Filling Holes in 3D Meshes using Image Restoration Algorithms

Santiago Salamanca, Pilar Merchán
E. de Ingenierías Industriales
Universidad de Extremadura
Avda. de Elvas s/n, Badajoz, Spain
{ssalaman, pmerchan}@unex.es

Emiliano Pérez
E. T. S. de Ingeniería Informática
Universidad Nacional de Educación a Distancia
C/ Juan del Rosal 16, Madrid, Spain
emiliano@issi.uned.es

Antonio Adán
E. S. de Ingeniería Informática
Universidad de Castilla la Mancha
P. de la Universidad 4, Ciudad Real, Spain
Antonio.Adan@uclm.es

Carlos Cerrada
E.T.S. de Ingeniera Informática
Universidad Nacional de Educación a Distancia
C/ Juan del Rosal 16, Madrid, Spain
ccerrada@issi.uned.es

Abstract

This work describes a method for filling holes in a 3D mesh based on 2D image restoration algorithms. Since these algorithms need an image as input, the first stage of the method concerns a 3D to 2D transformation. By projecting the 3D surface onto a squared plane, a 2D image is generated in such a way that the depth information is stored in each grid. The image restoration algorithms are applied to it. Once the image has been repaired, the inverse transformation 2D to 3D is performed and the repaired 3D surface recovered. To test the method, artificial holes have been generated on a set of 3D surfaces. The goodness of the results has been measured from the comparison between the 3D original surfaces and the 3D repaired ones. An evaluation with commercial software has been carried out to show the validity of the method. The image restoration algorithms have been applied to 3D cultural heritage modeling with good results. Specifically, sculptures of the collection from the National Museum of Roman Art in Spain have been reconstructed.

1 Introduction

The complete 3D surface reconstruction from the geometry information acquired with a 3D scanner involves several stages. The first one is the *register* of the partial views. In it, a common reference system is calculated for all of them. Next, the *integration* of the transformed partial views is tackled to obtain an unique mesh. Besides, a 3D surface processing that includes smoothing, remeshing or hole filling must be applied.

Figure 1. Mesh with holes belonging to the sculptural group of Eneas from National Museum of Roman Art in Spain

This work is about this last action: filling holes or surface reparation. There are two facts that lead to the appearance of holes in a mesh. One is the incorrect redefinition or loss of data happened in the register and/or integration process. This kind of holes does not involve a hard problem and they are filled without consider the surface information of the surroundings. The second fact is the loss of geometric information, for example, during the acquisition process. In this case, the holes are greater and the neighboring area must be used to fill them. Figure 1 shows a mesh with several large holes.

A *watertight mesh*, which is without holes, is a mani-

fold mesh. There are many applications that need manifold meshes: rapid prototyping, metrology, 3D model for computer vision or computer graphics tasks, etc. On the other hand, hole filling is one of the most tedious and less autonomous tasks in mesh processing. For all these reasons, the development of new methods that solve this problem in an optimal and automatic way is very important.

Methods for filling holes in 3D surfaces can be classify as:

1. Methods in which the hole filling is an implicit task in the generation of the 3D model.

2. Methods in which the hole filling is an independent process of the 3D mesh generation.

The input for the first kind of methods is the 3D coordinates of the surface with no topological information associated. From this data a representation with no holes (mesh, Bezier, implicit function...) is created. For example, in [2] an implicit function estimated from volumetric representation of the range data has been used to produce the mesh. This function is employed in the Marching Cube algorithm [10] to obtain the watertight mesh. In [4] a method of simplex meshes (meshes with constant topological properties) approximation to the range data of an object is presented. A set of topological operations that allow to obtain the mesh of any free form object is also put forward. A technique based on a volumetric method for reconstructing watertight triangle meshes from arbitrary, not oriented clouds of points, without use a sign distance field is presented in [7].

The second group of hole filling algorithms is more used than the first one. That is because with these algorithms, the 3D mesh modeling process becomes quite apart from the hole filling procedure, what allows the use of more flexible mesh restoration methods

Davis et al. [3] present an approach based on the definition of an implicit function defined in the surrounding of the hole to be filled. A diffusion process that extends the area over the volume is applied on this function. This simple method can generate manifold surfaces and it is very efficient for big meshes or meshes with high resolution because of its locality. In [5] a variant of this method is proposed.

