# Visibility-Based Feature Extraction From Discrete Models

Antoni Chica*

Department of Software

Polytechnic University of Catalonia

## Abstract

In this paper, we present a new visibility-based feature extraction algorithm from discrete models as dense point clouds resulting from laser scans. Based on the observation that one can characterize local properties of the surface by what can be seen by an imaginary creature on the surface, we propose algorithms that extract features using an intermediate representation of the model as a discrete volume for computational efficiency. We describe an efficient algorithm for computing the visibility map among voxels, based on the properties of a discrete erosion. The visibility information obtained in this first step is then used to extract the model components (faces, edges and vertices) —which may be curved— and to compute the topological connectivity graph in a very efficient and robust way. The results are discussed through several examples.

**CR Categories:** I.3.5 [Computer Graphics]: Curve, surface, solid, and object representations.—;

**Keywords:** point clouds, voxels, isosurfaces, segmentation, feature identification

## 1 Introduction

Todays laser scanning devices provide very dense 3D point clouds, from which the geometrical and topological structure of the model must be inferred. This is an important problem in model acquisition and reverse engineering. In this paper we address the problem of feature lines extraction from these discrete models and the reconstruction of their connectivity. Feature lines may be defined via the extrema of principal curvatures along corresponding principal directions [Demarsin et al. 2007].

If we imagine a tiny creature crawling on the surface, and we imagine it can only see along the surface, then it will only be able to see a portion of the surface that depends on its geometry. For example, while walking on a planar face, it will see until the confines of the face, but no further. However, when reaching an edge, the area that its sight covers suddenly increases, since a full new face enters into view. Even more so if it reaches a vertex. In this paper we implement this idea, approximating these areas in sight by constructing a voxelized approximation of the surface, and by counting the number of visible voxels.

By using an intermediate volume discretization, we are able to exploit a number of properties of erosion-based visibility for discrete uniform models, and we can obtain a visibility map in a reasonable computing time. The visibility information can then be used for extracting regions and feature lines of the scanned object.

*e-mail: achica@iri.upc.edu

Our proposal includes the following main contributions:

- A new algorithm for the efficient computation of the visibility map among the mixed cells of a discrete voxel model is presented. The algorithm exploits the regularity of the discrete model and can compute the visibility map in reasonable times.
- An algorithm for the extraction of model feature lines and regions based on the information of the visibility map is also presented. The algorithm is fast and can deal with curved regions and characteristics. A simple modification which allows the algorithm to deal with noise is also presented.
- As a by-result, the topological graph of the object surface is obtained. The graph represents the connectivity among vertices, edges and (potentially curved) faces.

## 2 Previous Work

Reconstruction from very dense point clouds is a well studied problem in computer graphics. There is a diverse set of methods which may be classified according to the structures used during the reconstruction process.

One possibility is to represent the surface as implicit functions, like the early work of Hoppe et al [Hoppe et al. 1992]. This technique computes an estimate of the normals of the surface using local principal component analysis. Then, a neighborhood graph is used to unify these normals, which in turn are used to obtain an implicit signed distance function. The final surface is extracted using Marching Cubes [Lorensen and Cline 1987]. Anisotropic basis functions [Dinh et al. 2001] have also been proposed to handle sharp feature preservation. Another possible way to represent the implicit functions is to use radial basis functions. Turk et al. [Turk and O'Brien 1999] use global radial basis functions which leads to quadratic complexity. Other works [Carr et al. 2001] reduce this complexity. Additionally, moving least squares approaches [Alexa et al. 2003] use the invariant set of a projection operator to define the extracted surface. Feature lines are preserved by excluding outliers from the fitting process, but their topology is not reconstructed explicitly.

Another set of methods [Amenta et al. 1998] [Amenta et al. 2001] consider the problem from a computational geometry point of view, applying voronoi techniques. Given a minimum sampling they guarantee a correct topology reconstruction, but they are heavily influenced by noise. However, recent extensions [Mederos et al. 2005] deal with this problem.

