A Practical Image Retouching Method

Vladimir Savchenko †, Nikita Kojekine ‡, Hiroshi Unno †

† Faculty of Computer and Information Sciences, Hosei University
{vsavchen,n00k1113}@k.hosei.ac.jp

‡ Faculty of Engineering, Tokyo Institute of Technology
karlson@stu.mech.titech.ac.jp

Abstract

In this paper, we present a novel fast algorithm for image retouching. A space-mapping technique is used to transform a missing (or damaged) part of a surface into a different shape in a continuous manner. Experimental results are included to demonstrate the feasibility of our approach. The proposed approach shows the obvious relationship between the surface retouching problem and image inpainting. We consider shape transformation as a general type of operation for restoring missing data, and attempt to approach the well-known problem of "fulfillment" of damaged or missing areas from a single point of view, namely, that of the space mapping technique.
***Keywords**: Radial basis functions, space mapping, image inpainting*

1. Introduction

Many recent studies have focused on exploring shape transformation as a basic operation in computer graphics (CG) and computer-aided geometric design (CAGD). The operation involves transforming a given geometric shape into another in a continuous manner. The approach proposed here for reconstruction of damaged image areas - image inpainting problem - originates from the obvious relationship between the 2D inpainting problem and 3D surface retouching (or reconstruction of missing 3D data). For references concerning to the image retouching, see the pioneering work of Bertalmio et al. [1], which describes an image-inpainting (non-texture) algorithm based on partial differential equations. The technique to fill a given region with a selected texture was presented by Hirani and Totsuka in [2]. See also the recent studies [3], [4], and [5].

In this paper we show that compactly supported radial-basis functions (CSRBFs) [6] offer a mechanism for obtaining extrapolated points of a damaged image, and, in fact, exhibit nice restoration results in image-retouching examples. We consider 3D shape transformation as a general type of operation for restoring missing data, and attempt to approach the well-known problem of "fulfillment" of damaged areas from a single point of view, namely, that of a space mapping technique. The main idea of the approach is to use the user-defined portions of the input damaged image to be retouched. The algorithm treats the input image as three separate channels (R, G, and B) and fills in areas to be inpainted. Actually, it is an extrapolation problem in which information from outside a masked area is propagating. It is understood, that the 3D space-mapping technique can be applied to image inpainting if three separate R, G, B channels are used and they are represented as elevation data in the form $z = f(x, y)$, where the z coordinate is used as one channel. In practice, for 3D case applications, the technique requires that a masked surface represented by polyhedrons be a closed, oriented manifold embedded in 3-space and have the property that, around every one of its points, there exists a neighborhood that is homeomorphic to a plane. That is, we can deform the surface locally into the plane without tearing it or identifying separate points with each other.

We propose a fast practical algorithm for image retouching, which we believe can generate further attention from the CG community. Let us point out here that we clearly understand that every technique has some strengths and some shortcomings. The main shortcoming of the proposed technique is that the technique will fail with retouching of synthetic examples (curve restoration) such as that in Figure 4 of reference [2]. Nevertheless, in practical applications dealing with real images, the possibility to use different levels or separated fragments of images that was realized in an Adobe Photoshop plug-in (developed by our group and available for download from Web [7]) allows us to avoid intermixing different colors for abruptly separated color areas. In practice, it gives satisfactory results. In addition, although we demonstrate the applicability of the proposed technique by restoration of ”elevation data,” this technique can be applied in 3D space and can serve for sculpting or editing a surface according to the user demands.

The rest of the paper is organized as follows. The next section gives a short overview of shape transformation techniques and studies related to the surface retouching problem. We discuss the notion of a 3D warping technique and software algorithm in Section 3. Application examples of image inpainting are presented in Section 4. Section 5 concludes the paper.

2. Related works

Mappings can be controlled by numerical parameters of predefined functions, by control points, and by differential equations. An overview of all transformation techniques is beyond the scope of this paper; here we present a short overview of papers concerning the problem.

Shape transformation is a useful tool for many applications such as computer animation, CAGD, forensic identification. Chen et al. [8] and Skaria et al. [9] present a very good overview of existing methods for shape reconstruction and modification. For more references, see [10].