Nooruddin and Turk [12] expound a method starting from the voxelization of the mesh. The algorithm is used for hole filling, model reparation (creating manifold connectivity) and mesh simplification. Its drawback is that it does not allow filling holes in surfaces with high curvature.

Wang and Oliveira [14] use the range data of an object to create an intermediate representation, a triangular mesh that is used to detect the holes from the edges of the mesh. Afterwards, a group of neighbor points to these edges is taken and an interpolation based on MLS (Moving Least Squares) [9] applied.

(a) Image taken from [13]

(b) Image taken from [1]

Figure 2. Image reparation examples.

All the methods cited above have a common feature: they have no general validity, that is, they can offer good solutions for some kinds of holes, but not for all. Since it is proved that some image restoration algorithms provide good results for a large variety of situations, we have applied to 3D mesh repair.

Two image restoration algorithms have been used in this work, both with no modifications for range images. The first one, proposed by Roth and Black [13], extends traditional Markov Random Field (MRF) models by learning potential functions over extended pixel neighborhoods. Field potentials are modeled using a Products-of-Experts framework [6]. This model has been trained on 2000 randomly cropped image regions that are taken from fifty images from the Berkeley Segmentation Database (natural scenes, people, buildings, etc.) [11]. The resulting Fields-of-Experts model can be used in denoising and inpainting applications. Figure 2(a) shows the result of this inpainting scheme in a text removal application.

The second algorithm is the one's by Criminisi et al. [1] and combines the advantages of two approaches: "texture synthesis" algorithms for generating large image regions from sample textures, and "inpainting" techniques for filling in small image gaps. The former has been demonstrated for textures - repeating two-dimensional patterns with some stochasticity; the latter focus on linear structures which can be thought of as one-dimensional patterns, such as lines and object contours. Figure 2(b) exhibits the kind of reparations achieved by this algorithm.

The paper outline is as follows: section 2 presents the stages of the procedure for filling holes using image restoration techniques. The details of the 3D surface transformation into a 2D image are described in section 3. Section 4

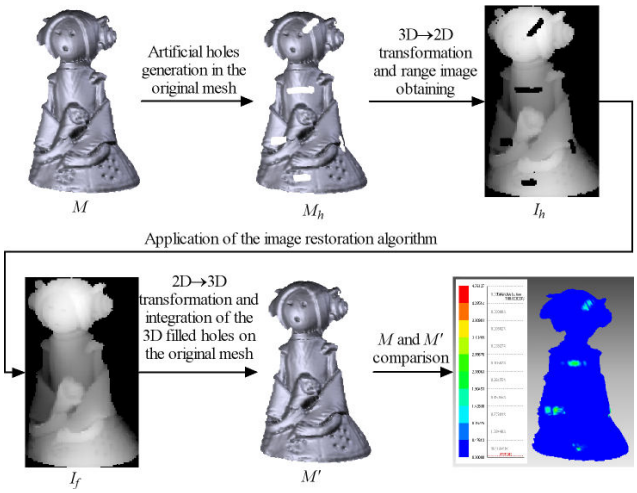


Figure 3. Procedure for 3D surface filling holes and evaluation of image restoration algorithms.

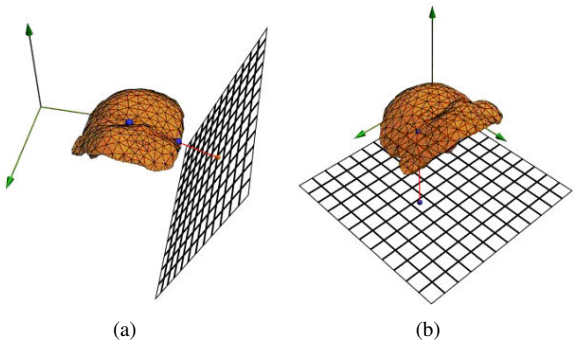


Figure 4. The projection plane has influence on the 2D representation of the 3D surface. If the plane is chosen as the one in (a), M_h is not recovered in a proper way. On the other side, with the projection plane shown in (b), M_h is retrieved successfully.

is devoted to show the experimental results. These results are compared to those obtained by a commercial software to assess our proposal, that has been applied to the reparation of meshes acquired from the sculptural collection of the National Museum of Roman Art. Finally, conclusions and future work are specified.

