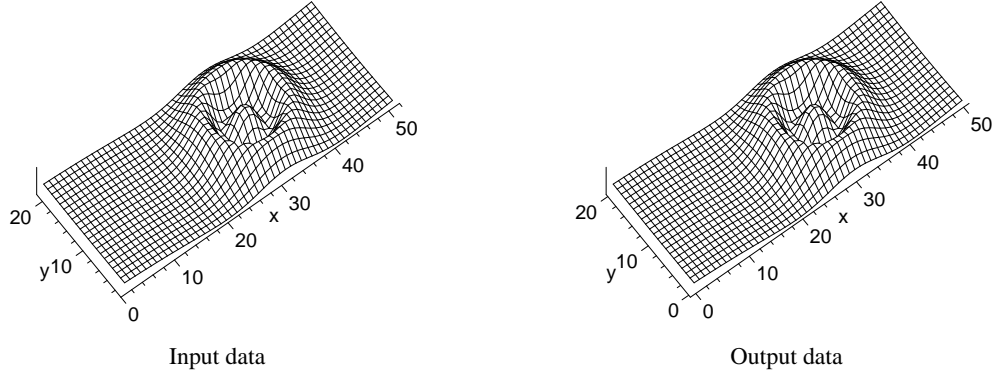


Figure 4: The input data and the output data from the resulting T-spline.

Since there always exists a bicubic B-spline surface interpolating the z-map data and giving the approximation error of zero, the trick of decreasing the value of σ ensures the convergence of the iterations. Moreover, in the extreme case where σ is set to zero and the tolerance is equal to zero, at most $(m+2) \times (n+2)$ control points are needed because they are sufficient to define a bicubic B-spline surface interpolating those z-map data points. This implies that the algorithm will find the surface in a finite number of steps.

3.4 Refining phase

When a region contains a violating point at which the approximation error is greater than the tolerance, this region is called an offending region. The violating point could be inside the region or on the border of the region. For offending regions, our strategy is to split them. While there are several different ways to split regions, we follow the principle that an offending region should, in general, be split roughly in half in the direction which the region has a larger knot difference. Specifically, assume the region is bounded by a box whose lower-left corner has coordinates (s_{min}, t_{min}) and whose upper-right corner has coordinates (s_{max}, t_{max}) . If $s_{max} - s_{min} > t_{max} - t_{min}$, then we split the region at $s = [(s_{max} + s_{min})/2]$. If $s_{max} - s_{min} < t_{max} - t_{min}$, then we split the region at $t = [(t_{max} + t_{min})/2]$. In case both the knot differences are the same, then either direction could be chosen for splitting.

The actual splitting of a face is accomplished by performing a local refinement in which the two endpoints of the line segment used to split the region are inserted into the T-mesh using the T-spline's local knot insertion algorithm. The knot insertion algorithm might introduce a few additional points into the T-mesh. After the refinement is finished, a new T-mesh topology D and the corresponding weight set W are specified. What remains to define a T-spline surface is to determine the control points.

Note that in the above approach we insert those splitting points that correspond to certain z-map data points and are not the existing control points. If an offending face contains no such splitting points, there is no way to split the face. In case all offending faces are not able to be split, we arbitrarily choose a point in the whole domain for insertion, which corresponds to a certain z-map point and is not an existing control point.

4 Implementation results

To test the algorithm, we have created a set of z-map data by evenly sampling the function

$$f(x, y) = 5 - 5\sin\left(\frac{400}{(x-31)^2 + (y-11)^2 + 392}\right)$$

within the domain $[1, 51] \times [1, 21]$. The step-length was 1 and the z-map data thus contained $51 \times 21 = 1071$ points. Figure 4(left) shows a mesh formed by these points.

An approximate T-spline surface was computed for tolerance $= 0.07$ and $\sigma = 0$. After the resulting surface was obtained, we sampled it with the step-length of 1. The sampled points formed a mesh which is shown in Figure 4(right). Figure 5 shows the T-mesh and the corresponding T-spline surface. The T-spline surface provided a visually excellent C^2 continuous approximation to the z-map data. We also checked the approximation errors between the T-spline surface and the original function $f(x, y)$ at denser sampling data points. All the errors were below the tolerance of 0.07.

This T-spline surface was created after eight iterations of fitting. At the first iteration, the T-mesh consisted of 4×4 control points. Each of subsequent iterations refined the previous T-mesh by adding more control points. The pre-image of the T-mesh at each iteration is shown in Figure 6.a-h. Eventually, the final T-mesh contained 521 control points. Observe Figure 6. The first few pre-images look more like global refinements. This suggests that in some situations an $M \times N$ ($M, N > 4$) rather than 4×4 bicubic B-spline surface could be used in the initialization step in order to speed up the computation.