A vast literature is devoted to the subject of scattered data interpolation. This can be used for space mapping and, if applied to some point set in the space, it changes that set into a different one. In spite of a flurry of activity in the field of scattered data reconstruction and interpolation, this problem remains difficult and computationally expensive. A very good overview of related studies, problems, and limitations could be found in Lee et al. [11], which addresses these problems and introduces a very fast algorithm for constructing C^2-continuous interpolation functions.

Shape reconstruction from given points can be thought of as a special case of transformation. One approach is to use methods of scattered data interpolation based on the minimum-energy properties [12], [13], [14]. These methods are widely discussed in the literature (see also [15] and [16]).

The benefits of modeling with the help of RBFs have been recognized in many studies. To the best of our knowledge, the first publications on using discrete 2D landmark points were that of Bookstein [17], [18]. RBFs were adapted for computer animation [19], medical applications [20], [17], and surface reconstruction [21], [22]. However, the required computational work is proportional to the number of grid nodes and the number of scattered data points. Special methods for reducing the processing time were developed for thin plate splines, and were discussed in [23], [24], and [25]. In spite of significant progress in the field of implementing RBFs and CSRBFs [26], [27] for reconstruction purposes, it is still an open question whether it is possible to handle realistic amounts of data in real time. We suppose that they are suitable for sufficiently moderate 3D data sets; for instance execution time is about 300 seconds for 36000 points, without time expenses for surface extraction as it was reported in [24]. Nevertheless, they possess many features that make them very attractive for CAGD applications dealing with modification of geometric objects.

3. Algorithm

For three dimension arbitrary area Ω, the thin-plate interpolation is the variational solution that defines a linear operator **T** when using the following minimum condition:

$$\int_\Omega \sum_{|\alpha|=m} m!/\alpha!(D^\alpha f)^2 d\Omega \to min, \tag{1}$$

where m is a parameter of the variational functional and α is a multi-index. It is equivalent to using the radial basis functions $\phi(r) = r^1$ or r^3 for $m = 2$ and 3 respectively, where r is a Euclidean distance between two points. Since the function $\phi(r)$ is not compactly supported, the corresponding system of linear algebraic equations (SLAE) is not sparse or bounded. Storing the lower triangle matrix requires $O(N^2)$ real numbers, and the computational complexity of a matrix factorization is $O(N^3)$. Thus, the amount of computation becomes significant, even for a moderate number of points.

Wendland in [6] constructed a new class of positive definite and compactly supported radial functions for 1D, 3D and 5D spaces of the form

$$\phi(r) = \begin{cases} P(r) & 0 \leq r < 1 \\ 0 & r > 1 \end{cases} \tag{2}$$

whose radius of support is equal to 1. $\phi(r) = (1-r)^2$, which is an interpolated function that supports only C^0 continuity, is used. However, other functions that support a higher continuity can be applied. An investigation [33] of the smoothness of this family of polynomial basis functions shows that each member $\phi(r)$ possesses an even number of continuous derivatives.

The volume spline $f(P)$ having values h_i at N points P_i is the function

$$f(P) = \sum_{j=1}^{N} \lambda_j \phi(|P - P_j|) + p(P), \tag{3}$$

where $p = v_0 + v_1 x + v_2 y + v_3 z$ is a degree one polynomial. To solve for the weights λ_j we have to satisfy the constraints h_i by substituting the right part of equation (3), which gives

$$h_i = \sum_{j=1}^{N} \lambda_j \phi(|P_i - P_j|) + p(P_i), \tag{4}$$

Solving for the weights λ_j and v_0, v_1, v_2, v_3 it follows that in the most common case there is a doubly bordered matrix **T**, which consist of three blocks, square sub-matrices **A** and **D** of size $N \times N$ and 4×4 respectively, and **B**, which is not necessarily square and has the size $N \times 4$.

We employed an approach that uses displacements of N control points as a difference between an initial and final geometric form. The approach is illustrated by local deformations of data points homeomorphic to a plane; nevertheless, it is important to reiterate here that we formulate the retouching problem in terms of the space-mapping technique, where transformations in x, y, z directions are allowed. Also, heights h_i, are not necessarily arbitrary points of Euclidian space E^n. Our approach gives an attractive possibility of using function mapping for controlling local deformations by placing arbitrary control points inside or outside an initial implicitly defined object G, and they are assumed to belong to the surface of a modified object G^m. Thus, the control points define the deformation of G resulting in G^m. In practice, it leads to the construction of a new defining function and the extraction of a new isosurface as it was discussed in [21]. In the paper we have used an algebraic sum of a sphere and a spline based on RBFs to reconstruct 3D solids (even non-convex solids) from a given set of scattered surface points. CSRBFs in combination with function mapping were used for surface reconstruction as it is discussed in [27].