2 Outline of the procedure

The purpose of this paper is to show the method developed for filling holes in 3D surfaces based on image restoration algorithms. Several algorithms have been taken into account and their results for the 20 meshes in figure 7 compared to choose the best one. The method comprises five stages:

1. Creation of artificial holes in a database of 3D meshes.
2. 3D $\rightarrow$ 2D transformation to obtain a image associated to a mesh.
3. Application of the restoration image algorithm.
4. After a 2D $\rightarrow$ 3D transformation, merging of the filled hole and the mesh obtained in stage 1.
5. Comparison between nodes builded in stage 4 and those that belong to the original mesh.

Figure 3 illustrates this procedure. The original mesh, labelled M, is subjected to a process of hole creation to provide M_h. In the next step a range image, I_h, is generated by projecting every point of the 3D mesh onto a plane and associating a depth value to every pixel of the projection. Range image I_h is the input data for the restoration image algorithms proposed in[1, 13]. As said, these algorithms have been taken directly , with no modifications for the range image. The filled image is denoted I_f. Fourth stage consists of the 3D transformation of the repaired patches in I_f to get M_f, which is merged with M_h to provide the resulting mesh M'. Finally, M' is compared with M to validate the goodness of the applied algorithm.

3 Range image generation

Among the stages list above, range image generation is critical. The 3D$\rightarrow$2D transformation supplies a image, I_h, that codifies the information of height for the 3D points belonging to the mesh to be repaired, M_h. This information is stored as grey levels in the map of bits. There are two key points to decide before performing the transformation:

1. Choice of the projection plane.
2. Choice of the image resolution.

3.1 Projection plane

The first decision to be made is the choice of the plane which M_h will be projected on. It is clear that this plane selection has large influence on the representation that the 2D image offers about the 3D information. Figure 4(a) shows a wrong choice, since the projections of the two marked points fit in the same point of the plane; it leads to a multivaluated function. This situation does not take place in figure 4(b).

In this work only partial views of the object captured with a 3D laser scanner have been used. It provides the 3D

coordinates of the mesh vertex with respect to its own reference system $S = \{OXYZ\}$, with axis OZ defined by the vision axis of the 3D scanner. Under these circumstances, the selection of plane XY as the suitable projection plane, which does not guide to ambiguous situations, is immediate.

3.2 Image resolution

When the projection plane has been chosen, the next step is the projection of M_h on it. This plane must be discretized to obtain the range image I_h. Although the rasterization of a mesh is always possible, it is necessary to define an optimal resolution. A low resolution leads to a degeneration of the 2D representation of M_h. Otherwise, a high resolution provides an amount of information (pixels) that causes the inefficiency of the algorithm in terms of time. Therefore, both situations must be avoided and the selection of the resolution becomes a critical matter that affects the image restoration process, that is to say the 3D hole filling process. Figure 5 illustrates these facts by showing three projections of M_h with three different resolutions (low, optimal and high).

To calculate the optimal resolution the size of the image in millimeters must be known. This size is a constant for a given mesh. Following the explanation in paragraph 3.1, if x_{max} and y_{max} are the maximum coordinate values of the nodes $n_i \in M_h$, and x_{min} and y_{min} are the minimum values, the size of the image is $w \times h \ (mm^2)$, where:

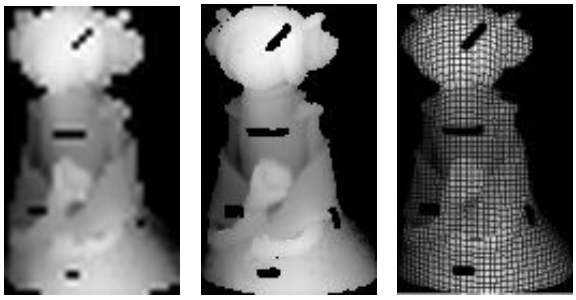
$$\begin{aligned} w &= x_{max} - x_{min} \\ h &= y_{max} - y_{min} \end{aligned} \tag{1}$$

This area is squared, and every grid becomes a pixel of the image I_h. The number of grids determines the resolution and the discretization of the projection M_h.

