Chapter 13
Surface Reconstruction from Contour Lines or LIDAR elevations by Least Squared-error Approximation using Tensor-Product Cubic B-splines

Shyamalee Mukherji

Abstract

We consider, in this paper, the problem of reconstructing the surface from contour lines of a topographic map. We reconstruct the surface by approximating the elevations, as specified by the contour lines, by tensor-product cubic B-splines using the least squared-error criterion. The resulting surface is both accurate and smooth and is free from the terracing artifacts that occur when thin-plate splines are used to reconstruct the surface.

The approximating surface, $S(x,y)$, is a linear combination of tensor-product cubic B-splines. We denote the second-order partial derivatives of S by S_{xx}, S_{xy} and S_{yy}. Let h_k be the elevations at the points (x_k,y_k) on the contours. S is found by minimising the sum of the squared-errors $\{S(x_k,y_k)-h_k\}^2$ and the quantity $\int\int S_{xx}^2(x,y)+2S_{xy}^2(x,y)+S_{yy}^2(x,y)\,dy\,dx$, the latter weighted by a constant λ.

Thus, the coefficients of a small number of tensor-product cubic B-splines define the reconstructed surface. Also, since tensor-product cubic B-splines are non-zero only for four knot-intervals in the x-direction and y-direction, the elevation at any point can be found in constant time and a grid DEM can be generated from the coefficients of the B-splines in time linear in the size of the grid.

Centre of Studies in Resources Engineering
Indian Institute of Technology, Bombay
P.O.: Powai - I.I.T.
Mumbai - 400 076
INDIA.
shyamali@csre.iitb.ac.in

13.1 Introduction

The contour lines of a topographic map are sometimes the only source of information that we have about the terrain elevations in a region. Tensor-product B-splines have been successfully used to model a wide variety of surfaces in various fields, e.g., turbulence simulation, regional gravity field approximation, buckling of shells under loads etc.. In this paper, we reconstruct the surface from contour lines by approximating the elevations by tensor-product cubic B-splines using the least squared-error criterion.

Some solutions that have been proposed for the problem of reconstructing the surface from contour lines are given in Section 2. Then, the least squared-error approximation technique is described in Section 3. The results obtained by approximating the elevation data, derived from the contours, by tensor-product cubic B-splines are given in Section 4. In Section 5, we show that the elevation of the reconstructed surface can be computed efficiently and also give an efficient way of computing the DEM. In Section 6, we reconstruct surfaces from LIDAR elevations using the same technique of least squared-error approximation by tensor-product B-splines. We conclude the paper in Section 7, with a summary of the advantages of reconstructing the surface using B-splines.

13.2 Solutions Proposed in the Literature

Thin-plate splines have been used to reconstruct the surface from contour lines but the computing time of the algorithms and terracing artifacts in the reconstructed surface are major drawbacks of these methods [1].

The most accurate approach, for reconstruction of the surface, is to interpolate the elevations using Hardy's multiquadrics [2, 3]. The reconstructed surface is a linear combination of multiquadrics, each centered on a data point, and a polynomial of degree one. The coefficients of the multiquadrics and the polynomial are determined by solving a system of linear equations whose size is the number of data points plus three. Since the multiquadrics have global support, the coefficient matrix of the system of equations is a full matrix and a direct solution of the system is not acceptable for large data sets with thousands of points. The parameters that define the reconstructed surface are the centers of the multiquadrics along with the coefficients of the multiquadrics, and the coefficients of the polynomial. Therefore, the number of parameters required is thrice the number of data points plus three. Also, all the multiquadrics contribute to the elevation of the surface at any point.

To alleviate these problems, Pouderoux et al. proposed a scheme for fast reconstruction of a C^1-continuous surface [4]. The scheme also allows more control over the numerical stability of the solution. In this scheme, the global domain of interest is sub-divided into smaller overlapping sub-domains. The

number of coefficients that define the reconstructed surface goes up as the sub-divisions overlap. The number of multiquadrics that contribute to the elevation (of the surface) at any point is greatly reduced but is still of the order of hundreds.

Franklin proposed a computation and storage-intensive algorithm for reconstructing the surface from contour lines [5].

Goncalves et al. used piecewise cubics to approximate the elevation data [6]. The piecewise cubics he used, however, are C^1-continuous.

Dakowicz et al. solved the problem of 'flat triangles' in TINs by inserting skeleton points in the TIN and assigning them elevations [7]. Various interpolation techniques were then used to interpolate the thus enriched contour elevation data and were compared. Slight breaks in slope at contour lines or oscillations are the artifacts that remain in the best surfaces obtained.