Feature topology extraction has been a useful preprocessing step for reconstruction algorithms. The two-stage process of Gumhold et al [Gumhold et al. 2001], recovers the model's feature lines without requiring any previous reconstruction of the point cloud. First, it assigns penalty weights to edges on a neighbor graph which yields a set of feature patterns. Then, it extracts feature lines and junctions by fitting wedges to the crease lines. To improve the robustness in the presence of noise, Pauly et al [Pauly et al. 2003] use a multi-scale method for extracting line-type features from point clouds. Points are classified according to the likelihood of them

being part of a feature line by means of a surface variation estimation, which is based on covariance analysis of local neighborhoods. Hysteresis thresholding is applied in order to compute a minimum spanning graph. This initial approximation of the feature lines is then smoothed using an active contours technique. In [Demarsin et al. 2007] a curve network is built on a point cloud that defines the border of various surface patches. Normals computed using the k-neighbours of each point are used to segment the cloud into point clusters. From these clusters candidate feature points are extracted and processed as a graph. To recover the sharp feature lines, the pruned minimum spanning tree of the graph is obtained, which is treated with a reconnection and a smoothing processes.

Jenke et al [Jenke et al. 2006] propose a statistical Bayesian approach. Prior assumptions on the measured objects are modeled as probability distributions, which are used to infer a maximum probability reconstruction by means of Bayes' rule. As a result, both topology and geometry are extracted.

Algorithms that extract isosurfaces from a discrete volume objects [Lorensen and Cline 1987] strive to ensure topological consistency and geometry fidelity. This last property often depends on the delicate ability to recover sharp features and to smooth the isosurface away from these edges. The Extended marching Cubes detects cells containing features and recovers them by inserting an additional point in each of these cells [Kobbelt et al. 2001]. The Dual Contouring algorithm uses quadratic error metric for computing the point positions to obtain a locally smooth surface [Ju et al. 2002]. Note that all these methods try to recover a surface that looks correct but they don't explicitly extract its features nor its topological graph. Moreover, surface extraction algorithms usually make use of scalar field or Hermite data [Ho et al. 2005].

Meanwhile, feature extraction has also been an important topic in discrete volume models, due to the feature insensitivity of Marching Cubes. Kobbelt et al [Kobbelt et al. 2001] use an Extended Marching Cubes method to derive vertex normals from the original scalar field. These are used to decide whether a cell contains a sharp feature and to compute additional vertices on the features. Edge-Sharpener [Attene et al. 2004] introduces a filter which allows to identify the chamfers of the 3D triangle mesh. The triangles on these chamfers are subdivided and the new vertices are positioned to recover the model's features.

As we have already introduced, our approach to feature extraction is guided by a measure of visibility computed for the mixed cells of a voxel model. More precisely, we need to compute cell-to-cell visibility between a pair of mixed cells considering that while and black voxels are opaque. A number of from-region algorithms exist, but most of them are only applicable to a certain range of scenes or require complex computations [Cohen-Or et al. 2003]. We have considered that erosion-based methods [Wonka et al. 2000][Décoret et al. 2003] are the most appropriate for our objectives. The main idea is to shrink the occluders and occludees in such a way that cell-to-cell visibility can be conservatively computed by a simple point-to-point visibility test. This approach can be easily apply to our proposal where occluders and occludees are cubes. Although elegant approaches that compute exact cell-to-cell visibility have been proposed [Nirenstein et al. 2002][Durand et al. 2002] they are computationally expensive and the results that we obtain using conservative visibility have good accuracy and robustness.

## 3 Overview

As a preprocessing step, we immerse the data samples in a voxelization of a user-specified resolution, from which we extract a subset of six-connected cells (that we call the *discrete band*) that contain the samples. These cells are the mixed cells (those that stab the surface). The surface being sought must be contained in the set union of all these mixed cells. Depending on the nature of the data and the resolution of the voxelization, the set of voxels containing data may not be six-connected, and the extraction of this set may require a repair step like those proposed by [Hornung and Kobbelt 2006], [Amenta et al. 2001] or [Esteve et al. 2005].

Then, our algorithm proceeds in three steps:

- In the first one, the visibility map of the mixed cells is computed and the maximum isotropic visibility distance is stored for each of the mixed cells in the model. The algorithm, detailed in Section 4, uses an efficient computation of the cell-to cell visibility based on erosion and on specific properties of the uniform discrete representations.
- Then, the visibility information is used for the detection and extraction of the geometric components of the model surface: vertices, edges and faces. The algorithm uses local operations and presents a small time complexity. It is able to extract non-flat faces. These geometric components define the nodes of the topological graph of the object surface
- In a final step, we generate the arcs of the graph by detecting the connectivity among vertices and edges. The final result is the topological connectivity graph of the object and the identification of the components, regions and feature lines, of its surface.