If the grid width is q $(mm/pixels)$, the image resolution is $1/q$ $(pixels/mm)$ and its size is $(n,\, m) = (w/q \times h/q)$ $(pixels)$. The pixel (i, j) stores the depth information, that is, the coordinate z_i of the node $n_i \in M_h$ which coordinates $(x_i,\, y_i)$ verify:

$$\begin{aligned} (i-1)\cdot q &\leq x_i < i \cdot q \\ (j-1)\cdot q &\leq y_i < j \cdot q \end{aligned} \tag{2}$$

The smaller the image resolution, the bigger the pixel size in millimeters and the number of nodes in M_h that verify condition (2). The pixel (i, j) stores the smallest z_i coordinate value among the z values of all the nodes that verify (2). The situation of several nodes belonging to the same pixel must be avoided, as it gives rise a loss of surface information. On the other hand, if the resolution is very high, the pixel size is reduced and maybe some pixels do not store any depth value, which causes empty regions turn up in the image. These regions could be filled by means of a mesh rasterization algorithm, however, as said, it is necessary to start with a resolution as good as possible to prevent the degeneration of the image I_h.



(a) (b) (c)

Figure 5. Projection of M_h to generate I_h. Three different image resolutions are shown. Note that for low resolution (a) a lot of detail is lost and for high resolution (c) empty regions appear. In both cases restoration algorithms do not work properly.

To fix the best resolution for image generation, the *optimal occupation per pixel* has been defined:

Definition 1 *The pixel $p_{ij} \in I_h$ is optimally occupied if exits one and only one node $n_i \in M_h$ that verifies (2) for this pixel.*

After this definition, the *Optimal Occupation Ratio*, O_{I_h}, which is the ratio between the number of pixels that validate definition 1 and the total number of pixels in the image, can be calculated:

$$O_{I_h} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} V_{ij}}{n \cdot m} \tag{3}$$

where $n{\cdot}m$ is the number of pixels of I_h and V_{ij} is a variable that points out if the pixel p_{ij} is optimally occupied. It can be formally defined as:

$$V_{ij} = \begin{cases} 1 & \text{Iff the pixel } p_{ij} \text{ is taken up optimally} \\ 0 & \text{Otherwise} \end{cases} \tag{4}$$

This way, O_{I_h} must be a small value for small image resolutions. As resolution is increased, the number of optimally occupied pixels is, and so does O_{I_h} up to a value for which every pixel is optimally taken up. From this point, O_{I_h} is decreased again, since the number of pixels in the image is increased while the number of optimally occupied

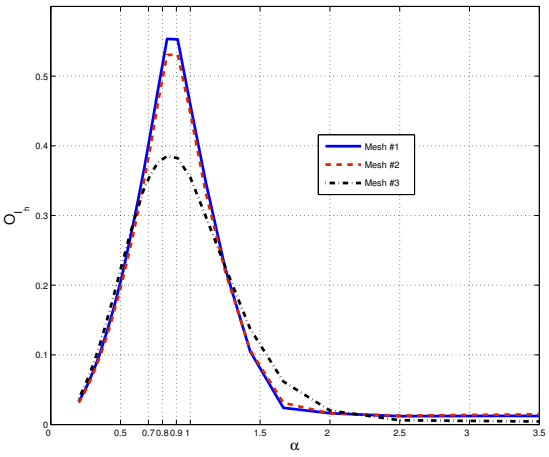


Figure 6. O_{I_h} vs. α depicted for three meshes.

pixels is not. The value of the resolution that must be chosen is what supplies a maximum for O_{I_h}.

To elucidate the best value for the size of the pixel (that is equivalent to the value of the image resolution), O_{I_h} has been calculated with several values of q_{I_h} for every mesh used in the experimentation. To make this calculation independent of the mesh characteristics, q_{I_h} has been defined as proportional to the mesh edge average length, d_m:

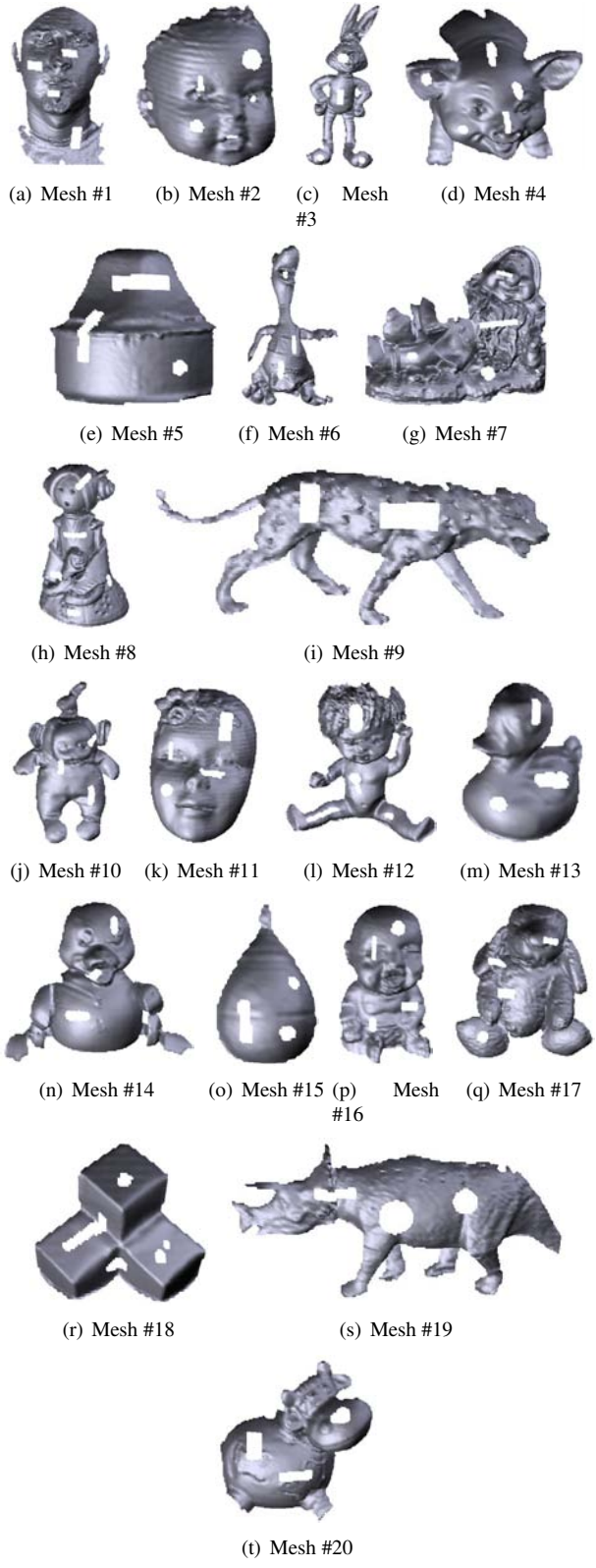
$$q_{I_h} = \alpha \cdot d_m \tag{5}$$

where α is the proportional factor and represents the variable related to O_{I_h}.

The value of α that causes a maximum of O_{I_h} has been obtained after a statistical analysis of the results provided by all the meshes we have worked with. This value is α =0.87, and it is the number that has been used to generate the image that will be repaired with the image restoration algorithms. In figure 6 the graphics for three different meshes are shown. In all of them, the maxima are easily identify.

4 Experimental results

Once the image has been restored and the repaired holes have undergone the inverse transformation 2D to 3D to be merged with the defective mesh and provide the repaired M', it is time to verify the results. This verification is made by comparing the data of the original mesh, M, with the data of the repaired one, M'. The comparison is based in the measurement of the Euclidean distance between the nodes belonging to M' and their related nodes of M. Obviously, this distance is 0 except for the nodes that fit in the filled holes.



(a) Mesh #1 (b) Mesh #2 (c) Mesh #3 (d) Mesh #4

(e) Mesh #5 (f) Mesh #6 (g) Mesh #7

(h) Mesh #8 (i) Mesh #9

(j) Mesh #10 (k) Mesh #11 (l) Mesh #12 (m) Mesh #13

(n) Mesh #14 (o) Mesh #15 (p) Mesh #16 (q) Mesh #17

(r) Mesh #18 (s) Mesh #19

(t) Mesh #20

Figure 7. Meshes with holes used to test the different algorithms.

Table 1. Mean distance between the nodes of M' y M for the three algorithms

Number of the mesh	Roth alg.	Criminisi alg.	Comercial alg.
1	0,9347	1,5342	1,1498
2	0,4521	0,6019	0,5226
3	0,7742	4,8585	0,7220
4	0,4127	0,9938	0,4575
5	0,6400	0,9855	0,6949
6	0,7153	2,2741	0,8207
7	0,9653	1,6980	0,8808
8	0,7045	2,1069	0,8516
9	0,8029	2,7879	1,0892
10	0,6166	2,6012	0,7571
11	0,7113	0,9590	0,7526
12	0,5521	2,6701	0,4256
13	0,5159	0,8566	0,4512
14	0,4830	1,0492	0,5755
15	0,3536	0,5818	0,3897
16	0,5307	1,3913	0,5927
17	0,8501	3,6414	0,6008
18	0,4349	1,4860	0,4885
19	0,9262	2,0049	1,2902
20	0,5870	4,4610	0,6276

The method has been tested with a database of partial views from 20 freeform objects sensed with a Minolta VI-300 scanner. The meshes have been damaged with a number of holes, from 2 up to 6, all with significant sizes and different kinds of surfaces (smooth surfaces, surfaces with sharp edges, flat surfaces, etc.), that have been restored by using the algorithms [1] and [13].

