A Hole-Filling Algorithm for Triangular Meshes Using Local Radial Basis Function

John Branch[1], Flavio Prieto[2] and Pierre Boulanger[3]

[1]Universidad Nacional de Colombia, Sede Medellín
jwbranch@unal.edu.co
[2]Universidad Nacional de Colombia, Sede Manizales
faprietoo@unal.edu.co
[3]Department of Computing Science
University of Alberta, Canada
pierreb@cs.ualberta.ca

Abstract. Creating models of real objects is a complex task for which the use of traditional modeling techniques has proven to be difficult. To solve some of the problem encountered, laser rangefinders are frequently used to sample an object´s surface from several viewpoints resulting in a set of range images that are registered and integrated into a final triangulated model. In practice, due to surface reflectance properties, occlusions and accessibility limitations, certain areas of the object´s surface are not sampled leaving holes which create undesirable artifacts in the integrated model. In this paper, we present a novel algorithm for the automatic hole–filling of triangulated models. The algorithm starts by locating hole boundary regions. A hole consists of a closed path of edges of boundary triangles that have at least an edge, which is not shared with any other triangle. The edge of the hole is then fitted with a b-spline where the average variation of the torsion of the b-spline approximation is calculated. Using a simple threshold of the average variation of the torsion along the edge, one can automatically classify real holes from man-made holes. Following this classification process, we then use an automated version of a radial basis function interpolator to fill the inside of the hole using neighboring edges. Excellent experimental results are presented.

1. Introduction

Creating accurate models of real environments is a nontrivial task for which traditional modeling techniques are inappropriate. In these situations, the use of laser rangefinders [5] seems attractive due to their relative independence from the sampled geometry and their short acquisition time. The combined use of range and color images is very promising and it has been demonstrated that together they can produce an unprecedented degree of photorealism [15, 16]. Unfortunately, surface properties (i.e., low or specular reflectance), occlusions and accessibility limitations cause the scanner to miss some surface elements, leading to incomplete reconstruction of the scene and introducing holes in the resulting models. Creating high-quality mesh representations of objects based on such incomplete information remains a challenge [24]. Due to the costs and difficulties involved in scanning real environments, it is desirable to have automatic or semiautomatic tools for helping users to improve the quality of incomplete data sets.

The problem of filling holes in a triangulated mesh can be divided into two sub-problems: hole identification and construction of the missing data using the available data near the holes. Unfortunately, none of these problems are trivial since holes created during the scanning of geometrically rich objects, such as detailed sculptures, can be quite complex [9]. However, in many practical cases, holes occurring in range images can be topologically simpler. This is the case of many holes found when scanning interior environments where most surfaces tend to be smooth and planar areas are abundant (for example: imagine a home or an office environment). For these situations, simpler algorithms for identifying holes and for parameterizing their neighbors can be specified to avoid the problems usually associated with more general cases.

This paper presents a novel algorithm for automatically identifying and filling holes in regions associated with smooth surfaces. Although our algorithm is targeted towards filling holes in smooth surfaces, it does not provide a general solution to the hole-filling problem. It is conceptually very simple and its implementation is straightforward. The algorithm takes a triangulated mesh, which is analyzed for the existence of boundary edges (edges that belong to a single triangle). The occurrence of a hole implies the existence of a cycle defined by boundary edges. Thus, once a boundary edge is found, the algorithm traces the entire boundary. A ring of points around the boundary is used for an interpolation procedure that will eventually fill in the hole. The points near the hole are then used to fit a surface using a radial basis function (RBF) interpolator. An important feature of our algorithm is that it guarantees that the reconstructed patches blend smoothly into the original surface. Moreover, the reconstructed surface preserves the original sampling rate of the original mesh. As the new primitives are distinguished from the original points, they can be processed further. Since the algorithm works after surface reconstruction, it can be used with any reconstruction technique and its processing is only limited to the size of holes. In this paper, we demonstrate the effectiveness of our approach in real data sets and show how it can significantly improve the overall quality of a triangular mesh model.

The paper is organized as follows: Section 2 discusses previous work. Section 3 presents the details of the hole-filling algorithm. Section 4 discusses results obtained using this algorithm, and Section 5 summarizes the contributions of this paper and proposes future developments.

2. Previous and Related Work

Hole filling is an important problem in object reconstruction and this work benefits from previous efforts in areas such as range image registration [6, 17, 20] and surface reconstruction from point clouds [2, 3, 11, 12].

Curless and Levoy [8] used a hole-filling technique to interpolate non-sampled surfaces in concave regions of objects. In this case, the added surfaces were intended to produce “watertight” models for reproduction using rapid prototyping machines. Their algorithm has little or no impact on the appearance of the objects. In our work, we are concerned about the reconstruction of holes that, if not fixed, would result in major rendering artifacts.