## 4 Visibility Map

In order to identify all the components of the isosurface contained in the discrete band, we first compute what we call the visibility map. Before describing our definition of this function, let us consider what may seem a more natural translation of the intuitive ideas of the introduction to the present setting. To this end, consider an unconstrained visibility map, as an integer function $uVis(c)$ defined on the domain of mixed cells $c$. Assuming that mixed cells are transparent and that the remaining cells are opaque, the value $uVis(c)$ represents the number of mixed cells visible from $c$, with the standard cell-to-cell visibility definitions ([Nirenstein et al. 2002]).

Now, let us consider that we obtain the visible set from any particular mixed cell. In 2D, vertices of the original isocurve will be contained in mixed cells with larger visible sets, while edges will be inside cells with smaller visibility. This effect is illustrated in Figure 1. However, trying to extend this idea to 3D models brings new issues, as artifacts appear on models with holes as the one shown in Figure 2, originating changes in visibility where no features exist. The unconstrained visibility from a mixed cell on a pierced face is occluded by the hole itself, which causes these artifacts.

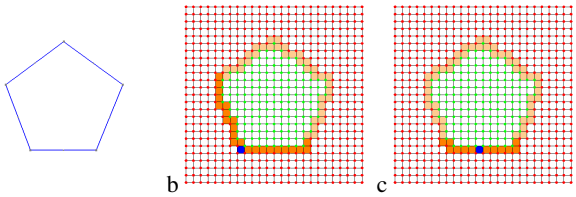


**Figure 1:** *(a) The discretization of a pentagon defines a set of cells which can be used to compute cell to cell unconstrained visibility. (b) Visibility from a cell containing a vertex results in the cells stabbed by its two adjacent edges, while (c) visibility from a cell on an edge results only in the cells stabbed by that edge.*

To overcome this problem, instead of using the size of the visible set of a mixed cell to detect features like vertices and edges, we use another visibility-based measure to divide the discrete band into

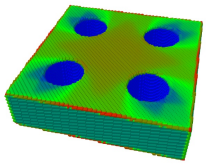

**Figure 2:** *The above figure is a render of all mixed cells of a model. Colors are used to represent different sizes of the corresponding visible set, with red representing the largest unconstrained visibility and blue the smallest.*

components. We define the visibility map as $Vis(c)$ which measures the number of complete rings of mixed cells centered at $c$ that are visible from this cell $c$. The first ring of a mixed cell $c$ is defined as usual as the set of mixed cells contiguous to $c$, and the $k-$ring is the set of cells being contiguous to the $(k-1)-$ring. Given one mixed cell, we first obtain the mixed cells in its 6-neighborhood, then the set of mixed cells on the 6-neighborhood of the previous set and so on. This process is stopped when a single cell in one of these rings is not visible from the source. The measure we take as a result is the radius of visibility of the source cell (the number of successive rings that are visible from it). The visibility map stores the radius of visibility of all mixed cells.

An interesting property of the visibility map may be noticed in Figure 3, which shows the visibility maps of an icosahedron and a mechanical part. Values are large at vertices, edges and the core of faces, while they become smaller at mixed cells near these features, isolating them from one another. This is caused by the mixed cells near an edge not being able to see the face on the other side.

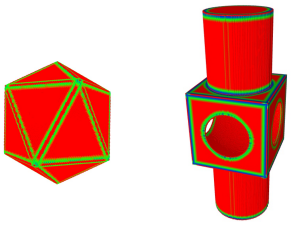

**Figure 3:** *Visibility maps of an icosahedron and a mechanical part where components are clearly identifiable. Red represents the largest visibility radius, while blue stands for the smallest. The maximum radius of visibility computed is 10.*

Now we need to determine how to know if two cells are visible or not (see Figure 4). In the figure, cells in light yellow are those between which we are trying to compute the visibility. A pair of points, one in each of these two cells is called a pair of witnesses if the segment joining them is completely contained in the union of the mixed cells. The figure on the left of 4 shows one such pair of witnesses, whereas the figure on the right has none.