In this paper, we choose to approximate elevation data using cubic B-splines as the natural cubic spline is the solution to the least squared-error approximation problem in one dimension and the B-spline is smooth (C^2-continuous). We find that approximating contour elevation data by tensor-product cubic B-splines results in a smooth surface. The solution is numerically stable. The reconstructed surface is defined by the coefficients of a small number of tensor-product B-splines. The number of B-splines required has been observed to be half the number of parameters that would be required to specify a surface that has been reconstructed by multiquadrics without applying Pouderoux's sub-division scheme. Also, exactly 16 tensor-product B-splines contribute to the elevation of the surface at a point. So, the elevation at any point can be found in real-time.

13.3 Least Squared-error Approximation by Tensor-product Cubic B-splines [8]

A cubic B-spline, $B(x)$, with uniform knot-spacing Δ and centered at x_i is given by $h(|x-x_i|/\Delta)$ where

$$h(p) = \begin{cases} 3p^3/6 - p^2 + 4/6 & 0 \le p < 1 \\ -p^3/6 + p^2 - 2p + 8/6 & 1 \le p < 2 \\ 0 & 2 \le p \end{cases}$$

A tensor-product cubic B-spline, centered at a point (x_1, y_1), is the product of a cubic B-spline $B(x)$ centered at x_1 and a cubic B-spline $B(y)$ centered at y_1.

Let $B_i(x)$ and $C_j(y)$ be cubic B-splines along the x and y-directions, respectively. Let us take n_x B-splines along the x-direction and n_y B-splines along the y-direction. Then, the spline surface $S(x,y)$ which is to be constructed is

$$S(x,y) = \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} c_{ij} B_i(x) C_j(y)$$

Let N be the number of points, p_k, at which the elevations, h_k, are known on the contour lines.

A smoothing term is added to the squared-error that is to be minimized and the function to be minimized is

$$F(c) = \sum_{k=1}^{N} \{S(p_k) - h_k\}^2 + \lambda J(c), \tag{13.1}$$

where λ is a positive constant and

$$J(c) = \int_{a_1}^{b_1} \int_{a_2}^{b_2} S_{xx}^2 + 2S_{xy}^2 + S_{yy}^2 \, dy \, dx$$

where S_{xx}, S_{xy} and S_{yy} are the second-order partial derivatives of $S(x,y)$ and $[a_1,b_1] * [a_2,b_2]$ is the domain of S.

This integral can be expressed as

$$\sum_{i=1}^{n_x} \sum_{j=1}^{n_y} \sum_{r=1}^{n_x} \sum_{s=1}^{n_y} E_{ijrs} c_{ij} c_{rs},$$

where

$$E_{ijrs} = A_{ijrs} + 2B_{ijrs} + C_{ijrs}$$

and

$$A_{ijrs} = \int_{a_1}^{b_1} B_i''(x) B_r''(x) \, dx \int_{a_2}^{b_2} C_j(y) C_s(y) \, dy,$$

$$B_{ijrs} = \int_{a_1}^{b_1} B_i'(x) B_r'(x) \, dx \int_{a_2}^{b_2} C_j'(y) C_s'(y) \, dy,$$

$$C_{ijrs} = \int_{a_1}^{b_1} B_i(x) B_r(x) \, dx \int_{a_2}^{b_2} C_j''(y) C_s''(y) \, dy.$$

A minimum of $F(c)$ must occur at a point c where all partial derivatives are zero.

Let

$$J(c) = c^T E c,$$

where E is a square matrix of dimension $n = n_x * n_y$ whose elements are

$$E_{(j-1)n_x+i,(s-1)n_x+r} = E_{ijrs}$$

for $i,r = 1,.....,n_x$ and $j,s = 1,.....,n_y$.

By differentiating (1), we get

$$(B^T B + \lambda E)c = B^T h, \tag{13.2}$$

where $h = (h_1,.....,h_N)^T$ and B is the $N \times n$ matrix

$$\begin{pmatrix} B_1(x_1)C_1(y_1) & B_2(x_1)C_1(y_1) & \ldots & B_{n_x}(x_1)C_{n_y}(y_1) \\ . & . & . & . \\ . & . & . & . \\ B_1(x_N)C_1(y_N) & B_2(x_N)C_1(y_N) & \ldots & B_{n_x}(x_N)C_{n_y}(y_N) \end{pmatrix}$$

where $p_k = (x_k, y_k)$.

Then, the solution to (1) is the solution c to (2).

The matrix $G = B^T B + \lambda E$ is positive semi-definite. The matrix is strictly positive definite if the only solution to $c^T Gc = 0$ is $c = 0$.

First observe that

$$c^T Ec = J(c) = 0$$

implies that S must be a linear polynomial $a + bx + cy$. Second, observe that

$$c^T B^T Bc = ||Bc||^2 = 0$$

implies that $S(p_k) = 0$ for all $k = 1, ..., N$. Thus, we have that $c^T Gc = 0$ implies that S is a linear polynomial which is 0 at every point p_k. Clearly then, if there are at least 3 points p_k which do not lie on a straight line, S would have to be 0 and all the coefficients c_{ij} would have to be 0. Since the points p_k can never all be collinear, we deduce that G is indeed nonsingular and the minimizer c of (2) is unique.