We have also applied the method to range data, which has 20×20 points. The range data is displayed in Figure 7(left) and a mesh formed by the range data is also displayed on the right for a clear view. We chose the tolerance $= 0.07$. We tested the method for $\sigma = 0$ and $\sigma = 0.05$. The algorithm stopped after six iterations for both situations with the maximum errors of 0.04 and 0.068. The pre-images are displayed in Figure 8. The T-mesh for $\sigma = 0$ contains 269 control points and is shown in Figure 9(left). Note that one control point at coordinates $(1, 20)$ has a very large z -value and therefore the figure is displayed in unconstrained scaling. The T-mesh for $\sigma = 0.05$ contains 257 control points and is shown in Figure 9(right). The T-spline surfaces for $\sigma = 0$ and $\sigma = 0.05$ are shown in Figure 10. These two figures are created by tessellating the T-spline surfaces with tessellation step $= 0.5$. Figure 10(left) is also displayed in unconstrained scaling. It should be pointed out that if we tessellate both T-spline surfaces with tessellation step $= 1$,

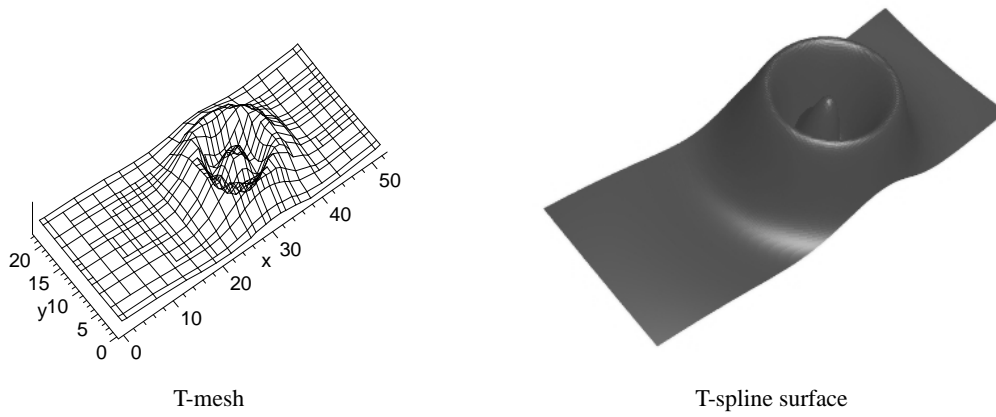


Figure 5: The T-mesh and the T-spline surface.