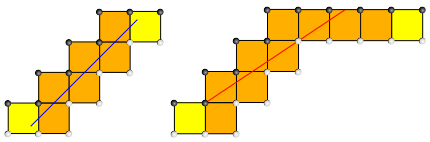

**Figure 4:** *2D examples of cell to cell visibility. In the picture on the left the two yellow cells see each other, while in the figure on the right they do not.*

If exact visibility between a pair of mixed cells is computed, even small models will take too much time to process. The strategy that is used here is to apply work on erosion-based conservative visibility by Decoret et al [Décoret et al. 2003], and adapt it to our setting. Consider an object $O$, in our case all of space except the discrete band, and a convex set of vectors $C$:

$$C = \{(x, y, z), |x| \leq \frac{1}{2} \wedge |y| \leq \frac{1}{2} \wedge |z| \leq \frac{1}{2}\} \qquad (1)$$

If a segment $PQ$ intersects $O \ominus C$, then any segment $P'Q'$ with $P' \in \{P\} \oplus C$ and $Q' \in \{Q\} \oplus C$ must intersect $O$. $O \oplus C$ stands for the Minkowski sum of both sets, while $O \ominus C$ defines the Minkowski erosion of $O$ by $C$. This property is illustrated in Figure 5, and it allows to reduce the cost of computing visibility between two cells to finding if the segment between their two centers intersects $O \ominus C$. It comes at the price of obtaining the erosion of an object, though.

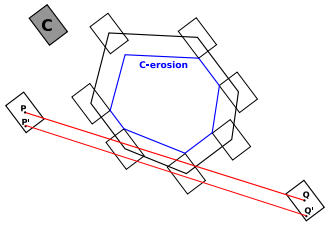


**Figure 5:** *If there is a segment from a point $P'$ in one cell to a point $Q'$ on another cell which does not intersect an object, then the segment between $P$ and $Q$, centers of their corresponding cells, does not intersect the C-erosion of the object. As a consequence, we do not miss any case of visibility by computing visibility between centers of cells against eroded volumes.*

Computing the exact erosion of general objects is a non-trivial task, but the object that concerns us is all of space except the discrete band and, as such, it is simpler to shrink. If we subdivide each cell into 8 equal subcells, $O \ominus C$ is all of space except those subcells obtained from mixed cells and those that are adjacent (through a vertex, an edge or a face) to a mixed cell. A 2D example of this is illustrated in Figure 6. Now, in order to check the visibility between two given mixed cells, we only need to test the segment which connects their centers against $O \ominus C$. We use a Bresenham-like algorithm to determine which subcells are crossed by the segment, so this is computed very efficiently.

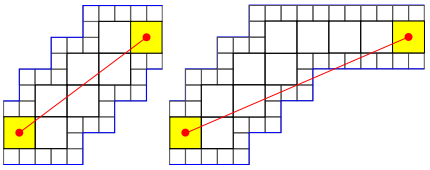

**Figure 6:** *Visibility computation for the examples shown in Figure 4. Testing a segment that connects two cell's centers for intersection with the eroded object (shown in blue) is enough using the erosion based technique.*

Using this technique to compute cell to cell visibility, we may now obtain the visibility map (like the ones in Figure 3). Further efficiency improvements are gained by using the fact that the model's components are separated by relatively small radius values. Instead of computing the visibility map, we may compute the *trimmed visibility map*, where for each mixed cell, its visibility radius computation is stopped as soon as it reaches a certain value. This value is

determined by the minimum radius we will apply to identify components in the next section. Moreover, coherency may be exploited by limiting the visibility radius computation of one mixed cell, to a maximum of the radius of a previously computed neighbor cell plus one.

## 5 Feature Extraction

Once the visibility map of a model has been computed, its components may be extracted by simply flooding the regions with a high enough visibility radius. What we do is to identify mixed cells as part of a component if their visibility radius is greater than a certain minimum. The chosen threshold has to result from a compromise between the necessity of separating components correctly, and the requirement of not missing any of them. Our experiments have generated good results using a minimum radius of 11. Figure 7 shows the results of this step on the icosahedron and the mechanical part models. Notice that while the values of the full visibility map at points in general position depend on the curvature of the face or edge and on the resolution of the voxelization, the way in which the visualization map dies off when approaching an edge is fairly insensitive to them (although the fall-off will happen at different distances). This is the reason why a value of 11 seems to be a good choice no matter the nature of the model or the resolution used.

Once the set of mixed cells has been divided into components, these need to be classified. We will separate them into those that contain a vertex, those which correspond to an edge, and those that represent the core of a face. The last two may be curved.