13.4 Results

The contour lines in Fig. 1 correspond to elevations ranging from 220 metres at the bottom of the map to 660 metres at the top of the map; at intervals of 20 metres. The lines were digitized with a planimetric accuracy of 2.28 metres. $14 * 19$ tensor-product cubic B-splines (corresponding to 14 B-splines along the x-direction and 19 B-splines along the y-direction) and a value of 0.0001 for λ were used to obtain the reconstructed spline surface in Fig. 2. The r.m.s. error (between the elevations of the reconstructed surface at the points on the digitized contour lines and the contour line elevations) achieved is 1.89 metres. The surface is smooth with minor artifacts at a few places at the edges of the surface. The surface obtained by interpolation using Hardy's multiquadrics is shown in Fig. 3 for comparison. The difference surface is shown in Fig. 4.

$32 * 30$ tensor-product cubic B-splines and a value of 0.01 for λ were used to achieve an r.m.s. error of 2.01 metres for the contour lines in Fig. 5. The reconstructed surface is shown in Fig. 6. The surface obtained by interpolation using Hardy's multiquadrics is shown in Fig. 7. The difference surface is shown in Fig. 8.

Thus, the reconstructed surface is defined by a small number of B-splines. The surface is free from the terracing artifacts that occur when thin-plate splines are used to reconstruct the surface.

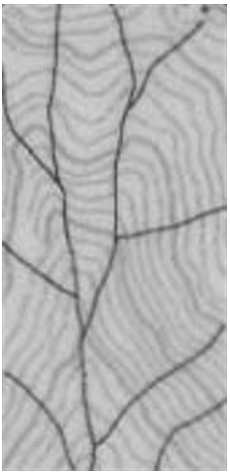

Fig. 13.1 Contour lines.

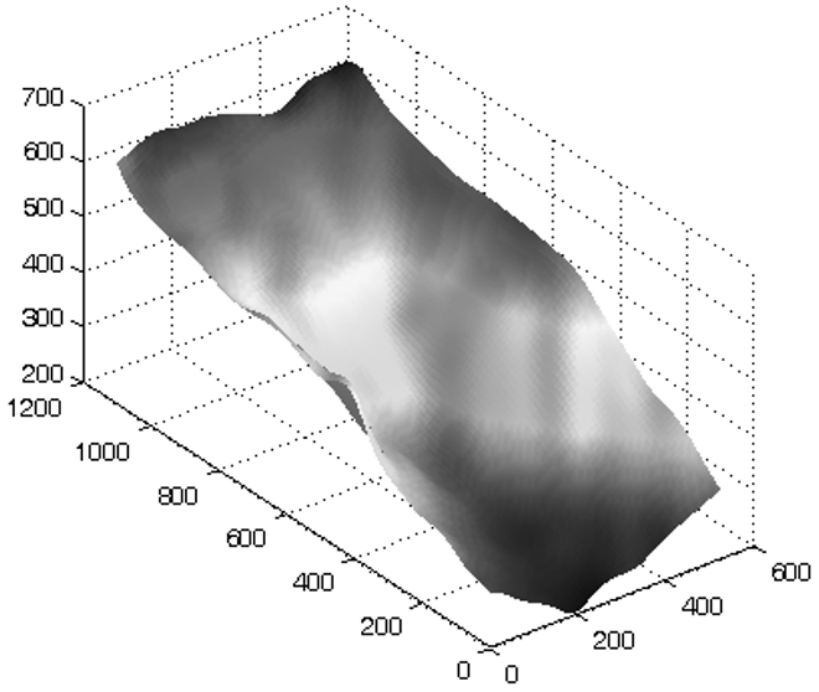

Fig. 13.2 The reconstructed surface.