Carr et al. [7] use polyharmonic radial basis functions (RBF) to fit an implicit representation to a set of sampled points. This technique consists of creating a signed distance function, fitting an RBF to the resulting distance function, and extracting iso-surfaces from the fitted RBF. It is general and produces very impressive results, but the entire set of points is treated as a single implicit function. In order to create the signed distance function, the system needs to know what portion of the space corresponds to the exterior of the surface and what portion corresponds to its interior, which may not always be readily available.

Davis et al. [9] use a volumetric diffusion approach, analogous to in-painting techniques [4, 18], to fill holes in range scans. The process consists of converting a surface into a voxel-based representation with a clamped signed distance function. The diffusion algorithm consists of alternated steps of blurring and compositing, after which the final surface is extracted using Marching Cubes [14, 28].

Alexa et al. [1] used point sets to represent shapes and employed an approach similar to ours as they also locally project points onto planes and fit surfaces through those points. The reconstructed surfaces are used to sub–sample the point set. Their method, however, does not attempt to fill holes on surfaces.

Wang and Oliveira [21] proposed a pipeline to improve the reconstruction of scenes represented as sets of range images. The pipeline consists of a segmentation step followed by the reconstruction of missing geometric and textural information for individual objects. The reconstruction of missing geometric data exploits the fact that real (indoor) scenes usually contain many planar and symmetric surfaces. Thus, a 3D Hough transformation is used to identify large planar regions, whose corresponding samples are replaced by texture-mapped polygons and are also removed from the point cloud [10, 22, 23]. In the remaining dataset, individual objects are segmented as clusters of spatially closed points, using an incremental surface reconstruction algorithm [11]. Inside each cluster, the point cloud is analyzed and searched for approximately symmetric patterns using a variation of the 3D Hough transformation [21]. As these patterns are identified; the algorithm automatically proceeds with reconstruction by mirroring data from one part of the model to another. Figure 1 shows a chair extracted from a real data set. The image on the left corresponds to the original samples with large holes. The image to the right shows the model recovered using the symmetry-based reconstruction algorithm described in [21]. It provides a significant improvement with respect to the original model, but some holes are still visible. Such holes essentially result from the lack of data on both sides of the model and from limitations of surface reconstruction algorithms [2, 3, 11, 12] that work effectively on areas with variable sampling density. The algorithm presented in this paper intends to fill these remaining holes.

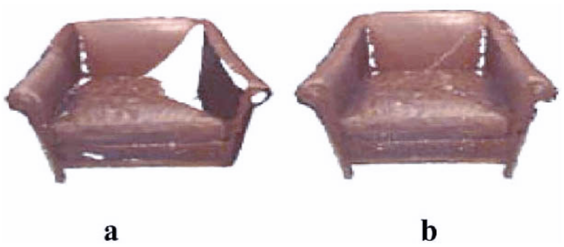


Figure 1. a) Chair from the range image has big holes due to occlusion, b) Symmetry-based reconstruction.

3. Hole-Filling Algorithm

With the objective of correcting the topological anomalies related to the absence of information in mesh representing objects, it is necessary to generate new points in those regions which have not been scanned.

The proposed methodology first identifies and analyzes the holes present in the grid to determine which ones must be filled and which ones belong to the topology of the object. For example, discontinuities are present in the eye area on the surface of the mask as shown in Figure 2. Figure 3 shows the block diagram of the proposed algorithm.

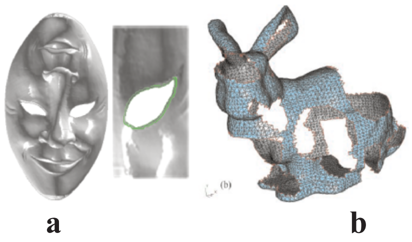


Figure 2. a) Surface Hole, b) Representation Hole.

The analysis of the holes consists of studying the torsion of the contour curve of every hole. This analysis is based on the idea that every hole belonging to the surface is smooth and regular, but the holes generated by occlusion often present irregularities reflected in high torsion of the contour. Some examples are shown in Figure 4. Next, an iterative process is started to fill the holes using neighboring points. These points are generated by means of local interpolators of radial basis functions originating from a neighborhood selected iteratively around a hole until a pre-selected fitting threshold is reached.