The experimentation tries to elucidate not only if it is possible to fill holes on a 3D surface by means of image restoration algorithms, but also the usefulness of this methodology. To prove this point, the holes have also been filled using a commercial software and the results have been included in the comparison. Figure 7 illustrates a picture of the meshes with holes we have been working with.

Table 1 summarizes the numerical results obtained. It shows, for every mesh, the mean distance between the nodes defined in M' by the filling algorithm and their related nodes in M. The second column of the table contains the results for algorithm [13]. The values provided by algorithm [1] are listed in the third column, whereas the figures achieved with the commercial software appear in column four.

The results are quite good for the algorithm in [13], since the mean distance is the smallest in 75% of the meshes. On the other hand, algorithm [1] offers the worst numbers for all the cases. Thus, the commercial software supplies the best solution for 25% of the meshes.

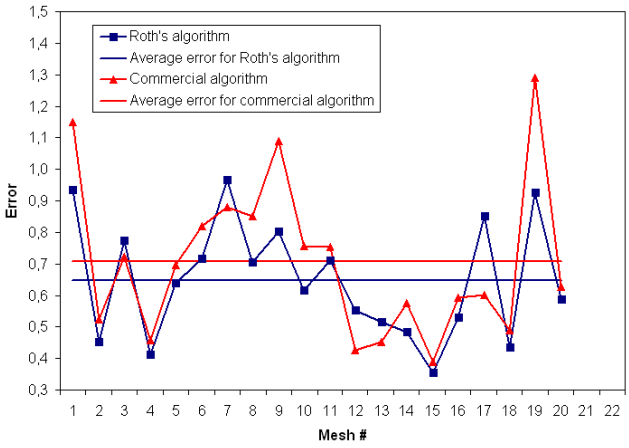


Figure 8. Error vs. mesh number for the Roth's (in blue) and commercial (in red) algorithms.

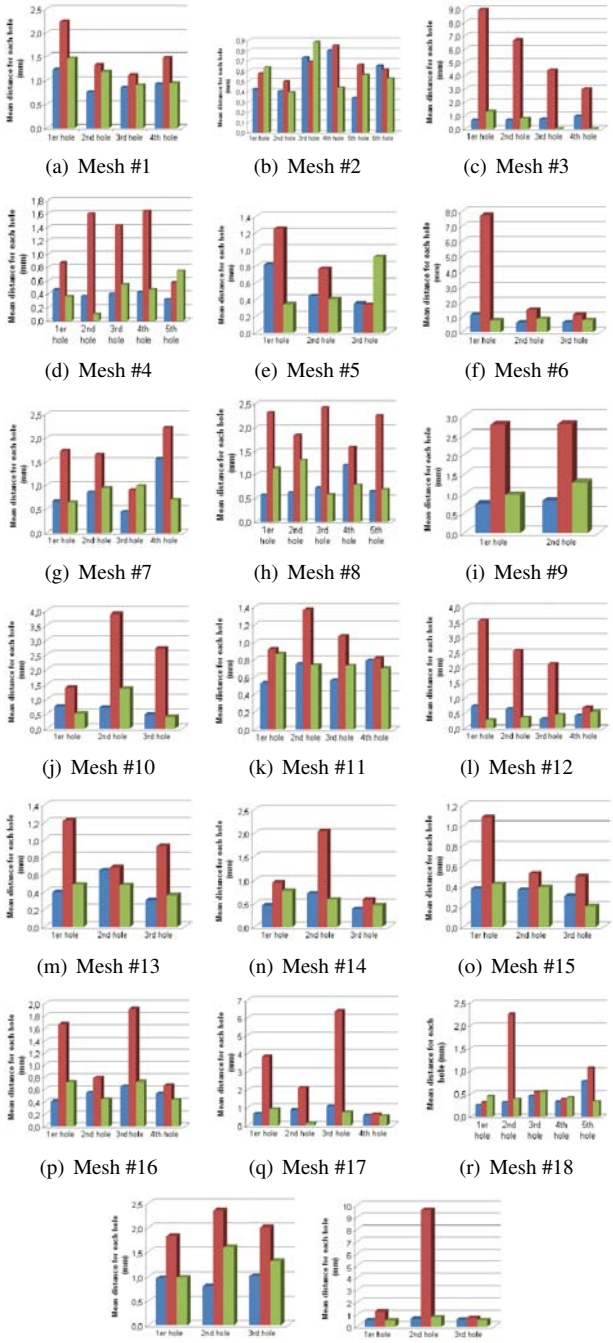
Figure 8 illustrates the data obtained with algorithm [13] and the commercial software together with the average values of both two series ($Av_{[13]}$=0.65, $Av_{ComSoft}$=0.71), given that the data provided by algorithm [1] are clearly worse ($Av_{[1]}$=1.98).

