Reconstruction of Variational Implicit Surfaces from Parallel Contours for Medical Imaging

D. Peri* and B. Crespin**

* Dipartimento di Ingegneria Elettrica, Università di Palermo, Italy
** LIRIS, Université Claude Bernard - Lyon 1, France

Abstract

We describe several algorithms for reconstructing a variational implicit surface (VIS) from contour data. These algorithms are applied on two kinds of data, polygons and grey-level images. We show that VIS are a natural choice for reconstruction purposes since they provide an automatic interpolation of sample points, and produce smooth surfaces.

1 Introduction

In this paper, we describe several experiments to reconstruct a variational implicit surface (VIS) from parallel contours. Many authors have studied this problem for different implicit models, because the implicit formulation allows to get smooth, continuous surfaces that have desirable properties for many applications (they are particularly well-suited for collision detection for example). But only a few authors investigated the use of VIS, although this model is a natural candidate for reconstruction purposes due to its interpolation properties: by definition, a variational surface interpolates a collection of sample points in space.
In the case of contour data, *i.e.* structured data organised in parallel slices, the reconstruction of an implicit surface is not straightforward because of the discrete nature of the data; implicit models in general are not particularly well-adapted to structured input due to their continuous formulation. However, VIS seem to be a good choice because in that case the interpolation is computed from all slices (and not as a piecewise interpolation between successive slices, which does not ensure continuity). In addition, structured input of different types can be processed within the same framework.

In the following, after a brief description of the variational model and existing implicit reconstruction techniques, we focus on two different kinds of data usually employed in medical imaging: polygons and grey-level images. In both cases, we describe how an efficient reconstruction can be carried and show the results applied on data obtained from CT scans.

2 State of the art

2.1 Implicit surface reconstruction

The problem of reconstructing a continuous, implicit surface from 3D sample points has been widely studied in the literature, mostly for data coming from medical imaging or 3D scanning. Only a few of these methods use structured data such as contour slices obtained from medical imaging as an input. We categorise these methods according to the implicit model employed.

In the first approach, an algebraic surface is deformed by external forces to fit the data points [17, 2, 7], but this is limited by the shape of the initial basic surface. Another idea is to find a geometric skeleton such that its associated implicit surface interpolates the data points. This approach was used in [15] where the skeleton is defined as random set of points refined iteratively through an energy-minimisation process, and enhanced by using the *medial axis* as an initial set [3, 10]. In the case of structured data organised in different slices, the process can also be enhanced by first computing the medial axis for each slice in two dimensions, which reduces the computation cost [1].

Apart from this well-known implicit models, many researchers have focused in the recent years on *interpolation* methods, *i.e.* the goal is to produce a function $f : \Re^3 \to \Re$ such as, for every sample point $p \in \Re^3$, we have $f(p) = 0$.

One possible approach when dealing with structured data is to compute a set of functions for every slice in two dimensions, then extend this set to a three-dimensional function [13, 16, 11]. But in that case specific techniques must be employed to preserve continuity between each plane. The method proposed in [25] can fit into this category, since a reconstruction algorithm is applied on a limited number of slices, but is restricted to simple shapes. In the more general context of reconstructing a surface from unorganised points, different approaches were presented such as piecewise surface reconstruction using a signed distance [12] or function sampling on a three-dimensional grid as in level-sets [23]. Variational surfaces are a particular case since the surface interpolates the data points by definition. Hence, the reconstruction problem is addressed naturally by this model that we present in the next section.

2.2 Variational implicit surfaces

Variational implicit surfaces or VIS [20] naturally take place among other implicit models, since the surface is defined as the zero-set of some function $f : \Re^3 \to \Re$. But as noticed above, in the variational case f has the property to exactly interpolate scalar values (not necessarily zero) specified at sample points. Thus, exact control over the produced surface is ensured, which makes VIS more suitable for reconstruction purposes than other implicit models such as skeletal implicit functions.