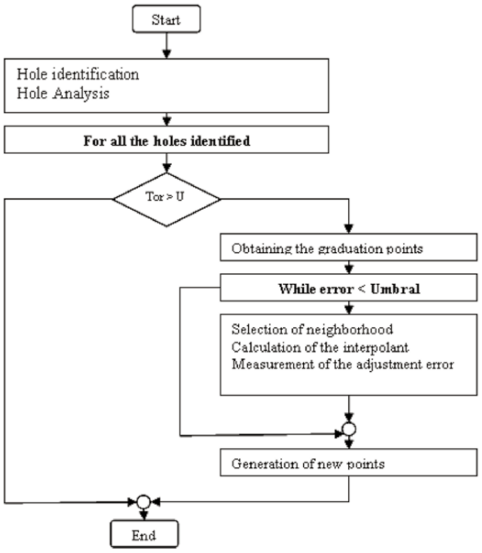
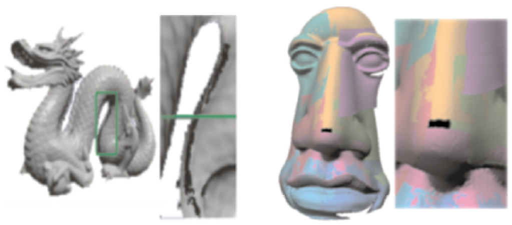

Start
Hole identification
Hole Analysis
For all the holes identified
Tor > U
Obtaining the graduation points
While error < Umbral
Selection of neighborhood
Calculation of the interpolant
Measurement of the adjustment error
Generation of new points
End

Figure 3. Block diagram of the algorithm.

3.1. Hole Identification

The first step for the process of hole filling is the process of identification, during which it is possible to find the different types of holes present in the topology of an object. There are those that really belong to the surface, and those that are caused by the acquisition process itself, i.e, such as holes due to occlusion or due to insufficient views as illustrated in Figure 4.



a **b**

Figure 4. a) Hole generated by occlusion, B) Hole generated by partial scanning.

If the object is a triangular mesh, a hole consists of a closed path of edges of border triangles. A border triangle is the one that has at least an edge which is not shared with any other triangle (that edge is called a limiting edge).

By following these border edges the holes can be detected automatically. However, it is possible to differentiate two kinds of borders: an internal limit which delimits a hole on a surface, and an external limit which delimits a patch or an island inside a hole or the limits of the surface. For the filling process, the path which represents the limits of the surface is eliminated within the set of detected holes. This elimination can be done by verifying that every one of the holes identified does not enclose a surface.

Initially, the algorithm takes a seed triangle located at any part of the grid, and it searches on the whole mesh until it finds a border triangle from which it starts a recursive search to find a closed path. This search is done by determining which one of the three edges is on a border, and then it searches for an adjacent border triangle that shares some of those vertices. The algorithm proceeds until the starting triangle is found again.

Figure 5 shows lateral views of the Standford bunny data set. It has five holes, two of them are part of the real object. The other holes were generated in early steps of surface reconstruction (acquisition and registration). Figure 6 shows the final result of the hole identification algorithm.

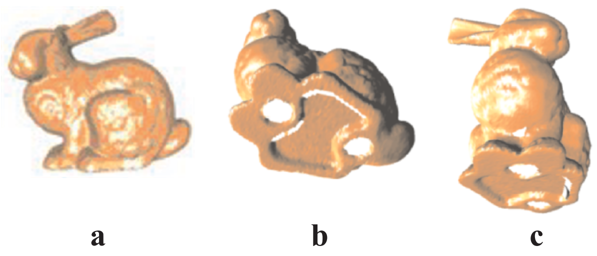


Figure 5. a) Lateral view of Stanford's Bunny, b) and c) View of the five holes that the object owns.

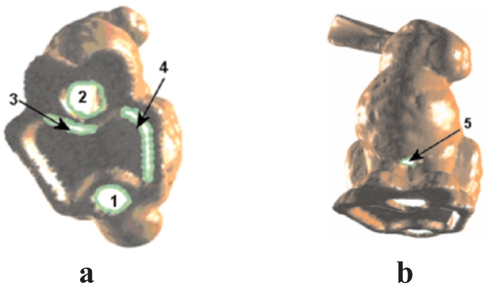


Figure 6. a) and b) Result of the algorithm of holes identification on the mesh, five identified holes.

3.2. Hole Analysis

The step following the detection of holes is analysis. This stage tries to determine whether a hole must be filled or not, whether the hole is present on the surface of the real object, or if the hole was caused during an intermediate stages of reconstruction.

There are an infinite number of configurations of holes on free form objects which makes it very difficult to identify the actual hole belonging to the surface. This is why the hole-filling process requires an interaction with the user. An attempt to automate this procedure consists in analyzing the contour curve of each hole. Keeping in mind that the smoothness of the contour is relative to the density of the sampling points, the holes present in man-made objects usually have smooth contours.

On the other hand, the holes caused by occlusion, present great variability of the contour curve as shown in Figure 7.

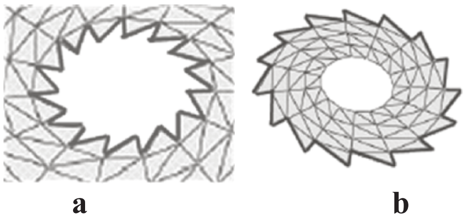


Figure 7. a) Internal contour, b) External contour.