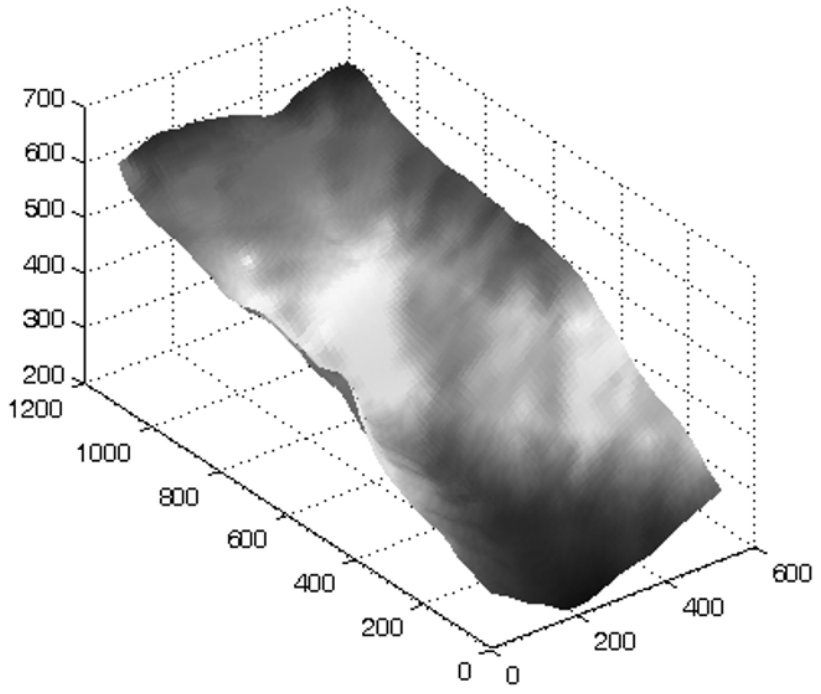

Fig. 13.3 Surface obtained by interpolation using Hardy's multiquadrics.

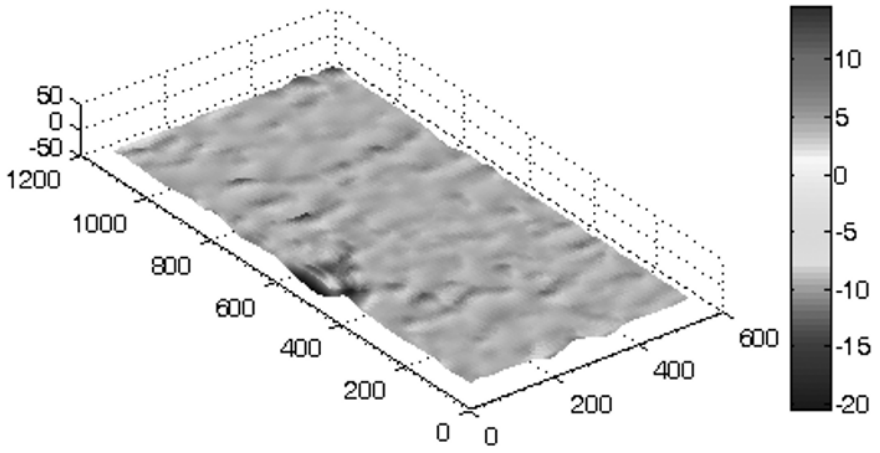

Fig. 13.4 The difference surface.

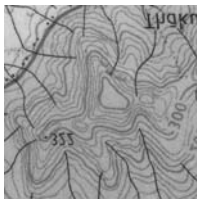

Fig. 13.5 Contour lines.

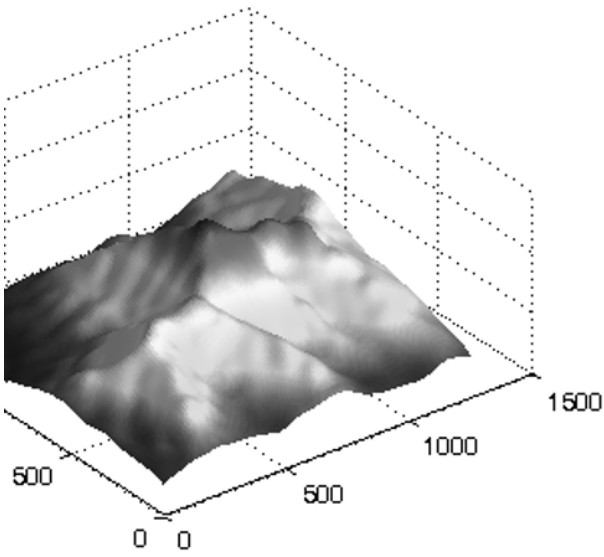

Fig. 13.6 The reconstructed surface for the contour lines in Fig. 5.

13.5 Computing the DEM

In this Section, we show that the elevation of the reconstructed surface at a point can be computed efficiently. We then describe an efficient way of generating the DEM with low memory requirements.

Since tensor-product cubic B-splines are non-zero only for four knot-intervals in the x-direction and the y-direction, exactly 16 tensor-product cubic B-splines contribute to the elevation at a point.

Let us assume that the B-splines that are non-zero at a point $(x1, y1)$ are $B_k(x)$, $B_{k+1}(x)$, $B_{k+2}(x)$, $B_{k+3}(x)$, $C_l(y)$, $C_{l+1}(y)$, $C_{l+2}(y)$ and $C_{l+3}(y)$. The elevation at $(x1, y1)$ is found by evaluating the right-hand side of Eqn. 2 at $(x1, y1)$ as follows:

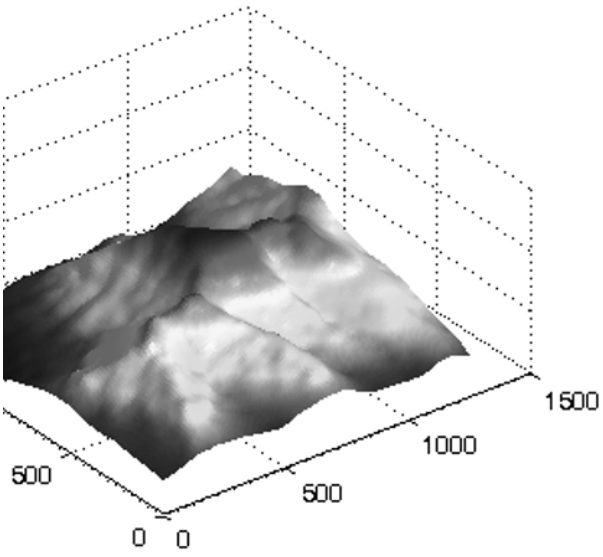

Fig. 13.7 Surface obtained by interpolation using Hardy's multiquadrics.

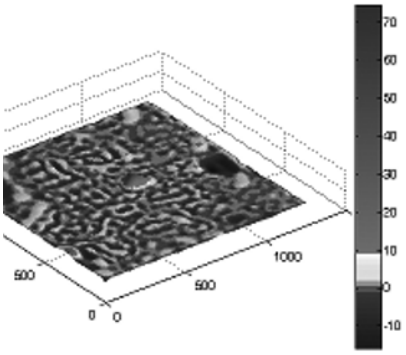

Fig. 13.8 The difference surface.

We first evaluate the cubic polynomials $P_j(x) = \sum_{i=k}^{k+3} c_{ij} B_i(x),\ j = l, \ldots, l+3$ at $x = x1$. To do this, we first find $p = |x1 - x_{k+1}|/\Delta_x$ where x_{k+1} is the center of the B-spline $B_{k+1}(x)$ and Δ_x is the knot-interval for the $B_i(x)$. $P_j(x1)$ is, then,

$$c_{kj}(-p^3 + 3p^2 - 15p + 1) + c_{k+1,j}(3p^3 - 6p^2 + 4)$$
$$+c_{k+2,j}(-3p^3 + 3p^2 + 3p + 1) + c_{k+3,j}p^3 \tag{13.3}$$

The quantities in the brackets require 17 operations {where an operation is either an addition (subtraction) or a multiplication}. So, finding the four $P_j(x1)$ requires $17 + 4 * 7 = 45$ operations.

Then, we find $q = |y1 - y_{l+1}|/\Delta_y$ where y_{l+1} is the center of the B-spline $C_{l+1}(y)$ and Δ_y is the knot-interval for the $C_j(y)$. The elevation at $(x1, y1)$, which is $\sum_{j=l}^{l+3} P_j(x1) C_j(y1)$, can be expressed in a form that is analogous to the expression in (3) above. A rearrangement of the terms yields

$$q^3\{-P_l(x1) + 3(P_{l+1}(x1) - P_{l+2}(x1)) + P_{l+3}(x1)\}$$
$$+3q^2\{P_l(x1) + P_{l+2}(x1) - 2P_{l+1}(x1)\} + 3q\{-5P_l(x1) + P_{l+2}(x1)\}$$
$$+P_l(x1) + P_{l+2}(x1) + 4P_{l+1}(x1) \tag{13.4}$$

the computation of which requires 21 operations.

Thus, the elevation at any point can be found with $45 + 21 = 66$ operations, or, in constant time. (In contrast, in Pouderoux's sub-division scheme, a small multiple of 800 operations would be required.)

When generating a DEM, we start with the distinct x co-ordinates, of the points of the DEM grid (Fig. 9), which lie between x_2 and x_3 (x_1 and x_{n_x} lie at least Δ_x to the left of and Δ_x to the right of the first and last data points, respectively, in the horizontal direction; and there are no data points in the intervals $[x_1, x_2]$ and $[x_{n_x-1}, x_{n_x}]$), compute the respective $p = |x - x_2|/\Delta_x$ and compute and store the quantities in the brackets in expression (3), for these x co-ordinates. This takes $17N_x$ operations where N_x is the number of distinct x co-ordinates of the grid that lie between x_2 and x_3. We then compute and store the four $P_j(x1), j = 1, \ldots, 4$ for each of these x co-ordinates. This takes $N_x * 4 * 7 = 28N_x$ operations.

We then take the y co-ordinates of the DEM grid that lie between y_2 and y_3 and find the corresponding $q = |y - y_2|/\Delta_y$. The elevation at the grid-points, $(x1, y1)$, whose x co-ordinates lie between x_2 and x_3 and y co-ordinates lie between y_2 and y_3, is $\sum_{j=1}^{4} P_j(x1) C_j(y1) = P_1(x1)(-q^3 + 3q^2 - 15q + 1) + P_2(x1)(3q^3 - 6q^2 + 4) + P_3(x1)(-3q^3 + 3q^2 + 3q + 1) + P_4(x1)q^3$. Thus, the computation of the elevations at these grid-points takes a total of $17N_y + N_y * N_x * 7$ operations where N_y is the number of distinct y co-ordinates of the grid that lie between y_2 and y_3.