Interpolation of (x, y, z) points is implemented in $\mathbf{R}^3$ and defines a relationship between coordinates of points in the original and deformed objects. Landmarks situated in the mask area Ω^+ defined by the user (Figure 4(b) and 5(a), black areas) naturally do not belong to the area Ω^R to be retouched (Figure 5(a)). The difference between Ω^+ and Ω^R defines the landmark area Ω^L. The inverse mapping function that interpolates the z-heights and is needed to calculate reconstructed (destination) points z_i^d is given in the form

$$z_i^s = f(P) + z_i^d, \tag{5}$$

where the components of the volume splines $f(P)$ interpolating displacements of starting points z_i^s are used to calculate points that belong to the area Ω^R.

In practice, the reconstruction operation consists of the following steps: data sorting for the landmarks area Ω^L, construction of an SLAE, solution of the SLAE, and evaluation of the functions for a region to be retouched. In fact, while the solution of the system is the limiting step, construction of the matrix and evaluation of the functions may also be computationally expensive.

3.1. Sorting scattered data

Our first goal is to build a variable-depth octal tree data structure [34] from the original point data. Afterwards this octal tree is used to search for neighbors of any point among the given N points. The neighbors are points of a sphere of radius r whose origin is located at the given point.

To make the sub-matrix **A** a band diagonal, we need to re-enumerate the initial points in a special way. An efficient approach based on the use of variable-depth octal trees for space subdivision is proposed. It allows us to obtain the resulting matrix as a band-diagonal matrix that reduces the computational complexity.

We propose the following algorithm (presented in pseudo-code here):

```
//input_list            - input list of points
//output_list           - output list of points
//neighbors_ids_list - temporary list of integers
i:=0;
while (input_list.length >= 0) do
begin
   // add first element of input_list to output_list
   // remove first element from input_list
   output_list.add(input_list[0]);
   input_list.remove(0);
   while (i < output_list.length) do
   begin
      // find in input_list all neighbors of output_list[i] and
      // put their indices into neighbors_ids_list
      // this can be done with the help of octree
      neighbors_ids_list=FindNeighbors(output_list[i],input_list);
      for j:=0 to neighbors_ids_list.length do
      begin
         // add neighbor element of input_list to output_list
         // remove this element from input_list
         output_list.add(input_list[neighbors_list[j]]);
         input_list.remove(neighbors_ids_list[j]);
      end
   end
end
```

As a result of applying this algorithm a band-diagonal sub-matrix can be constructed as it is illustrated in Figure 1. To store the band-diagonal matrix **A**, we use a so-called profile form or a slightly modified Jennings envelope scheme [35]. An array can be used for diagonal elements; values of non-diagonal elements and corresponding indices of the first non-zero elements in the matrix lines are placed in two additional arrays.

3.2. SLAE solution

The attractiveness of using implicit methods such as conjugate gradient methods for large sparse systems has been well recognized in different applications. If T is positive

Figure 1. An example of typical sub-matrix A. Stucture of the matrix produced for the example shown in Figure 4.

definite and symmetric, the algorithm cannot break down, in theory [37]. Conjugate gradient methods work well for matrices that are well-conditioned. In practical applications, this restriction can limit the accuracy with which a solution can be obtained, and thus we prefer to use explicit SLAE solution methods for matrices stored in profile form. For a symmetric and positive definite matrix, a special factorization, called Cholesky decomposition, is about twice as fast as alternative methods for solving linear equations by Gaussian **LU** decomposition [36]. A combination of block Gauss solution and Cholesky decomposition was proposed by George and Liu in [37], and in our software tools we follow their proposal.