Since the contour line is a curve, it can be studied and classified according to its geometric properties as curvature and torsion. To classify the contour curves we use the torsion and not the curvature because a hole on a surface can have a wide range of curvature variations. It is important to establish the behaviour of every curve in the space, such as its smoothness or high variability.

The study of the torsion of a curve depends on the behaviour of the osculating plane. The osculating plane is the plane nearest to the curve at an arbitrary point A. This plane crosses the A point and it contains the tangent $\vec{T}$ and the normal $\vec{N}$ to the curve at A, as illustrated in Figure 8.

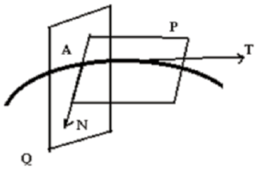


Figure 8. Osculating plane (Vector for N and T necessary).

From point-to-point through the curve, the position of the osculating plane varies in a similar way to the tangent vector. This simple characteristic allows characterization of the curvature. The variation of the osculating plane allows calculation of the torsion of a curve. Similar to the curvature, the variation of the osculating plane is measured according to the arch longitude. That is, if ψ is the angle between the osculating planes at a fixed point A and a neighbouring point X, and if Δs is the arch longitude AX, then the torsion τ at the point A is defined as the limit

$$\tau = \lim_{\Delta s \to 0} \frac{\psi}{\Delta s}.$$

The sign of the torsion depends on the side of the curve towards which the osculating plane turns when it is moving across the curve. However, according to differential geometry, the properties of the curve at a point are those properties which depend on an arbitrarily small environment. The properties of this type of curve are defined in terms of derivative on the curve at the point. The estimation of the torsion is defined as follows:

$$\tau = \frac{|(\vec{f}'(t) \times \vec{f}''(t)) \;\bullet\; \vec{f}'''(t)|}{|\vec{f}'(t) \times \vec{f}''(t)|^2}.$$

To estimate the torsion of the curve of at least three degrees are necessary. In order to compute these derivatives, the boundary points are fitted with the Bézier's parametric curve $\vec{f}(t)$ defined as follows:

$$\vec{f}(t) = \sum_{i=0}^{n} \vec{p}_i B_{i,3}(t),$$

where:

$$x(t) = \sum_{i=0}^{n} x_i B_{i,3}(t),$$

$$y(t) = \sum_{i=0}^{n} y_i B_{i,3}(t),$$

$$z(t) = \sum_{i=0}^{n} z_i B_{i,3}(t),$$

and where $\vec{p}_i$ are the control points of the Bézier curve and $B_{i,3}(t)$ are the Bernstein polynomials of the third degree. These are defined by:

$$B_{0,3} = (1-t)^3,$$

$$B_{1,3} = 3t(1-t)^2,$$

$$B_{2,3} = 3t^2(1-t)\;,$$

$$B_{3,3} = t^3.$$

The contour is partially approximated by means of Bézier curves of the third degree with sets of four continuous points until the estimation of the torsion is computed at every point along the contour.

Once they are obtained, the equations of the torsion for the segments along the curve are evaluated at the last point. Since Bézier curves guarantee that the obtained curve has the extreme points of the set from which it is calculated, the error of approximation that is present at the intermediate points does not affect the torsion's estimation (see Figure 9).

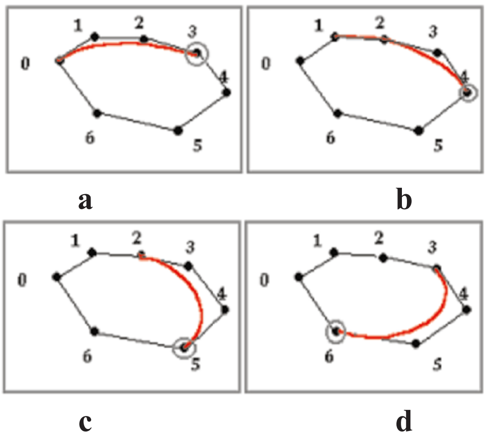


Figure 9. Approximation to the contour curve by means of Bézier's curves and points on which the torsion is estimated.

Finally, the variance of the torsions is calculated to measure the level of dispersion of the measured values at each point. The holes whose contours have a variance of torsion higher than a pre-established threshold, will be filled The value of this measure is obtained by:

$$S_{torsion} = \frac{\sum_{i=0}^{n} (\tau_i - \bar{\tau})^2}{n}.$$

In general, low values of dispersion suggest a surface hole. Figure 10 shows different cases of the hole and its values of dispersion. In these cases, low dispersion is considered as values less than 0.1. Although the smoothing of long contour curves depend on sample