In the usual terminology of VIS, the sample points are called *constraints*, each constraint being a point p_i associated to a value c_i. This implies that f must satisfy:

$$\forall i \in \{1, n\}, \quad f(p_i) = c_i \tag{1}$$

where n is the number of constraints.

A solution for this system can be found if f is defined as a sum of n weighted, *radial-basis functions* h called RBF, and if there exists at least two constraints with oppositely-signed values. Then:

$$\forall P \in \Re^3, \quad f(P) = \sum_{j=1}^{n} w_j h(|P - p_j|) + D(P) \tag{2}$$

where $|P - p_j|$ denotes the euclidean distance between P and p_j and w_j is a scalar value representing the weight associated to constraint p_j and $D(P)$ is a degree one polynomial.

The formulation of the RBF is of critical importance, especially in the context of reconstruction. The most commonly used RBF are thin-plate functions [9], usually $h(x) = x^2 \log x$ in two dimensions, and $h(x) = x^3$ in three dimensions. The zero-set of function f in that case is a C^∞ continuous surface.

Then, satisfying the set of constraints in equation 1 can be expanded in:

$$\forall i \in \{1, n\}, \quad \sum_{j=1}^{n} (w_j h(|p_i - p_j|) + D(p_i)) = c_i \tag{3}$$

This reduces to solving the linear system $Az = c$, where:

$$A = \begin{bmatrix} h_{11} & h_{12} & \dots & h_{1n} & 1 & p_1^x & p_1^y & p_1^z \\ h_{21} & h_{22} & \dots & h_{2n} & 1 & p_2^x & p_2^y & p_2^z \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ h_{n1} & h_{n2} & \dots & h_{nn} & 1 & p_n^x & p_n^y & p_n^z \\ 1 & 1 & \dots & 1 & 0 & 0 & 0 & 0 \\ p_1^x & p_2^x & \dots & p_n^x & 0 & 0 & 0 & 0 \\ p_1^y & p_2^y & \dots & p_n^y & 0 & 0 & 0 & 0 \\ p_1^z & p_2^z & \dots & p_n^z & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4}$$

$$h_{ij} = h(|p_i - p_j|), p_i = (p_i^x, p_i^y, p_i^z)$$

$$z = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \\ d_0 \\ d_x \\ d_y \\ d_z \end{bmatrix} \quad c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Well-known methods such as Cholesky or LU decomposition can then be used to find z. Of course, since the thin-plate RBF have infinite support, the decomposition is very expensive. However, the computation time can be reduced using fast evaluation techniques [6]. Alternatively, the determination of the solution can also be greatly enhanced by using compactly-supported RBF with finite radius, *i.e.* $h(x) = 0$ if x is greater than some fixed radius [21, 14]. In this case, the linear system of equation 4 is a sparse one since many h_{ij} tend to zero.

This computation aspect is also important when evaluating the function, that is computing $f(P)$

for some $P \in \Re^3$. Here again, fast evaluation techniques [6] or compactly-supported RBF [21, 14] can be used to enhance distance computations through spatial partitioning.

2.3 Reconstruction issues with VIS

Using this variational approach, reconstructing a continuous surface from unorganised points is straightforward, since these points are considered as constraints with zero value, called *surface constraints*. The main problem is to add *off-surface* constraints, that is constraints with non-zero value. But this problem is usually easily solved using normal vectors associated to sample points: for each surface constraint, a *normal* constraint is found along the normal vector by adding a constant length. This idea has been widely used for reconstructing a VIS from sample points obtained from scattered-surface data [22, 14, 6, 8], polygons sets [24] or medical imaging [5].

The choice of the RBF usually depends on the density of the points set. But it should be noted that only RBF with infinite support such as thin-plate can ensure the continuity of the resulting surface if density of the data is too low [5].