Next, we compute $P_5(x1)$ and use $P_j(x1), j = 2, \ldots, 5$ to find the elevations at the grid-points, whose x co-ordinates lie between x_2 and x_3 and y co-ordinates

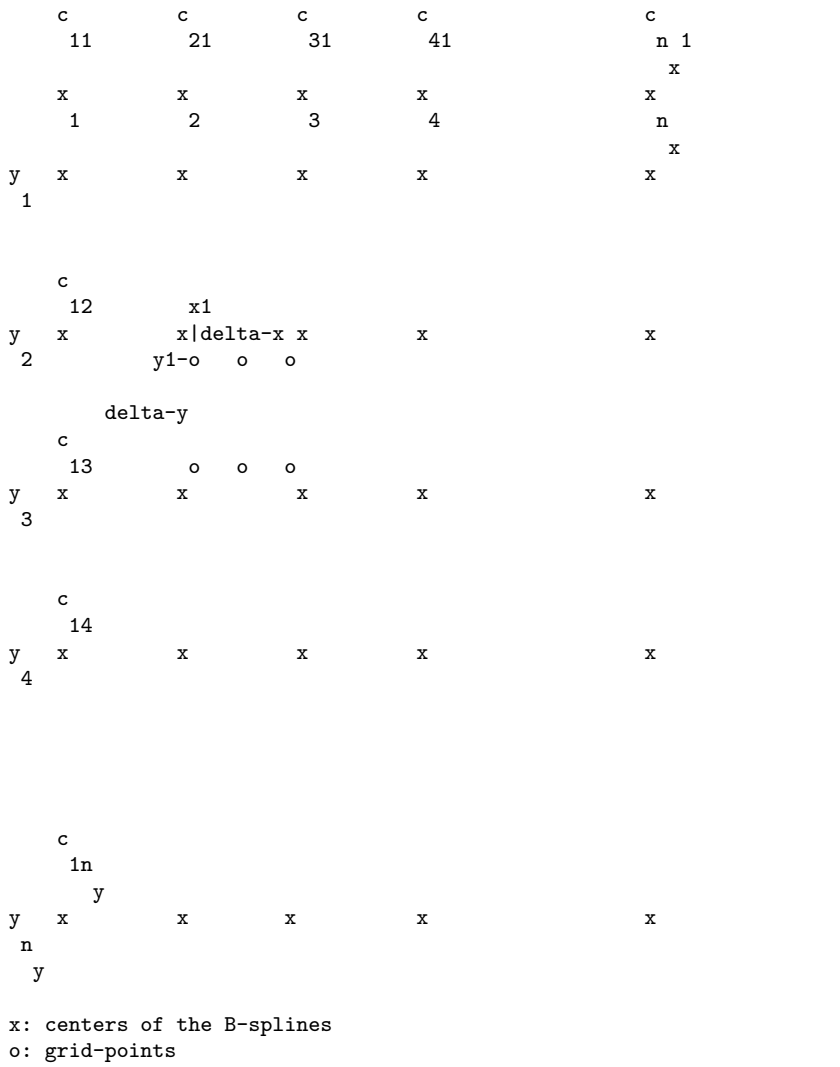

Fig. 13.9 Computing the DEM

lie between y_3 and y_4. We proceed down the grid in this manner, computing the elevations at the grid-points, till we reach the bottom of the grid. Then we return to the top of the grid and repeat the entire procedure starting, this time, with the distinct x co-ordinates of the grid-points between x_3 and x_4. This process continues till the lower right corner of the grid is reached.

Thus, the computation of the elevations at the grid-points requires $17N + 7Nn_y + 17M(n_x - 3) + 7MN = 7MN + 17M(n_x - 3) + 7Nn_y + 17N$ operations for an $M * N$ grid. n_x and n_y can be atmost N and M. So, the computation time

is linear in the size of the grid and the constant is small, viz., 31. This is not the case with multiquadric-based DEMs wherein all multiquadrics contribute to the elevation at every grid-point.

13.6 Approximating LIDAR elevation data

We rotated the LIDAR scan-lines so that they are vertical as in Fig. 10. The surface obtained by approximating the elevations by tensor-product quadratic B-splines is shown in Fig. 11. This is a rural area scanned from an altitude of 1350 metres with a point spacing of 1.1 metres. The r.m.s. deviation of the surface from the LIDAR elevations is 1.8 cm.. Brovelli et al. approximates the LIDAR elevation data by bilinear splines [9]. The corresponding surface is shown in Fig. 12.