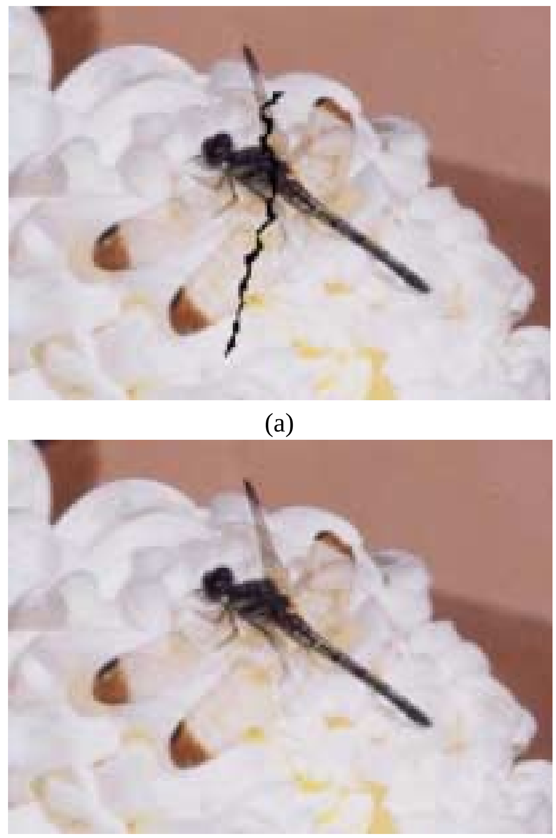
After breaking up one linear set into a triangular set of equations, these equations can be solved by forward substitution and back substitution three times if we have three right-hand sides.

4. Experiments

Floating point arithmetic is used; thus, before processing the input data we prefer to normalize them in order to reduce the number of possible truncation errors. To illustrate the applicability of the space-mapping technique to the image inpainting problem we show some examples.

The cost of inpainting is not linear to the size of the inpainted area; nevertheless, the inpainting time of the photograph of a dragonfly shown in Figure 2 obtained with our algorithm was 0.06 seconds on a 1-GHz Athlon PC. A synthetically produced black scratch is the reconstruction area Ω^R. Adobe Photoshop® GUI interface allows us to select automatically this area, after that a landmark area Ω^L as a slight extension of the Ω^R area (one or two pixels are sufficient) is automatically calculated [7].

For the example shown in Figure 3(b), Bertalmio et al. [1] reported an inpainting time (for one color channel) of approximately 7 minutes on a 300 MHz Pentium II PC. In our experiments, we noticed that it is possible to repaint small areas without noticeable loss of restoration quality. In our case, this means that the cost of inpainting is becoming nearly linear and it allows to see the results of repainting in an interactive way. Thus, for the example shown in 3(c), nearly all details are recovered in the restored image, obtained in less than one second in three-four steps. Table 1 shows the processing time for one-step reconstruction. The algorithm presented by Oliveira et al. [3] is two to three orders of magnitude faster than current methods based on the solution of partial differential equations. For the example shown in Figure 3(a), they reported an inpainting time of 0.49 sec on a 1-GHz Athlon PC. But the authors noted in [3] that their algorithm is suitable only for inpainting of small areas. Figure 4(c) demonstrates that our algorithm can repaint disconnected and sufficiently large areas (see the bottom right-hand black-and-white area).



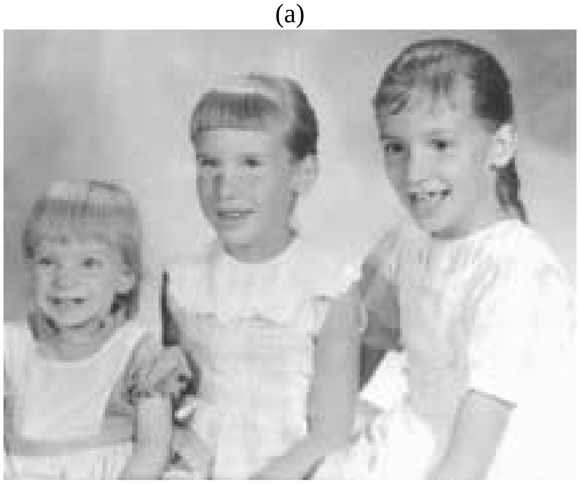
(a)

(b)

Figure 2. (a) Damaged picture of a dragonfly. (b) Result of restoration.