To our knowledge, the specific problem of reconstructing a three-dimensional VIS from contour data has only been addressed in [19], and only for a very small set of slices (see [19] Figure 7). The basic idea is to use the set of points defining each contour in two dimensions as surface constraints for the VIS in three dimensions, using the z coordinate of the contour as the z coordinate for each of its points. Here again, the normal vector associated to each point is used to compute normal constraints. The resulting surface is everywhere continuous and, as noted in [19], it produces smooth rounded caps on the ends, naturally extending the surface beyond first and last slice. Moreover, the distance between two slices can be adjusted to shrink or extend the surface by modifying the z coordinate associated to each slice. However, the authors do not give much information about the input data, which makes it hard to reproduce their results.

In the remainder of this paper, we will focus on the reconstruction of a VIS from two different kinds of contour data.

3 Reconstruction from polygonal contours

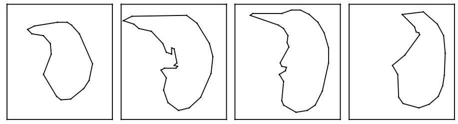

Figure 1: Four slices from the lung dataset used as a test example (top to bottom)

In this section, we describe our attempts to reconstruct a VIS from a set of polygonal contours. The test set is composed of 47 parallel polygons, obtained from a CT scan of a right lung, which was also used to test another reconstruction method described elsewhere [1]. Each polygon has about 20 vertices, the total number of vertices being $N = 545$; some of these polygons are showed on Figure 1. Each polygon is oriented, which means that we can compute the normal vector at each vertex.

Two reconstruction approaches applied to this set are presented. The first one is inspired by [19] and makes use of the normal vectors. In the second one, these are replaced by the interior of the polygons. We discuss the properties of each method in the last part.

3.1 Reconstruction using normal vectors

As in the method described above [19], our first approach to reconstruct a variational function f from polygonal contours is to make use of the normal vectors at each vertex.

First, we consider each vertex as a surface constraint (*i.e.* with zero value). This means we have a set C of N constraints p_i ($i \in [1, N]$) such as $f(p_i) = 0$.

To generate a consistent, orientable surface, we add a *normal* constraint ${p'}_i$ for each vertex v_i located at a predefined distance d of v_i along the normal vector $\vec{n_i}$ computed at v_i; $d = 0.1$ was chosen in our case. Since v_i belongs to an oriented polygon in the plane, we can use v_{i-1} and v_{i+1} to compute n_i. Then we must choose the value α such that $f({p'}_i) = \alpha$; in our case, we choose $\alpha = 1$, but experiments showed that lower or greater values do not change the overall resulting surface. Although this strategy gives good results in

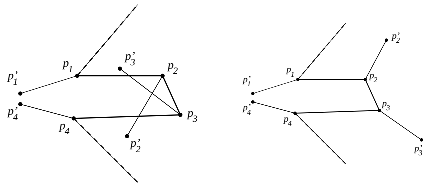

Figure 2: (a) Example of problematic configurations for normal constraints p_2 and p_3 (b) Using opposite normal constraints

most cases, the distance d must be chosen carefully in order to avoid self-intersecting configurations as in Figure 2a. In order to solve this problem, d must be sufficiently small to make sure that ${p'}_i$ does not belong to the polygon. A better alternative consists in using the opposite normal vector $-\vec{n_i}$ when a concavity occurs, *i.e.* when $v_{i-1}\widehat{v_i}v_{i+1} < \pi$. In that case, ${p'}_i$ becomes an *opposite normal* constraint such as $F({p'}_i) = -\alpha$ (see Figure 2b).