Smaller components correspond to vertex components and can be detected by a threshold (for the examples shown through the paper, components composed of 25 or less cells are classified as vertices). In order to distinguish between edges and faces, we compute, for every cell in a component, how many of its 6-neighbors are in the same group. Then, for each component, we take the mean of this value for all its cells. The resulting quantity measures the dimensionality of the related component, therefore it may be used to differentiate edges from faces. It results that this dimensionality is less than 3.5 for edge components, while it is greater than 3.5 for faces. The second column of Figure 7 is obtained as a result of this method.

The dimensionality-based analysis of the components results in the automatic detection of all the nodes of the topological connectivity graph of the object surface, namely its vertices and edges. What remains is to complete the graph by computing the arcs that represent the connectivity between graph nodes. The final step of the algorithm computes these connectivities and creates the complete graph of components, like those shown in Figure 7. The detection of connectivities is performed by looking for mixed cells that belong to edge components in the vicinity of vertex components. Vertex components are grown by means of a dilation operator. If the intersection of an edge component with the dilated vertex component is not empty, an arc between the two is added to the connectivity graph.

## 6 Dealing with noise

As presented so far, the method is capable of detecting and extracting features from noise-free models with accuracy. Nevertheless, typical scanned objects present noise which needs to be dealt with. This surface roughness, that results from the discretization onto a volume of a noisy point cloud, would decrease the visibility of the feature cells. As a result, the algorithm would not be able to detect and extract sharp features.

In order to address this problem, the algorithm that computes cell-to-cell visibility is slightly modified. Normally, the algorithm would follow the segment that connects the centers of two cells, checking for intersections with the erosion of the occluders. Instead of stopping as soon as an intersection is found, the traversal is allowed to continue through occupied space for a certain length. If and only if the pieces of the segment inside the eroded occluders have a total length of less than $\delta$, the two cells are considered to be visible. This $\delta$ distance, which we call the *opacity* factor, cannot be too large, as it would cause the fusion of different components. Through testing, we have found that an opacity factor of three times the size of a cell's edge produces very good results.

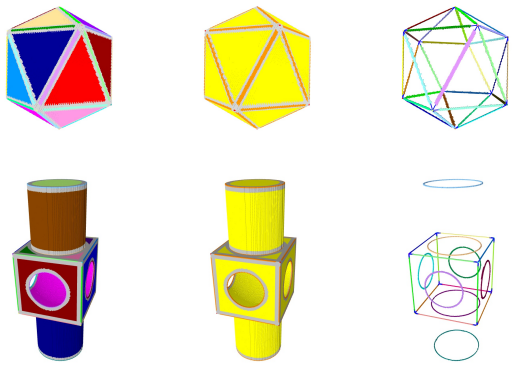

**Figure 7:** *Components identified in the icosahedron and the mechanical part models. First column shows color-coded components, while the second one shows them being classified into vertices, edges and faces. Vertices are in red, edges in orange and faces in yellow. Finally, the last column presents a connectivity graph where vertex components are represented as spheres.*

This technique results in a reduction of the noise influence on the visibility map, but a component may still be divided into some disconnected components. As feature components are separated by large enough bands of non-feature cells, a solution is to join close enough components.

Applying the combination of these two simple techniques to the visibility-based feature extraction method, significantly reduces the impact of noise on it.

## 7 Results and Discussion

Several examples are presented and discussed in this section. Figures 3, 7 and 9 show these models. All the objects used have been rotated randomly, resulting in voxelizations that show the performance of the visibility and edge extraction algorithms in a sufficiently general case. In this way faces are not axis-aligned.

In Figures 3 and 7, the visibility-based edge extraction approach shows that it can correctly extract the features of both very simple objects and also of others of moderate complexity. Curved edges and faces are also obtained, as well as the connectivity of the features of the whole model. More complex objects are presented in Figure 9. The features of the FANDISK model are recovered correctly with two exceptions which may be seen in the final connectivity graph. Due to the fact that the corresponding features do not fully separate their adjacent faces, they are integrated into a larger face component. The CROSS and COVER models also show the good results yielded by our method, though an issue is noticeable in the smaller holes of the COVER model (see highlight rectangle in Figure 9 and the enlargement in Figure 10). The face components in those holes do not cover the entire cylinder, but this is a result of the cylinder being voxelized at a relatively small resolution compared to its curvature. Larger resolutions contribute to facilitate the visibility-based edge identification (see Figure 8).