Hirani and Totsuka in [3] proposed an elegant fast algorithm for image retouching of textured images. In fact, their

(a)



(b)

(c)

Figure 3. (a) Old photograph (courtesy of Amy Miller of Photo-Medic). (b) Result produced with Bertalmio, Sapiro, Caselles, and Ballester's algorithm [1]. (c) Restored image obtained with our algorithm.

algorithm exploits a ”cloning” technique, where the user has to select an area, which is used for recovering a damaged area. After that frequency and spatial domain information is used to merge a damaged area with selected by the user ”cloning” area. Such approach allows getting excellent inpating results; see Figure 5(b). Nevertheless, it is a tedious work to repaint even a small scratch because the user has to select various ”cloning” areas as it is shown in paper [2]. Figures 5(c) and 6(b) show that using locally defined neighboring points from a landmark area Ω^L initially surrounding the scratch produces sufficiently good visual appearance of the retouched scratch. In this examples a simple strategy of using a sliding window as a local landmark area Ω^L is used and each consequent step uses data calculated on previous step. The mask Ω^+ is generated automatically as a contraction of the area Ω^R. The algorithm was implemented in Java, Householder QR matrix decomposition is used; an Adobe Photoshop plug-in realization of the proposed technique in C++ is now under construction.

5. Conclusions

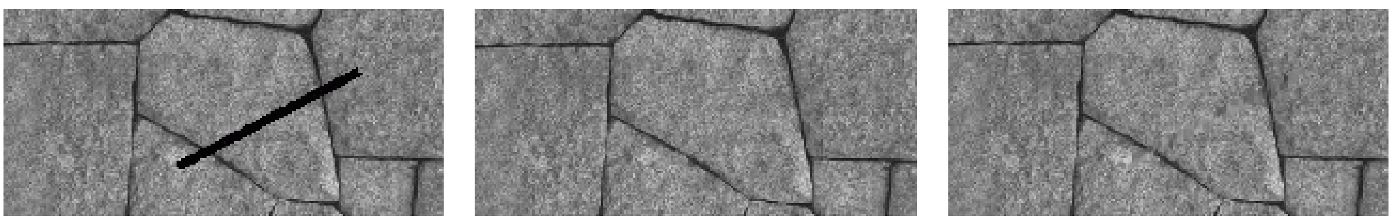
In this paper, we presented a simple and fast algorithm for image retouching. To achieve the objectives of simplicity and speed, we replace the reconstruction process by a space mapping technique, which allows reconstruction of disconnected areas. Our approach shows the obvious relationship between the surface- and image-retouching problems. The method has the property that the mask Ω^+ need not include exactly all the regions to be retouched; it allows Ω^+ to be refined interactively in many steps to preserve the algorithm's speed, and even to improve the quality of restoration.

We have found that CSRBFs produce good results in the sense of visual appearance, accuracy of restoration, and processing time. The algorithm is two to three orders of magnitude faster than methods based on partial differential equations. The work presented here does not discredit previous more image-oriented models based on level-set formulation. Moreover, our approach does not ensure a capability for automatic filling regardless of the shape and topology of the inpainting domain, as discussed in previous papers.

Proper selection of the radius of support is not critical for interactive image retouching. In practice, the user can estimate the size of the radius of support while defining a reconstruction area. Nevertheless, it is critical to achieve optimal computational results. Our plan is to find a reasonable way of selecting the radius of support. We are also going to use this approach to design an interactive system for 3D CAGD applications (an integrated system is now under construction), and to apply it to volumetric data structures and sets of laser data. While many areas remain unexplored, we believe that the many potential applications of our approach

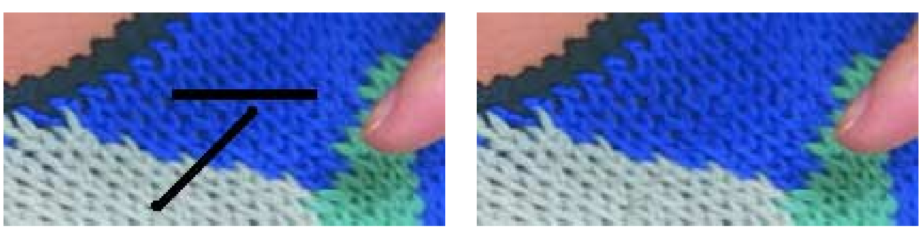
(a) (b) (c)

Figure 4. (a) Old photograph. (b) Selected retouching area. (c) Result produced with our algorithm. The algorithm can repaint disconnected areas.



(a) (b) (c)

Figure 5. (a) "Stone", from [2]. (b) Result produced with Hirani and T. Totsuka's algorithm [2]. (c) Restored image obtained with our algorithm.