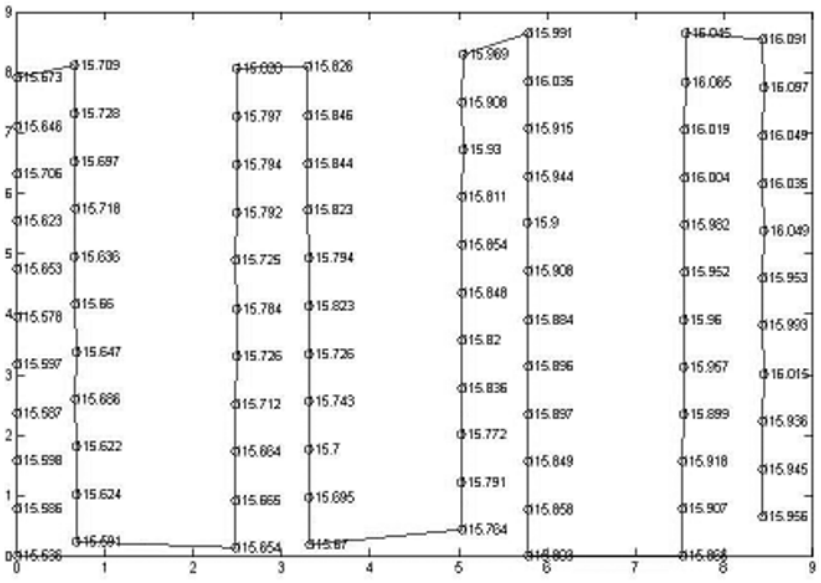

Fig. 13.10 LIDAR elevations. (The values shown are in excess of 100 metres.)

LIDAR elevation data with gaps (Fig. 13) (rural area again) are approximated well by tensor-product cubic B-splines as is shown in Fig. 14. The value of lambda used is 0.000001 and the r.m.s. deviation of the surface from the LIDAR elevations is 2.6 cm..

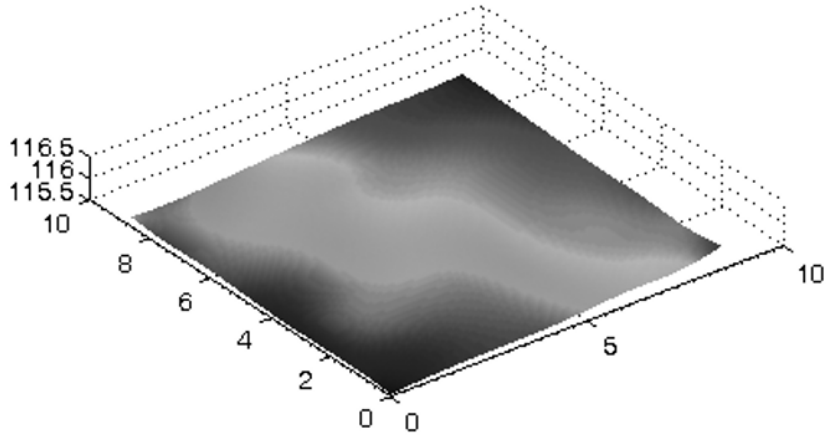

Fig. 13.11 Surface approximating LIDAR data.

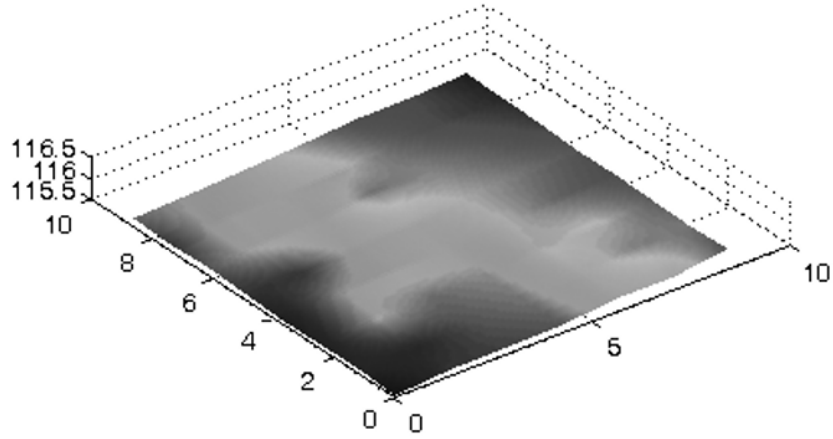

Fig. 13.12 Approximation by bilinear splines.