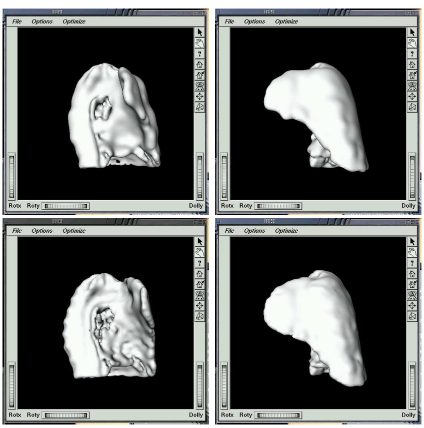

Figure 3: Reconstruction using normal vectors (front and side view) with 1090 (top) and 2342 (bottom) constraints. The crease observed at the bottom of the lung disappears on the second surface

Using this approach, a VIS is reconstructed from the polygonal contours by a set of $2N = 1090$ constraints. The resulting iso-surface for the lung

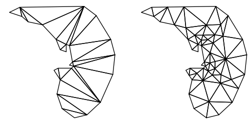

Figure 4: (a) Example of constrained Delaunay triangulation of one contour (b) Refined triangulation

dataset is shown on Figure 3 (top). It was computed using the software described in [24], which itself is based on Bloomenthal's implicit surface polygonizer [4] to extract the mesh.

Although this method produces a smooth surface that interpolates the input vertices correctly, some undesirable effects such as creases can be observed at the bottom of the organ. This is mainly due to the fact that geometric arrangements in adjacent polygons do not necessarily match. In such cases, our basic approach cannot give an efficient solution to this problem, because only vertices of the input polygons are used to reconstruct the surface; information from edges linking these vertices is not taken into account. To overcome this, a simple solution is to refine the constraints set by adding new surface and normal constraints corresponding to problematic edges' midpoints. We find such edges by evaluating the variational function value at the midpoint: if it is greater than a predefined threshold, then two constraints (surface and normal, or surface and opposite normal) are added for this midpoint. When all problematic edges have been identified, the weights of the variational function are re-computed. This method provides a way to measure the overall error between the input data and the reconstructed surface. It was applied to generate the more accurate surface shown on Figure 3 (bottom), with an overall least-squares error of 5.

3.2 Reconstruction using interior constraints

Another way to generate a VIS from polygonal data is to define *interior* constraint instead of *normal* constraints: since the polygons are oriented, we can characterise their interior and exterior, and hence specify that the variational function should have the same behaviour. An *interior* constraint p_j is simply defined as a point inside the polygon having a negative value, *i.e.* $f(p_j) = -\alpha$.

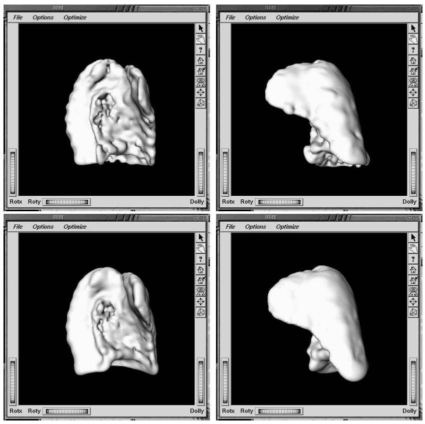

Figure 5: Reconstruction using interior constraints with 2224 (top) and 3568 (bottom) constraints

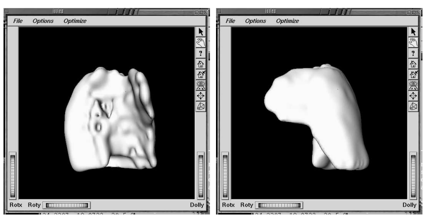

Figure 6: Reconstruction using interior constraints and one slice out of five (698 constraints)

To generate interior constraints for one polygonal contour, we first compute a set of triangles inside using a *Delaunay constrained triangulation* which is very efficient in two dimensions (see Figure 4a). Then the interior constraints are defined as the centres of these triangles. This approach was used to compute the VIS shown on Figure 5 (top).