(a) (b)

Figure 6. (a) "Wool", one additional sloping scratch was added to the test image from [2]. (b) Restored image obtained with our algorithm.

Table 1. Computation time for image/surface retouching.

Test File	AMD Athlon 1GHz, 128 MB RAM, Microsoft Windows 2000
Figure 2 (550 $\times$ 388 24bpp) Size of $\Omega^R(\Omega^L)$ area: 380(580) pixels Selected radius: 0.03	Data sorting: 0.02 sec $\mathbf{LL^t}$ decomposition, forward and back substitution: 0.02 sec Reconstruction time: 0.02 sec
Figure 3 (469 $\times$ 377 24bpp) Size of $\Omega^R(\Omega^L)$ area: 5741(11972) pixels Selected radius: 0.03	Data sorting: 0.751 sec $\mathbf{LL^t}$ decomposition, forward and back substitution: 7.19 sec Reconstruction time: 0.44 sec
Figure 4 (238 $\times$ 342 24bpp) Size of $\Omega^R(\Omega^L)$ area: 2076(3018) pixels Selected radius: 0.03	Data sorting: 0.1 sec $\mathbf{LL^t}$ decomposition, forward and back substitution: 0.12 sec Reconstruction time: 0.1 sec
Figure 5 (206 $\times$ 97 24bpp) Size of Ω^R area: 425 pixels Selected radius: 0.02	OS: Red Hat Linux 7.2 CPU: Intel Pentium3 700MHz Memory: 320MB JDK:Java(TM) 2 Runtime Environment, Standard Edition Reconstruction time: 0.186 sec

can generate further attention from the CG community.

References

[1] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, Image Inpainting, *Computer Graphics*, SIGGRAPH 2000, 417-422, July 2000.

[2] A.N. Hirani, and T. Totsuka, Combining frequency and spatial domain information for fast interactive image noise removal, SIGGRAPH 96, 269-276, 1996.

[3] M.M. Oliveira, B. Bowen, R. McKenna, and Y-S. Chang, Fast Digital Image Inpainting, in *Proceedings of the Visualization, Imaging, and Image Processing* IASTED *Conference*, Marbella, Spain, 261-266, Sept. 2001.

[4] S. Esedoglu and J. Shen, Digital Inpainting Based on the Mumford-Shan-Euler Image Model, IMA *Preprint* 1812, available from http://www.ima.umn.edu/ esedoglu/preprints/odyframe.htm

[5] A. Sarti, R. Malladi and J.A. Sethian, Computing Missing Boundaries in Images, in *Proceedings of the Visualization, Imaging, and Image Processing* IASTED *Conference*, Marbella, Spain, 495-500, Sept. 2001.

[6] H. Wendland, Piecewise polynomial, positive defined and compactly supported radial functions of minimal degree, *AICM*, 4, 389-396, 1995.

[7] http://www.savchenko.com/products/csrbf/gallery/

[8] M. Chen, A. E. Kaufman and R. Yagel (eds), *Volume Graphics*, Springer, 2000.

[9] S. Skaria, E. Akleman, F. I. Parke, Modeling Subdivision Control Meshes for Creating Cartoon Faces, in *Proceedings of the International Conference on Shape Modeling and Applications*, Genova, Italy, 216- 225, May 2001.

[10] B. Wyvill and K. van Overveld, Warping as a Modeling Tool for CSG/Implicit Models, in *Proceedings of the International Conference on Shape Modeling and Applications*, Aizu-Wakamatsu, Japan, 205-213, March 1997.

[11] S. Lee, G. Wolberg, and S. Y. Shin, Scattered Data Interpolation with Multilevel B-splines, *IEEE Transactions on Visualization and Computer Graphics*, 3(3), 228-244, 1997.

[12] J. H. Ahlberg, E. N. Nilson, and J. L. Walsh, *The Theory of Splines and Their Applications*, Academic Press, New York, 1967.

[13] J. Dushon, Splines Minimizing Rotation Invariants Semi-norms in Sobolev Spaces, in W. Schempp and K. Zeller (eds), *Constructive Theory of Functions of Several Variables*, Springer-Verlag, 85-100, 1976.

[14] V. A. Vasilenko, Spline-functions: *Theory, Algorithms, Programs*, Novosibirsk, Nauka Publishers, 1983.