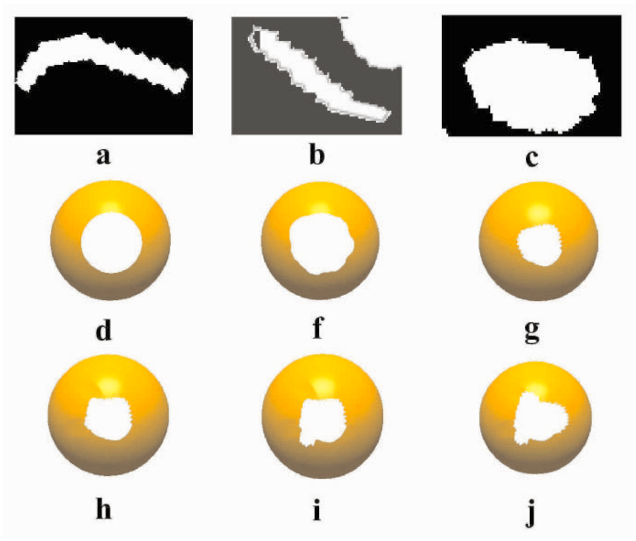


Figure 10. Measurement of the torsion of contour curve in six different cases a) 0.31, b) 0.42, c) 0.245, d) 1.1E-4, e) 3.2E-3, f) 0.04, g) 0.09, h) 0.38, i) 0.51, j) 0.63.

density. We assumed that in a real data range the density is adequate and that the mesh does not yet have any reduction procedure.

3.3. Calculation of the Interpolant

Once the holes to be filled are classified, the missing points are computed by an interpolation function using a continuous interpolation scheme. In order to do this, a function *h(x)* is calculated from a set of points distributed in a homogenous way around the contour of the hole. This interpolating function is constructed using a radial basis function.

The procedures based on radial basis functions have proven to be very useful in the reconstruction of shapes from noisy, disperse, and incomplete data [25, 26, 27]. Recent studies about RBF are centred on the reconstruction of a big set of points produced by modern acquisition devices [11, 17, 7, 26].

The radial basis functions are circularly symmetric functions centred on a point called *centre*. To calculate an interpolant of RBF, let us consider a set of points $P = \{\vec{p}_1, \ldots, \vec{p}_N\}$ sampled from a surface S and with a set of normals, $N = \{\vec{n}_1, \ldots, \vec{n}_n\}$, which indicate the orientation of the surface. The main goal is to build a function $h(\vec{r})$ in such a way that the set of zeroes satisfies the equation $\vec{r} \in \Re^3 : h(\vec{r}) = 0$, approximating the set of points P.

The typical interpolation function $h(\vec{r})$ interpolating P is defined as:

$$h(\vec{r}) = \sum_{p_i \in p} [g_i(\vec{r}) + \lambda_i] \phi_\sigma \left(\left\| \vec{r} - \vec{C}_i \right\| \right),$$

where $\phi_\sigma(s) = \phi(s/\sigma)$, $\phi(s) = |s|^3$ is a tri-harmonic radial basis function used for the approximation of the absent surface The λ_i are the set of weights associated to each centre, g_i are typically a polynomials of second or third degree, and the $\vec{C}_i$ are the set of centres.

For hole filling, it is necessary to compute independent interpolators local to every hole. Therefore, for every hole, a different interpolant function is estimated with a reduced set-of-points. This set-of-points should be as big as possible and homogeneously distributed so that the obtained function can estimate the topology of the missing points.

3.4. Centres Selection

The computation of the interpolating radial basis functions, which is not a compact support, is expensive. Therefore, the selection of the centres or set-of-points on which the interpolant will be calculated carefully. The estimation of the adequate neighbourhood is done by means of an iterative procedure (see Figure 11).

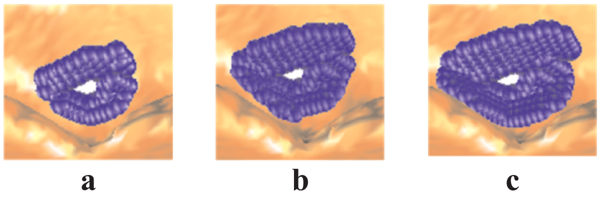


Figure 11. a) Iteration 1, Neighbourhood size: 38. b) Iteration 2, Neighbourhood size: 106. c) Iteration 3. Neighbourhood size: 172.

This process starts with a small number of selected centres, as a set of points radially near to each one of the vertices on the contour curve, the size of radio is initially established by the user, this selection is shown in Figure 12. An interpolant is calculated for these points and the precision is measured on a set of control points. If the precision measured is not equal to the threshold or lower than a pre-established threshold, the size of radio is duplicated, and a new selection process is done until a threshold is reached. In every one of these iterations an interpolant is calculated. The evaluation of the quality of the interpolation is done over a set of reference points, which initially belong to the neighbourhood of the hole but are not used to calculate the interpolant.

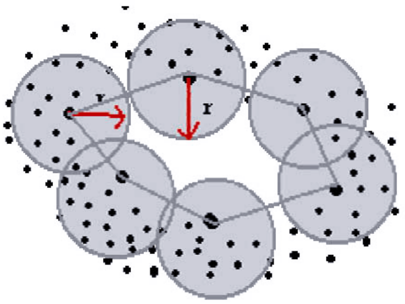


Figure 12. Estimation of the set of centres.