13.7 Conclusions

We conclude, from the results in Section 4, that tensor-product cubic B-splines lead to a good reconstruction. The reconstructed surface is also free from the terracing artifacts that occur when thin-plate splines are used to reconstruct the surface. The smoothness of the surface results from the inherently smooth nature of cubic B-splines, which are C^2-continuous. An ad-

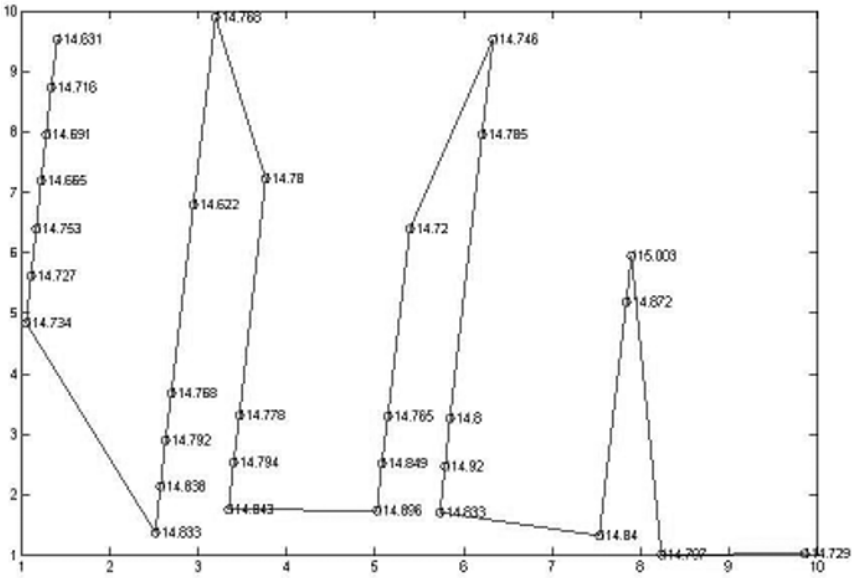

Fig. 13.13 LIDAR elevations with gaps. (The values shown are in excess of 100 metres.)

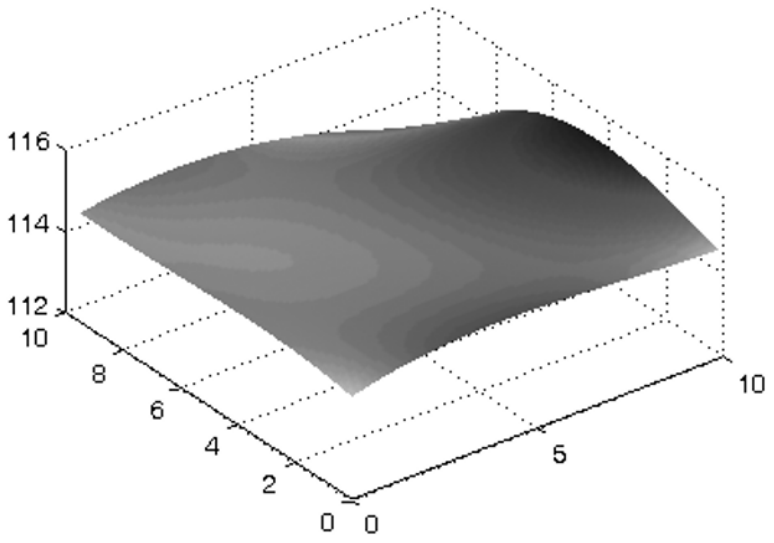

Fig. 13.14 Surface approximating LIDAR data with gaps.

vantage of this method of reconstructing the surface is that the coefficients of a small number of tensor-product cubic B-splines define the surface.

Tensor-product cubic B-splines are non-zero only for four knot-intervals in the x-direction and y-direction. Therefore, exactly 16 tensor-product B-splines contribute to the elevation at a point. So, the elevation at any point can be found in constant time and a grid DEM can be generated from the coefficients of the B-splines in time linear in the size of the grid.

13.8 Acknowledgement

This work was supported by the Indian Space Research Organization.

References

[1] Gousie MB (1998) Contours to digital elevation models: Grid-based surface reconstruction methods. PhD Thesis, Rensselaer Polytechnic Institute, Troy, New York
[2] Franke R (1982) Scattered data interpolation: tests of some methods. Math Comp 38(157):181-200
[3] Hardy RL (1971) Multiquadric equations of topography and other irregular surfaces. J Geophys Res 76:1905-1915
[4] Pouderoux J, Gonzato JC, Tobor I, Guitton P (2004) Adaptive hierarchical RBF interpolation for creating smooth digital elevation models. Proc 12th Ann ACM Intl Workshop GIS 2004, Washington, DC, USA, 232-240
[5] Franklin WR (2000) Applications of analytical cartography. Carto & GIS 27(3):225-237
[6] Goncalves G, Julien P, Riazanoff S, Cervelle B (2002) Preserving cartographic quality in DTM interpolation from contour lines. ISPRS J Photogram & Remote Sens 56:210-220
[7] Dakowicz M, Gold CM (2003) Extracting meaningful slopes from terrain contours. Intl J Comput Geom & Applns 13(4):339-357.
[8] Floater MS (2000) Meshless parameterization and B-spline surface approximation. In: Cipolla R, Martin R (eds) The mathematics of surfaces IX. Springer, Berlin Heidelberg NewYork
[9] Brovelli MA, Cannata M, Longoni UM (2004) LIDAR data filtering and DTM interpolation within GRASS. Trans in GIS 8(2):155-174