[15] R. M. Bolle and B. C. Vemuri, On Three-Dimensional Surface Reconstruction Methods, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(1), 1-13, 1991.

[16] G. Greiner, Surface Construction Based on Variational Principles, in P. J. Laurent et al. (eds), *Wavelets, Images and Surface Fitting*, AL Peters Ltd., 277- 286, 1994.

[17] F. L. Bookstein, Principal Warps: Thin Plate Splines and the Decomposition of Deformations, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(6), 567-585, 1989.

[18] F.L. Bookstein, *Morphometric Tools for Landmark Data*, Cambridge University Press, 1991.

[19] P. Litwinovicz and L. Williams, Animating Images with Drawing, *Computer Graphics*, SIGGRAPH'94, 409-412, 1994.

[20] J. C. Carr, W. R. Fright, and R. K. Beatson, Surface Interpolation with Radial Basis Functions for Medical Imaging, *IEEE Transaction on Medical Imaging*, 16(1), 96-107, 1997.

[21] V. Savchenko, A. Pasko, O. Okunev, and T. Kunii, Function Representation of Solids Reconstructed from Scattered Surface Points and Contours, *Computer Graphics Forum*, 14(4), 181-188, 1995.

[22] G. Turk and J. F. O'Brien, Shape Transformation Using Variational Implicit Functions, *Computer Graphics*, SIGGRAPH'99, 335-342, 1999.

[23] R. K. Beatson and W. A. Light, Fast Evaluation of Radial Basis Functions: Methods for 2-D Polyharmonic Splines, *Tech. Rep. 119*, Mathematics Department, Univ. of Canterbury, Christchurch, New Zealand, Dec. 1994.

[24] W. Light, Using Radial Functions on Compact Domains, in P. J. Laurent et al. (eds), *Wavelets, Images and Surface Fitting*, AL Peters Ltd., 351-370, 1994.

[25] J. C. Carr, T.J. Mitchell, R.K. Beatson, J.B. Cherrie, W. R. Fright, B. C. McCallumm, and T.R. Evans, Reconstruction and representation of 3D Objects with Radial Basis Functions, *Computer Graphics*, SIGGRAPH'2001, 67 - 76, 2001.

[26] B. Morse, T. S. Yoo, P. Rheingans, D. T. Chen, and K.R. Subramanian, Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions, in *Proceedings of the Shape Modeling conference*, Genova, Italy, 89-98, May 2001.

[27] N. Kojekine, V. Savchenko, D. Berzin, I. Hagiwara, Software Tools for Compactly Supported Radial Basis Functions, CGIM 2001, IASTED Fourth International Conference on Computer Graphics and Imaging, Honolulu, Hawaii, August 13-16, 234-239, 2001.

[28] G. Barequet and M. Sharir, Filling gaps in the boundary of a polyhedron, *Technical Report 277/93*, Department of Computer Science, Tel Aviv University, 1993, Computer-Aided Geometric Design.

[29] T. Hermann, Z. Kovacs, and T. Varady, Special applications in surface fitting, in W. Strasser, R. Klein and R.Rau (eds), *Geometric Modeling: Theory and Practice*, Springer, 14-31, 1997.

[30] M. I. G. Bloor and M. J. Vilson, Spectral approximation to PDE surfaces, *CAD*, 28(2), 145 - 152, 1996.

[31] J.A. Setian, *Level Set Methods: Evolving Interfaces in Geometry*, Fluid Mechanics, Computer Vision, and Material Sciences, Cambridge University Press, 1996.

[32] R. T. Whitaker and D. E. Breen, Level-set Models for the Deformation of Solid Objects, in *Proceedings of Implicit Surfaces'98*, Seattle, USA, 19-35, June 1998.

[33] H. Wendland, On the smoothness of positive definite and radial functions, 14 September 1998, (*Preprint submitted to Elseivier Preprint*).

[34] H. Samet, *The Design and Analysis of Spatial Data Structures*, Addison-Wesley Pub Co, 1986.

[35] A. Jennings, A Compact Storage Scheme for the Solution of Symmetric Linear Simultaneous Equations, *Comput. Journal*, 9, 281-285, 1966.

[36] W.H. Press, S.A. Teukolsky, T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*, Cambridge University Press, 1997.

[37] A. George and J. W. H. Liu, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall: Englewood Cliffs, NJ, 1981.