Figure 13 shows the result of the selection of the adequate neighbourhood for the calculation of the interpolating function on a real object. Once the initial neighbourhood is obtained, the reference points set must be determined to measure the quality of the interpolating function. This set-of-points should be kept constant. In the proposed algorithm, the set-of-points of the initial neighbourhood is clustered to obtain homogeneous regions which describe different variations of the topology in the regions around the hole. A cluster type K-mean [19] is used, where the parameter K will be equal to the number of vertices that form the hole's contour.

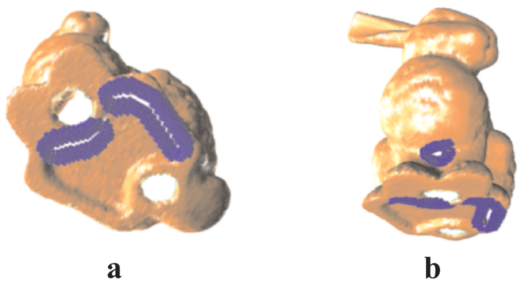


Figure 13. a) and b) final neighborhood for each hole in the bunny.

Once the set of regions defined for every cluster is obtained, a point is randomly selected from each of them (see Figure 14), which will represent every region. In this way, it is assured that the evaluation is done homogeneously around the hole. If an interpolating function reaches the threshold with high precision, it means that it represents the topology of the hole's neighbourhood.

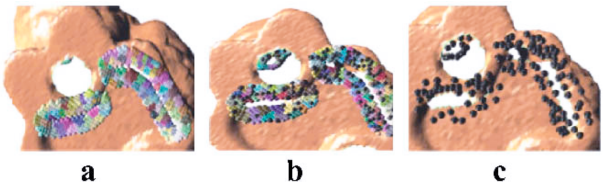


Figure 14. a) Neighbourhood clusterization. b) and c) Selection of the reference points set.

3.5. Filling the Hole

In order to fill each hole, it is important to remember that the reconstructed segment preserves the sampling density of the original mesh; that is, the sampling density that is measured for each one of the holes.

In general, two important criteria are used for determining the new points that fill each hole. First, the position of new points should be inside the hole and the new triangles added to the hole must be easy to merge with the original mesh. The local triangulation is an efficient procedure for hole-filling because it avoids the remeshing of the cloud-of-points. Additional procedures such as the normal estimation over the new points and the new normal of the contour points that will be different due the new

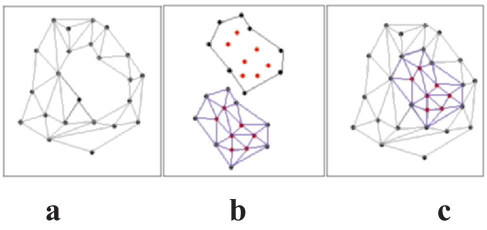


Figure 15. Hole triangulation a) Initial mesh, b) Contour extraction, new points generation and local triangulation, c) Filling hole.

segment of the surface, can also be made locally (see Figure 15). Second, the density of the new set-of-points must be of that vicinity.

In order to generate the new segment of the mesh, we use an iso-surface algorithm with a RBF interpolator [7]. Given a density function S, an iso-surface at value a is defined as the set of all points $\vec{p}$ where $f(\vec{p}) = a$. In the context of surface reconstruction, where S represents a signed-distance function. The reconstructed surface corresponds to the iso-surface where f = 0.

The function S is sampled at regular intervals to construct a manifold mesh of polygons representing the desired iso-surface at a specified resolution. The density of the new segment of the mesh is equal to the mean value of the original mesh. Facet vertices are ordered such that the cross-product of adjacent edges (the facet normal) is consistent with the gradient of the density function S. The marching cubes algorithm is a well-known general purpose algorithm and we use it to fill the holes as illustrated in Figure 16.

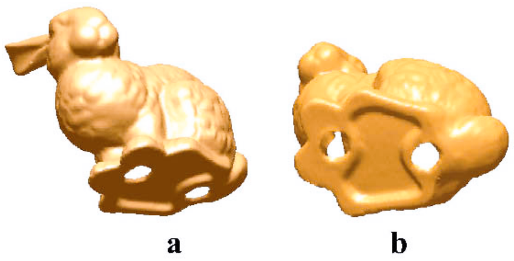


Figure 16. a) and b) show the filled holes identified in Section 3.1.

4. Experiments and Results

All tests were made with a computer with a 3.0GHz processor, 1.0Gb of ram memory running under the Microsoft XP operating system. The implementations of the models were made in C++ and MATLAB. In addition, a graphical motor in OpenGL was programmed to obtain the graphical representation of the images. The data were acquired with the Kreon sensor available in the Advanced Man-Machine Interface Laboratory−Department of Computing Science, University of Alberta, Canada.