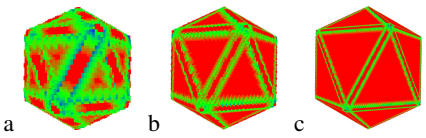


**Figure 8:** *Components become easier to identify using larger resolutions when computing their visibility maps. The images show the visibility map of an icosahedron discretized at different resolutions: (a)* $64 \times 64 \times 64$ *(b)* $128 \times 128 \times 128$ *(c)* $256 \times 256 \times 256$

Figure 11 shows the improved feature extraction method applied to a model corrupted with white gaussian noise of varying standard deviations. The first row displays the results when using a standard deviation of $\sigma = 0.22$, which moves 32% of the samples by more than 0.22 times the cell's edge length. The algorithm recovers straight and curved edges, the noise not being visible on the connectivity graph. Increasing the noise to a standard deviation of $\sigma = 0.32$ (second row) causes only one of the vertices to be merged with an adjacent edge. If we step it up to $\sigma = 0.45$, many components merge incorrectly, but most features continue to be present.

It is also worth mentioning that the introduction of the voxel grid as a helper object, in order to ease the visibility computation, introduces some anisotropy to the algorithm. Despite this fact, the method behaves well for general rotations of the model as shown in Figure 12.

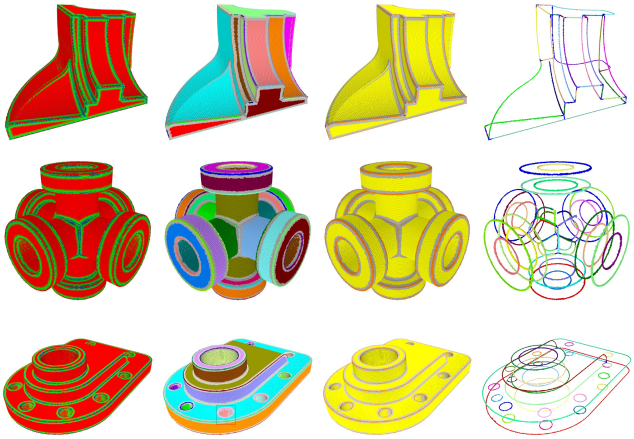

**Figure 9:** *From left to right: visibility maps, identified components, their classification and the resulting connectivity graphs. From top to bottom: the models of the FANDISK, a CROSS pipe and another mechanical part (COVER).*

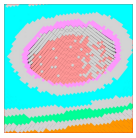

**Figure 10:** *Enlargement of region of the COVER mechanical part within the rectangle in Figure 9.*

Table 1 presents the size of the models and the running times of the four phases of our algorithm. We list the resolution and the number of mixed cells, because the running time of the algorithm ultimately depends on the latter. The running times were obtained on a Pentium 4 at 3.4 GHz with 1 GB of RAM. Most of the time

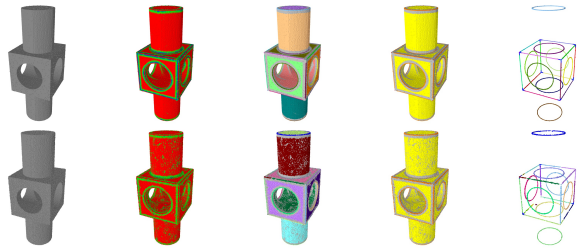

**Figure 11:** *From left to right: binary grids, visibility maps, identified components, their classification and the resulting connectivity graphs. Top row shows the binary grid of a mechanical part corrupted with white gaussian noise of* $\sigma = 0.22$*, while bottom row shows the same model corrupted using a gaussian noise of* $\sigma = 0.45$*, respectively. The length unit is taken to be the side of a cell.*

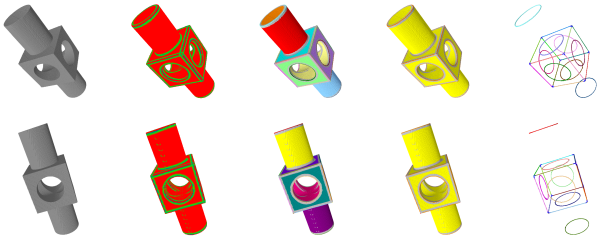

**Figure 12:** *From left to right: binary voxelizations of a mechanical part, the corresponding visibility maps, identified components, their classification and the resulting connectivity graphs. Rotating the model with respect to the voxelization does not adversely affect the results.*

is consumed on the first phase which computes the visibility map. In fact this phase's running times are expressed in seconds, while those of the other three are in milliseconds. The times for the identification and classification of the components are not very significant. Even the final step that extracts the connectivity graph is much faster than the visibility map computation. Although the weight of the visibility part is very important, the whole algorithm is significantly faster than equivalent methods [Jenke et al. 2006], while it recognizes clean features.