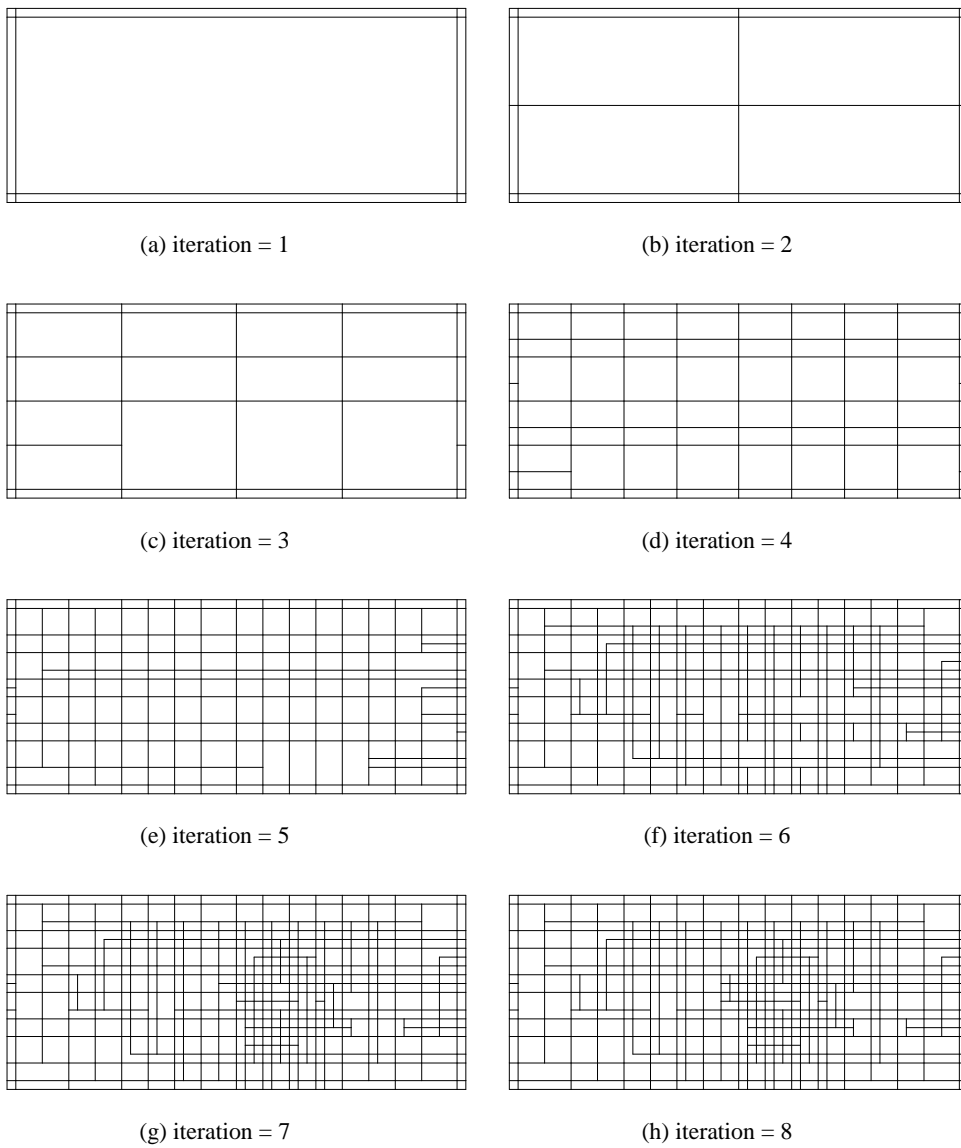


Figure 6: The pre-images for iteration 1-8.

then the figures look almost the same as Figure 7(right). This means both T-spline surfaces satisfy the tolerance requirement. Obviously, the surface for $\sigma = 0.05$ is much smooth or fair.

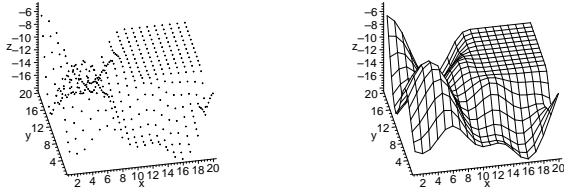


Figure 7: The range data and the mesh formed by the range data.

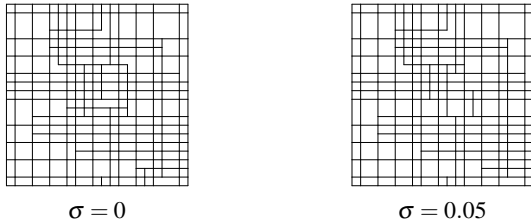


Figure 8: The pre-images of the resulting T-meshes for $\sigma = 0$ and 0.05 .

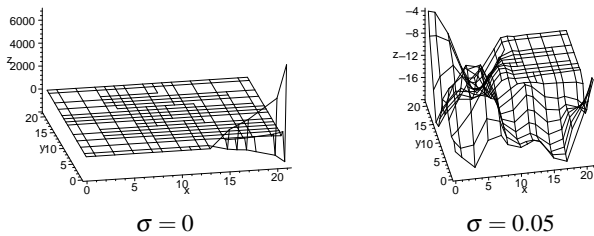


Figure 9: The T-meshes for $\sigma = 0$ and 0.05 .

5 Conclusions

We have described an algorithm of constructing a (standard or semi-standard) T-spline surface to approximate z-map data. The use of T-splines promises an adaptive approach in that local refinement is carried out at the problematic regions. This enables the algorithm to output a more compact surface. The choice of a B-spline surface in the initialization step ensures the refined T-splines are either standard or semi-standard, and thus simplifies the computation. The introduction of a discrete fairness functional helps create a smooth surface. Therefore the method could be useful in some applications such as 3D shape reconstruction and 2D image reconstruction.

Our algorithm can be easily extended to fit gridded data set where the points are arranged in an $M \times N$ grid in the 3D space. Since the data is gridded, each point P_{ij} could be simply assigned with parameter (s_i, t_j) for $i = 1, \dots, M$ and $j = 1, \dots, N$. Therefore what we need to do is just to replace each scalar $z[i, j]$ by 3D point P_{ij} . As a result, we obtain a 3D parametric T-spline surface instead of a function surface $z = f(x, y)$. A more challenging problem is how to extend the algorithm to fit a set of scattered data points or

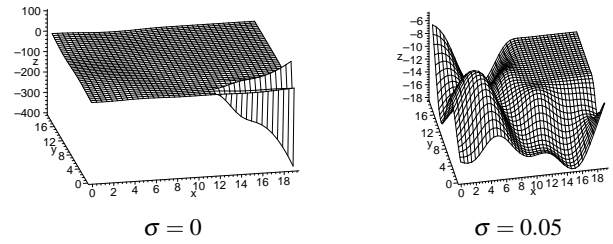


Figure 10: The resulting T-spline surfaces for $\sigma = 0$ and 0.05 .

an arbitrarily topological triangular mesh. This is currently under investigation.