The mean distance between holes, that is, the distance between the nodes generated to fill one hole in M' and the corresponding nodes in M, and this for every hole in the meshes, has also been calculated. This is a more significant error than the former since the surfaces are more homogeneous per hole than per mesh. The best algorithm will be that which better fits all kind of surfaces. Figure 9 depicted the results achieved using bar graphics. In them, y axis is for the mean distance between nodes in a hole and x axis for the algorithms: the first bars (in blue) for algorithm [13], the second bars (in red) for algorithm [1] and the third bars (in green) for the commercial software. Algorithm [13] demonstrates to be the best again. It gives the best results in 56% of the times, algorithm [1] works better in only 2.67% of the cases and the commercial software in 41.33% of them.

Figure 10 shows a picture of the restoration performed on several meshes from the sculptural collection of the National Museum of Roman Art in Spain, using the expounded method and algorithm [13]. The first one corresponds to the mesh in figure 1.

5 Conclusions and future works

This work has proved that using image restoration algorithms to fill holes in 3D meshes can be valid. To do



(a) Mesh #1 (b) Mesh #2 (c) Mesh #3

(d) Mesh #4 (e) Mesh #5 (f) Mesh #6

(g) Mesh #7 (h) Mesh #8 (i) Mesh #9

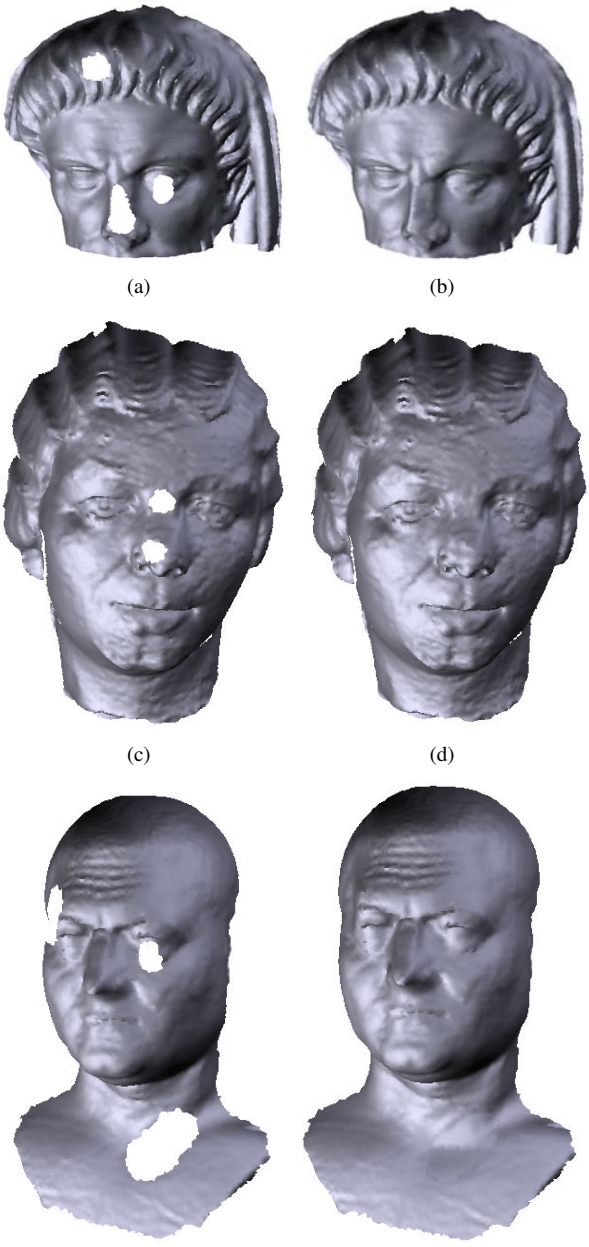
(j) Mesh #10 (k) Mesh #11 (l) Mesh #12

(m) Mesh #13 (n) Mesh #14 (o) Mesh #15

(p) Mesh #16 (q) Mesh #17 (r) Mesh #18

(s) Mesh #19 (t) Mesh #20

Figure 9. Error per hole for the 20 meshes in figure 7



(a) (b)