Notice that the results are very good despite using only a conservative scheme for the visibility computation. However, if the parts being reconstructed had thin sheets, these could be completely obliterated by the erosion used in that phase. The only solution is to choose a discretization scale well below the size of features that we want to recover.

## 8 Conclusions

The visibility-based feature extraction approach we have proposed works on a binary voxelization which may be obtained from a number of sources, like point samples. We have shown that this method

| | Resolution | Mixed cells | Visibility(seconds) | Components(ms) | Classification(ms) | Graph(ms) |
|---|---|---|---|---|---|---|
| ICOSA | $256^3$ | 220786 | 134 | 327 | 93 | 1960 |
| PART | $512^3$ | 400600 | 254 | 285 | 125 | 1131 |
| FANDISK | $512^3$ | 497513 | 253 | 434 | 193 | 5967 |
| CROSS | $512^3$ | 615218 | 616 | 500 | 181 | 6365 |
| COVER | $768^3$ | 970741 | 567 | 859 | 280 | 1024 |

**Table 1:** *Resolution, number of mixed cells and running times of the four steps of the algorithm for the models shown in Figures 7 and 9.*

is able to identify, classify and connect planar and curved components of mechanical models. We are currently working on using these results to extract the shape of the corresponding isosurfaces, and recovering their sharp features, using our previous work [Chica et al. 2007].

The technique presented here may be used for model acquisition, reverse engineering and signature computation for shape matching applications.

Further enhancements are envisioned including reducing its execution time by performing the visibility computations in an adaptive multiresolution way, so that computations are carried away with a reasonably low-resolution voxelization except near complicated high-frequency details where oversampling is used.

Thin parts may require very high resolutions to be captured. One possible solution would be to adjust the visibility computation algorithm, so that it works with an adaptive space partitioning structure like an octree.

## Acknowledgements

The author would like to thank Pere Brunet, Alvar Vinacua and Isabel Navazo for their advice and valuable discussions on visibility algorithms.

Chica's Research has been supported by the CICYT Spanish Agency under Grant TIN-2004-08065-C02-01 and by a Graduate Research Fellowship from the Spanish Government.

## References

ALEXA, M., BEHR, J., COHEN-OR, D., FLEISHMAN, S., LEVIN, D., AND SILVA, C. T. 2003. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics 9*, 1, 3–15.

AMENTA, N., BERN, M., AND KAMVYSSELIS, M. 1998. A new Voronoi-based surface reconstruction algorithm. *Computer Graphics 32*, Annual Conference Series, 415–421.

AMENTA, N., CHOI, S., AND KOLLURI, R. K. 2001. The power crust, unions of balls, and the medial axis transform. *Computational Geometry 19*, 2-3, 127–153.

ATTENE, M., FALCIDIENO, B., ROSSIGNAC, J., AND SPAGNUOLO, M. 2004. Edge-sharpener: Recovering sharp features in triangulations of non-adaptively re-meshed surfaces. *Proc. of EG/ACM SIGGRAPH Symposium on Geometry Processing 2004*, 62–69.

CARR, J. C., BEATSON, R. K., CHERRIE, J. B., MITCHELL, T. J., FRIGHT, W. R., MCCALLUM, B. C., AND EVANS, T. R. 2001. Reconstruction and representation of 3D objects with radial basis functions. In *SIGGRAPH 2001, Computer Graphics Proceedings*, ACM Press / ACM SIGGRAPH, E. Fiume, Ed., 67–76.

CHICA, A., WILLIAMS, J., ANDUJAR, C., BRUNET, P., NAVAZO, I., ROSSIGNAC, J., AND VINACUA, A. 2007. Pressing: Smooth isosurfaces with flats from binary grids. *Computer Graphics Forum*.

COHEN-OR, D., CHRYSANTHOU, Y. L., SILVA, C. T., AND DURAND, F. 2003. A survey of visibility for walkthrough applications. *IEEE Transactions on Visualization and Computer Graphics 09*, 3, 412–431.

DÉCORET, X., DEBUNNE, G., AND SILLION, F. 2003. Erosion based visibility preprocessing. In *Proceedings of the Eurographics Symposium on Rendering*, Eurographics, 281 – 288.