In order to calibrate the model and to validate the correct behavior of the interpolator over 3D points, several tests were made on synthetically generated holes. The generation of synthetic holes is necessary to evaluate the quality of the points obtained with the interpolator since in real cases, the degree of precision of the points generated with respect to the real section of the surface is impossible to measure. The test consisted of generating synthetic holes on real range image, extracting a near points neighborhood by means of a kd−tree structure. Next, the hole is filled with the proposed strategy and the error of fit between the extracted data of the real surface and the new points is measured. The error reported in Table 1 corresponds to the error of the distance between both sets of points. Figure 17, shows the variation of neighborhood size to fill the hole.

Table 1. Variation of neighborhood size to fill the hole.

Number of Points	**Size of Hole=50**	**Surface size = 6051**
Neighborhood Size	**error**	**% of Surface**
100	1.79E-2	1.65%
200	1.60E-2	3.31%
500	1.03E-3	8.26%
1000	0.83	16.53%

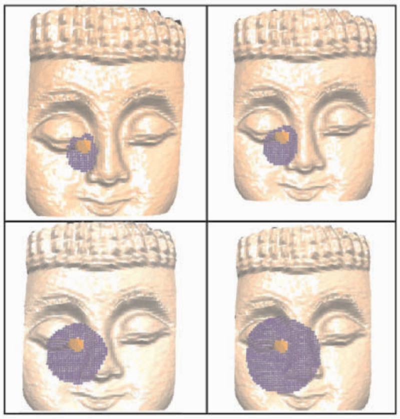


Figure 17. Different neighborhood sizes.

To show the behavior of our algorithm one can see in Figures 18 and 19 results for several real 3D cases. These images show that the algorithm generates smooth segments to fill the surface holes in different configurations and correctly identifies each one of the holes.

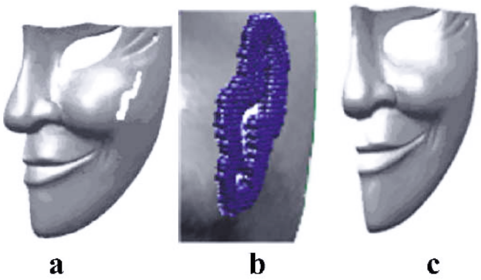


Figure 18. a) Original mesh, b) set of center and c) hole filled with RBF interpolant.

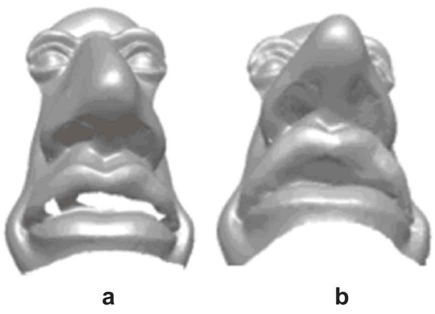


Figure 19. a) Original mesh, b) hole filled with RBF interpolating function.

5. Conclusion

We have presented a new technique for filling holes in a triangulated model using a local radial basis function interpolant defined for each one of the holes detected using the torsion fluctuation around the contour curve. In the algorithm, each contour curve is approximate with a Bézier curve segment from which the torsion is calculated analytically. The method is simple and effective, since the radial basis function fits the surface smoothly and always generates a closed manifold triangular mesh.

When big holes are present in a mesh, the interpolating function cannot adequately fit the surface. No big holes can exist if a good scanning process is done, that is, holes whose sizes do not exceed 3% of the total size of the mesh. Our algorithm has only one parameter: the predefined threshold for the variation of the torsion to determine if a hole must be filled. The other values like the size of vicinity are automatically calculated.

The threshold for the variation of the torsion to identify a hole in the mesh will be affected by the sample density and the method to make the mesh. So an approach to automatically define this value from a given mesh would be desirable. In some cases the holes would be on a plane. For these cases our algorithms must complete with a normal variation analysis of limit edges; but this is not a general case because the holes too often are caused by occlusion and are not trivial surfaces.

References

1. Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D., and Silva, C., "Point Set Surfaces", *IEEE Visualization,* pp. 21–28, 2001.
2. Amenta, N., Bern, M., and Kamvysselis, M., "A new Voronoi-based surface reconstruction algorithm", *SIGGRAPH'98*, pp.415–421, 1998.
3. Bajaj, C., Bernardini, F., and Xu, G., "Automatic Reconstruction of Surfaces and Scalar Fields from 3D Scans", *SIGGRAPH'95*, pp.109–118, 1995.
4. Bertalmio, M., Shapiro, G., Caselles, V., and Ballester, C. "Image Inpainting", *SIGGRAPH'00*, pp.417–424, 2000.
5. Besl, P., "Advances in Machine Vision", *Advances in Machine Vision,* Springer Verlag, Chapter 1 – Active Optical Range Sensors, pp. 1–63, 1989.
6. Besl, P. and McKay, N., "A method for registration of 3D shapes", *IEEE Trans. on PAMI*, 14 (2). pp. 239–256, 1992.