(c) (d)

(e) (f)

Figure 10. Original and repaired meshes from the sculptural collection of the National Museum of Roman Art in Spain.

this, firstly the generation of a 2D image that is the input to the restoration algorithm is needed. The proposed method calculates the optimal resolution so that the losses after the projection of the data are minimum. Furthermore, it is independent of the mesh, due to the definition of the image resolution (5).

There are two important conclusions that stand out: Roth's algorithm [13] can be used for hole filling with better results than the ones offered by a commercial software. On the contrary, not all the image restoration algorithms are valid for hole filling. Thus, the algorithm by Crimisini [1] has not worked properly. That can be because this method uses an exemplar based technique that repairs (fills) by generating the new textures (the new depth values, for this work) by means of the sampling and copy of pixels from the image to be repaired. Image inpainting algorithms are, perhaps, more appropriated for hole filling since they use local information, the information that is close to the hole.

Future works will be focused in the use of geometrical and textural information as data managed by the hole filling algorithm. The training should also be modified for this new situation, since the one used in [13] was performed with natural images, which is not the kind of images it is going to be applied to. The method has been proved only for partial views of the object. We are currently working on a variation of the projection of M_h which takes into account complete meshes.

Acknowledgements

The authors would like to thank to the personnel of the National Museum of Roman Art in Spain their facility and support.

This work has been supported by the Spanish projects DPI2006-14794-C02, DPI2005-03769 (CICYT projects) and PCI08-0052-1401.

References

[1] A. Criminisi, P. Perez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on Image Processing*, 13(1):1200–1212, September 2004.

[2] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceeding of SIGGRAPH'96*, pages 302–312. ACM, 1996.

[3] J. Davis, S. R. Marschner, M. Garr, and M. Levoy. Filling holes in complex surfaces using volumetric diffusion. In *Proc. of the First International Symposium on* 3*D Data Processing, Visualization, and Transmission*, pages 428–438, Padua, Italy, 2002. IEEE.

[4] H. Delingette. General object reconstruction based on simplex meshes. *International Journal of Computer Vision*, 32(2):111–146, September 1999.

[5] T.-Q. Guo, J.-J. Li, J.-G. Weng, and Y.-T. Zhuang. Filling holes in complex surfaces using oriented voxel diffusion. In *International Conference on Machine Learning and Cybernetics*, pages 4370–4375, Dalian, China, August 2006. IEEE.

[6] G. E. Hinton. Products of experts. In *Proc. of the Ninth International Conference on Artificial Neural Networks*, volume 1, 1999.

[7] A. Hornung and L. Kobbelt. Robust reconstruction of watertight 3D models from non-uniformly sampled point clouds without normal information. In *ACM International Conference Proceeding Series; Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 264, pages 41–50, 2006.

[8] M. Jones, M. Jones, J. Baerentzen, and M. Sramek. 3d distance fields: a survey of techniques and applications. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):581–599, 2006.

[9] P. Lancaster and K. Salkauskas. Surfaces generated by moving least squares methods. *Mathematics of Computation*, 37(155):141–158, July 1981.

[10] W. E. Lorensen and H. E. Cline. Marching Cube: a high resolution 3D surface construction algorithm. In *Proceeding of SIGGRAPH '87*, pages 163–168. ACM, July 1987.

[11] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th International Conference on Computer Vision*, volume 2, pages 416–423, 2001.

[12] F. S. Nooruddin and G. Turk. Simplification and repair of polygonal models using volumetric techniques. *IEEE Transactions on Visualization and Computer Graphics*, 9(2):191–205, April-June 2003.

[13] S. Roth and M. J. Black. Fields of experts: A framework for learning image priors. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 860–867, San Diego, California, June 2005. IEEE.

[14] J. Wang and M. M. Oliveira. Filling holes on locally smooth surfaces reconstructed from point cloud. *Image and Vision Computing*, 25(1):103–113, January 2007.