Although the number of constraints is relatively high, the resulting surface still exhibits undesirable holes and creases. Therefore, as in the previous approach, a refinement step is necessary. It is performed in that case by using Ruppert's Delaunay refinement algorithm described in [18], which produces guaranteed-quality triangles (see Figure 4b), and by adding the centres as new interior constraints. The resulting surface in that case is shown on Figure 5 (bottom).

3.3 Discussion

Both the normal or interior approaches give good results. Of course, the interpolation is driven more by the vertices than by the edges, but in both approaches it is possible to refine the constraints set to get closer to the polygonal contour. Still, it seems that the interior constraints approach generate a higher number of points for the same kind of results. On the other hand, we made some reconstruction experiments using a fewer number of slices as an input (one slice out of five), in order to get a low resolution surface with faster computation times. This idea can also be found in [25] for interpolating between slices. The results were better using the interior approach, mainly because the infinite RBF allows to "spread" the influence of the interior constraints as shown on Figure 6. This is not the case with normal constraints that are located too close to their corresponding surface constraints.

Moreover, as suggested by one referee, the interior approach could be enhanced by associating different values to interior constraints, depending on the distance to the border. The resulting variational function would then give a better interpolation, but issues regarding computation times and how to choose efficient constraint values still have to be addressed.

4 Reconstruction from Gray-scale images

We now show the reconstruction of a variational surface from a set of contours extracted from gray-scale images. That is, our goal VIS should interpolate parallel contours displaced along a vertical axis. The main difference with the reconstruction from contours is that we have much more detailed data which, of course, generates more constraints. Extracting contours and reconstructing interpolating VIS is accomplished by means of a series of stages described as follows.

In most applications a proper segmentation is required to obtain the slices contours from the original gray-level images. In the example we show VIS reconstruction of a lung carried out a set of CT-scan gray-level images. To accomplish this, every lung slice must be isolated from the rest of other similar objects laying in the same image. We opted for semi-automatic segmentation based on thresholding, labeling and manual selection of the lung slice. Segmentation was carried on each of the original image, one at a time, resulting in a set of segmented binary images containing only pixels belonging to lung slices. The segmented slices were then processed as in the following steps (see Figure 7).

4.1 Contour extraction and VIS constraints generation

We extracted contours by collecting all the pixel having at most three 4-connected neighbours (out of four) that belong to the segmented slice. This ensures 4-connectivity of all the contour pixels. Pixel coordinates in the image plane give the the first two components of the contour points while the third is taken from the image displacement along the vertical axis as in the previous section.

We then generated VIS constraints for each given contour point. A couple of constraints is generated for each point (surface and normal). The surface constraint is placed at contour point location, its value set to zero. Gradient of the segmented binary image is evaluated at the contour point location: the normal constraint is then placed at a distance d away from the contour point in the direction of the gradient. Our experiments showed that $d = 0.5$ works fine for our data set.

4.2 Stacked stripes

Our example data set is composed of 40 gray-level images producing a slice each. After the described processing, 20219 contour points were produced. Such huge number of points implies long computation time and requires great amounts of memory just to store the linear system matrix. Computation time issue arises also during VIS evaluation, consequently afflicting rendering (polygonization or ray-tracing).

Instead of gathering all the constraint points to generate one comprehensive VIS we implemented "stacked stripes" by gathering sequentially contiguous parallel contours in groups of three and generating a VIS -"stripe" - for each of the groups as shown in Figure 8.

The entire surface can then be defined as:

$$f(x,y,z) = \sum_{i=1}^{n} f_i(x,y,z) \tag{5}$$

where:

$$f_i(x,y,z) = \begin{cases} 0, z < z_i \\ f'_i(x,y,z), z_i \leq z < z_{i+1} \\ 0, z \geq z_{i+1} \end{cases} \tag{6}$$

and $f'_i(x,y,z)$ is the VIS constructed from contours i, $i+1$ and $i+2$ and z_i and z_{i+1} the coordinates in the z-axis of the first two contours. This way, contiguous stripes share constraints from two slices and this partial overlapping ensures the expected local behaviour of the surface. We then obtain the desired surface by composing all the stripes (Figure 9). Results are showed in Figure 10.