7. Carr, J., Beatson, R., Cherrie, J., Mitchell, T. Fright, W., and McCallum, B., "Reconstruction and Representation of 3D Objects with Radial Basis Functions", *SIGGRAPH' 01*, pp. 67–76, 2001.
8. Curless, B. and Levoy , M., "A Volumetric Method for Building Complex Models from Range Images", *SIGGRAPH' 96*, pp. 303–312, 1996.
9. Davis, J., Marschner, S., Garr, M., and Levoy, M., "Filling Holes in Complex Surfaces Using Volumetric Diffusion" *Proc. First International Symposium on 3D Data Processing, Visualization, Transmission*, pp. 428–861, 2002.
10. Efros, A. and Freeman, W., "Image Quilting for Texture Synthesis and Transfer", *SIGGRAPH'01*, pp. 341–348, 2001.
11. Gopi, M. and Krishnan, S., "A Fast and Efficient Projection Based Approach for Surface Reconstruction", *Intern. Journal of High Performance Computer Graphics, Multimedia and Visualization*, 1 (1), pp. 1–12, 2000.
12. Hoppe, H., DeRose, T., Duchamp, McDonald, T. J.A., and Stuetzle, W., "Surface reconstruction from unorganized points", *SIGGRAPH'92,* pp. 71–78, 1992.
13. Lancaster, P. and Salkauskas, K., *Curve and Surface Fitting: an Introduction.* Academic Press. 1986.
14. Lorensen, W. and Cline, H., "Marching cubes: A high resolution 3D surface construction algorithm", *Proc. SIGGRAPH'87*, pp. 163–169, 1987.
15. McAllister, D., Nyland, L., Popescu, V., Lastra A. and McCue, C., "Real Time Rendering of Real World Environments", *Proc. Rendering Techniques'99*, pp. 145–60, 1999.
16. Nyland, L., et al., "The Impact of Dense Range Data on Computer Graphics", *Proceedings of Multi-View Modeling and Analysis Workshop*, pp. 3–10, 1999.
17. Nyland, L., Lastra, A., Mc Allister, D., Popescuand, V., McCue, C., and Fuchs, H., "Capturing, Processing and Rendering Real-World Scenes", In *Videometrics and Optical Methods for 3D Shape Measurement, Electronic Imaging,, Photonics West*, SPIE Vol. 4309, pp. 107–116, 2001.
18. Oliveira, M., Bowen, B., McKenna, R., and Chang, Y., "Fast Digital Image Inpainting", *International Conference on Visualization, Imaging and Image Processing (VIIP 2001)*, Marbella, Spain, pp. 261–2665, 2001.
19. Hartigan, J. and Wong, M. A., "A K–Means Clustering Algorithm", *Journal of Applied Statistics,* 28 (1), pp. 100–108, 1979.
20. Pulli, K., "Multiview Registration for Large Data Sets", *3DIM'99*, pp. 160–168, 1999.
21. Wang, J. and Oliveira, M., "Improved Scene Reconstruction from Range Images", *Proc. EUROGRAPHICS' 2002*, pp. 521–530, 2002.
22. Wei, L.Y. and Levoy, M., "Texture Synthesis over Arbitrary Manifold Surfaces", *SIGGRAPH'01*, pp. 355–360, 2001.
23. Ying, L., Hertzmann, A., Biermann, H., and Zorin, D., "Texture and Shape Synthesis on Surfaces", *Eurographics'2001, Rendering Workshop,* pp. 301–312, 2001.
24. Yu, Y., Ferencz, A., and Malik, J., "Extracting Objects from Range and Radiance Images", *IEEE Transactions on Visualization and Computer Graphics*, 7 (4), pp. 351–364, 2001.
25. Carr, J. C., Beatson, R. K., McCallum, B. C., Fright, W. R., McLennan, T. J., and Mitchell, T. J., "Smooth surface reconstruction from noisy range data", *Proceedings of the 1st international conference on computer graphics and interactive techniques in Australasia and South East Asia*, Melbourne, Australia, *February 11 –14,* ACM Press, pp. 119–ff,2003.
26. Buhmann, M., "Radial Basis Fuction: Theory and Implementations", *Cambridge Monographs on Applied and Computational Mathematics*, 2003.

27. Carr, J.C., Beatson, R.K., Cherrie, J. B., Mitchell, T.J., Fright, W. R., McLennan, T.J., and Evans, T.R., "Reconstruction and representation of objects with radial basis function", Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pp. 67–76, 2001.
28. C. Montani, R. Scateni, R. Scopigno, A modified look-up table for implicit disambiguation of marching cubes, Visual Comput. 10 (1994) 353–355.