Note that our approach to finding the control points of the T-spline is global in that all the control points have to be re-computed by the least squares method at each iteration. The advantage of this approach is that for the specified T-mesh topology, the T-spline surface obtained is optimal in the L_2 -norm. The disadvantage is that the computation could be time-consuming, especially when the number of the data points is huge. Therefore the problem of how to develop an adaptive, local fitting algorithm warrants future study.

Acknowledgements

This work is supported by the URC-SUG 8/04 of Nanyang Technological University.

References

- BLAIS, F. 2004. Review of 20 years of range sensor development. *Journal of Electronic Imaging* 13, 1, 231–240.
- CARR, J., BEATSON, R., CHERRIE, J., MITCHELL, T., FRIGHT, W., MCCALLUM, B., AND EVANS, T. 2001. Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of ACM SIGGRAPH 2001*, ACM Press, 67–76.
- CELNIKER, G., AND GOSSARD, D. 1991. Deformable curve and surface finite elements for free-form shape design. In *Computer Graphics (Proceedings of ACM SIGGRAPH 91)*, vol. 25, 257–265.
- CHEN, X., AND SCHMITT, F. 1994. Surface modelling of range data by constrained triangulation. *Computer-Aided Design* 26, 8, 632–645.
- CHIVATE, P., AND JABLOKOW, A. 1995. Review of surface representation and fitting for reverse engineering. *Computer Integrated Manufacturing Systems* 8, 3, 193–204.
- CHOI, B., CHUNG, Y., AND PARK, J. 1997. Applications and extension of z-map model. In *Proceedings of the 3rd Pacific Conference on Computer Graphics & Applications, Seoul*, 363–381.
- FORSEY, D., AND BARTELS, R. 1995. Surface fitting with hierarchical splines. *ACM Trans. Graphics* 14, 2, 134–161.
- FRANKE, R., AND NIELSON, G. 1991. Scattered data interpolation and applications: a tutorial and survey. In *Geometric Modelling: Methods and Their Application*, H. Hagen and D. Roller, Eds. Berlin: Springer-Verlag, 131–160.
- HOPPE, H., DE ROSE, T., DUCHAMP, T., McDONALD, J., AND STUETZLE, W. 1992. Surface reconstruction from unorganized points. In *Computer Graphics (Proceedings of ACM SIGGRAPH 92)*, vol. 26, 71–78.
- JANG, D., KIM, K., AND JEONG, J. 1999. Adaptive Gregory patch approximation to z-map data. *Int J Adv. Manuf. Technol* 15.
- KRISHNAMURTHY, V., AND LEVOY, M. 1996. Fitting smooth surfaces to dense polygon meshes. In *Proceedings of ACM SIGGRAPH 96*, ACM Press, 313–324.

- LEE, S., WOLBERG, G., AND SHIN, S. 1997. Scattered data interpolation with multilevel B-splines. *IEEE Trans. Visualization and Computer Graphics* 3, 3, 228–244.
- MURAKI, S. 1991. Volumetric shape description of range data using blobby model. In *Computer Graphics (Proceedings of ACM SIGGRAPH 91)*, vol. 25, 227–235.
- PARK, H., AND KIM, K. 1995. An adaptive method for smooth surface approximation to scattered 3d points. *Computer-Aided Design* 27, 12, 929–939.
- SAKAR, B., AND MENQ, C. 1991. Smooth surface approximation and reverse engineering. *Computer-Aided Design* 23, 9, 623–628.
- SCHMITT, F., BARSKY, B., AND DU, W. 1986. An adaptive subdivision method for surface-fitting from sampled data. In *Computer Graphics (Proceedings of ACM SIGGRAPH 86)*, 179–188.
- SEDERBERG, T. W., ZHENG, J., BAKENOV, A., AND NASRI, A. H. 2003. T-splines and t-nurccs. *ACM Trans. Graph. (ACM SIGGRAPH 2003)* 22, 3, 477–484.
- SEDERBERG, T. W., CARDON, D. L., FINNIGAN, G. T., NORTH, N. S., ZHENG, J., AND LYCHE, T. 2004. T-spline simplification and local refinement. *ACM Trans. Graph. (ACM SIGGRAPH 2004)* 23, 3, 276–283.
- SOLINA, F., AND BAJCSY, R. 1990. Recovery of parametric models from range images: The case for superquadrics with global deformations. *IEEE Trans. PAMI* 12, 2, 131–147.