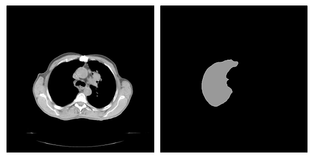

Figure 7: Gray-level image processing: original CT scan image, segmented image with highlighted contour

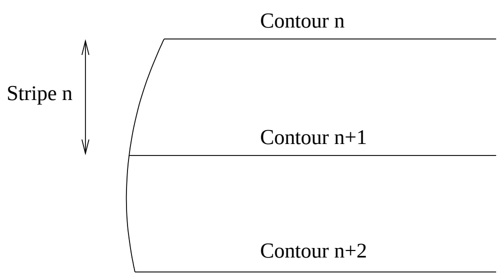

Figure 8: Each stripe is defined by means of three contiguous contours and evaluated in the space comprised between the first two contours.

Stacked stripes allowed us to obtain smooth surfaces out of a great number of constraints points while keeping computation time and hardware requisites acceptable. It seems that VIS generation could be further optimised when reconstructing from grey-level images by using fast evaluation techniques [6], since the data is very dense in that case. Other possible optimisations include parallelising the "stripes" computation. Some artifacts visible on the surface can be accounted to the sharp

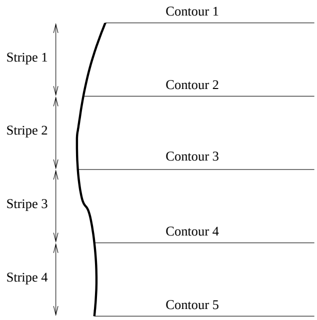

Figure 9: The desired surface is obtained by composing all the stripes.

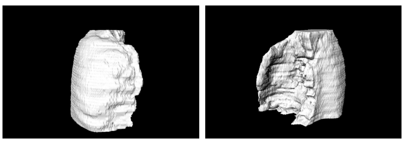

Figure 10: Polygonal model: frontal and side views

edged pixelised contours produced by the segmentation step. This will be addressed in future work.

5 Conclusion

We have presented in this paper several results concerning the reconstruction of a VIS from contour data, either polygonal or grey-scale. In both cases, our experiments showed that VIS are a very good candidate for reconstruction purposes, since its properties give natural, continuous interpolation of the contour points. They are also easily refinable by adding more constraints to the model, thus allowing to produce more detailed surfaces if necessary.

In the future, we intend to optimise our reconstruction algorithms by applying better techniques for the computation and evaluation of the variational function. Investigating the interpolation properties of VIS is also appealing for other purposes, for example semi-automatic segmentation or computer-aided contouring of medical data. Other properties of VIS could also be used, especially for simulation and collision detection.

Acknowledgements

The authors wish to thank V. Baudet, D. Sarrut, the Centre Leon Berard and Christie's Hospital for providing data, and G. Turk, J. Bloomenthal and J. Shewchuk for their software.

References

[1] M. Amrani, B. Crespin and B. Shariat. Skeletal Implicit Surface Reconstruction from Sections for Flexible Body Simulation. *Information Visualization Proceedings*, 2001.

[2] E. Bardinet, L. D. Cohen and N. Ayache. A parametric deformable model to fit unstructured 3D data. *Technical Report 2617*, INRIA Sophia Antipolis, 1995.

[3] E. Bittar, N. Tsingos and M.P. Gascuel. Automatic Reconstruction of Unstructured 3D Data: Combining a Medial Axis and Implicit Surfaces. *Eurographics'95 Proceedings*, pp. 457–468, 1995.

[4] J. Bloomenthal. An Implicit Surface Polygonizer. in *Graphics Gems IV*, pp. 324–349, Academic Press, 1994.