DEMARSIN, K., VANDERSTRAETEN, D., VOLODINE, T., AND ROOSE, D. 2007. Detection of closed sharp edges in point clouds using normal estimation and graph theory. *Comput. Aided Des. 39*, 4, 276–283.

DINH, H. Q., TURK, G., AND SLABAUGH, G. 2001. Reconstructing surfaces using anisotropic basis functions. In *ICCV '01*, 606–613.

DURAND, F., DRETTAKIS, G., AND PUECH, C. 2002. The 3d visibility complex. *ACM Transactions on Graphics 21*, 2 (April).

ESTEVE, J., BRUNET, P., AND VINACUA, A. 2005. Approximation of a variable density cloud of points by shrinking a discrete membrane. *Computer Graphics Forum 24*, 4, 791–807.

FAUGERAS, O. D., AND HERBERT, M. 1986. The representation, recognition, and locating of 3d objects. *The international journal of robotics research 5*, 3, 27–49.

GATZKE, T., AND GRIMM, C. 2006. Feature detection using curvature maps and the min-cut/max-flow algorithm. In *Geometric Modeling and Processing 2006*, 578–584.

GELFAND, N., AND GUIBAS, L. J. 2004. Shape segmentation using local slippage analysis. *Eurographics Symposium on Geometry Processing*, 214 – 223.

GUMHOLD, S., WANG, X., AND MCLEOD, R. 2001. Feature extraction from point clouds. In *Proc. 10th Int. Meshing Roundtable*.

HO, C.-C., WU, F.-C., CHEN, B.-Y., CHUANG, Y.-Y., AND OUHYOUNG, M. 2005. Cubical marching squares: Adaptive feature preserving surface extraction from volume data. *Computer Graphics Forum (Eurographics 2005) 24*, 3, 537–546.

HOPPE, H., DEROSE, T., DUCHAMP, T., MCDONALD, J., AND STUETZLE, W. 1992. Surface reconstruction from unorganized points. *Computer Graphics 26*, 2, 71–78.

HORNUNG, A., AND KOBBELT, L. 2006. Robust reconstruction of watertight 3d models from non-uniformly sampled point clouds without normal information. In *Proceedings of Eurographics Symposium on Geometry Processing 2006*, 41–50.

HUANG, R., MA, K.-L., MCCORMICK, P., AND WARD, W. 2003. Visualizing industrial ct volume data for nondestructive testing applications. In *Proceedings of IEEE Visualization 2003 Conference*, IEEE.

JENKE, P., WAND, M., BOKELOH, M., SCHILLING, A., AND STRASSER, W. 2006. Bayesian point cloud reconstruction. In *Proc. Eurographics 2006*, 853–857.

JU, T., LOSASSO, F., SCHAEFER, S., AND WARREN, J. 2002. Dual contouring of hermite data. *ACM Transactions on Graphics 21*, 3, 339–346.

KOBBELT, L. P., BOTSCH, M., SCHWANECKE, U., AND SEIDEL, H.-P. 2001. Feature-sensitive surface extraction from volume data. In *SIGGRAPH 2001, Computer Graphics Proceedings*, ACM Press / ACM SIGGRAPH, E. Fiume, Ed., 57–66.

LORENSEN, W. E., AND CLINE, H. E. 1987. Marching cubes: A high resolution 3d surface construction algorithm. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, vol. 21, 163–169.

MEDEROS, B., AMENTA, N., VELHO, L., AND FIGUEIREDO, L. H. 2005. Surface reconstruction from noisy point clouds. In *Symposium of Geometry Processing '05*, 53–62.

NIRENSTEIN, S., BLAKE, E., AND GAIN, J. 2002. Exact from-region visibility culling. In *In Proc. of the 13th Workshop on Rendering, 2002.*, 191–202.

PAULY, M., KEISER, R., AND GROSS, M. 2003. Multi-scale feature extraction on point-sampled surfaces. In *Eurographics 2003*, 281–289.

TURK, G., AND O'BRIEN, J. F. 1999. Shape transformation using variational implicit functions. *Computer Graphics 33*, Annual Conference Series, 335–342.

WONKA, P., WIMMER, M., AND SCHMALSTIEG, D. 2000. Visibility preprocessing with occluder fusion for urban walkthroughs. In *Rendering Techniques 2000 (Proceedings of the Eurographics Workshop on Rendering 2000)*, Springer-Verlag Wien New York, B. Proche and H. Rushmeier, Eds., Eurographics, 71–82.