[5] J.C. Carr, W.R. Fright and R.K. Beatson. Surface interpolation with radial basis functions for medical imaging. *Transactions on Medical Imaging*, **16**(1), 1997.

[6] J.C. Carr, R.K. Beatson, J.B. Cherrie, T.J. Mitchell, W.R. Fright, B.C. McCallum and T.R. Evans. Reconstruction and Representation of 3D Objects with Radial Basis Functions. *ACM SIGGRAPH Proceedings*, pp. 67–76, 2001.

[7] Isaac Cohen and Laurent Cohen. A Hybrid Hyperquadric Model for 2-D and 3-D Data Fitting. *Technical Report 2188*, INRIA, 1994.

[8] Huong Quynh Dinh, Greg Turk and Greg Slabaugh. Reconstructing Surfaces by Volumetric Regularization Using Radial Basis Functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**(10):1358–1371, 2002.

[9] J. Duchon. Spline minimizing rotation-invariant semi-norms in Sobolev spaces. in *Constructive Theory of Functions of Several*

Variables (W. Schempp and K. Zeller), **571**:85–100, 1977.

[10] E. Ferley, M.P. Cani Gascuel and D. Attali. Skeletal Reconstruction of Branching Shapes. *Implicit Surfaces'96 Proceedings*, pp. 127–142, 1996.

[11] E. Galin and S. Akkouche. Fast Surface Reconstruction from Contours using Implicit Surfaces. *Implicit Surfaces'98 Proceedings*, pp. 139–144, 1998.

[12] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald and W. Stuetzle. Surface Reconstruction from Unorganized Points. *SIGGRAPH'92 Proceedings*, pp. 71–78, 1992.

[13] Mark W. Jones and M. Chen. A New Approach to the Construction of Surfaces from Contour Data. *Computer Graphics Forum*, **13**(3):75–84, 1994.

[14] B.S. 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. *Proceedings of Shape Modeling International*, 2001.

[15] S. Muraki. Volumetric shape description of range data using "Blobby Model". *SIGGRAPH'91 Proceedings*, pp. 227–235, 1991.

[16] V. V. Savchenko, A. A. Pasko, O. G. Okunev and Tosiyasu L. Kunii. Function Representation of Solids Reconstructed from Scattered Surface Points and Contours. *Computer Graphics Forum*, **14**(4):181–188, 1995.

[17] S. Sclaroff and A. Pentland. Generalized Implicit Functions for Computer Graphics. *SIGGRAPH'91 Proceedings*, pp. 247–250, 1991.

[18] J. Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. *Workshop on Applied Computational Geometry Proceedings*, LNCS, 1996.

[19] Greg Turk and James O'Brien. Shape Transformation Using Variational Implicit Functions. *SIGGRAPH Proceedings*, pp.335–342, 1999.

[20] Greg Turk, Huong Quynh Dinh, James O'Brien and Gary Yngve. Implicit Surfaces that Interpolate. *Proceedings of Shape Modelling International*, pp. 62-71, 2001.

[21] H. Wendland. Piecewise polynomial, positive defined and compactly supported radial functions of minimal degree. *Advances in Computational Mathematics*, **4**:389–396, 1995.

[22] H. Wendland. Surface reconstruction from unorganized points. *Technical Report*, Preprint Göttingen, 2002.

[23] Ross Whitaker and David Breen. Level-set models for the deformation of solid objects. *Proceedings of Implicit Surfaces'98*, pp. 19–36, 1998.

[24] G. Yngve and G. Turk. Robust Creation of Implicit Surfaces from Polygonal Meshes. *IEEE Transactions on Visualization and Computer Graphics*, **8**(4):346–359, 2002.

[25] Egor Zindy, Christopher J. Moore, David Burton and Michael Lalor. Morphological Definition of Anatomic Shapes using Minimal Datasets. *Information Visualisation Proceedings*, pp. 366–370, 2000.