Software for a X Ray Tomograph

Rosario Martínez Gómez, Israel Vite Silva, and Luis G. de la Fraga

Cinvestav, Department of Computing.
Av. Instituto Politécnico Nacional 2508. 07360 México D.F., México
Phone (+52) 55 50613755, E-mail: fraga@cs.cinvestav.mx

Abstract A software tool-box has been developed, which allows, not only the volume reconstruction from X ray tomography images, but also the visualization of reconstructed isosurfaces. Although it may be considered that the software needed is tightly linked to the hardware, we'll prove that we need five different (hardware independent) software components for the whole process: (1) acquisition, (2) projection's centering, (3) reconstruction, (4) isosurfaces segmentation and (5) visualization. The reconstruction component was taken from Xmipp, an old software for 3D reconstruction of biological macromolecules, segmentation was developed using k-means algorithm and the visualization was built using the splatting technique. In addition we compare splatting with another two surface visualization techniques such as simple voxels and deformable simplex meshes. The most of the software components were developed in C, C++ and perl. The graphical user interface for the visualization component was developed in C++ using Qt and OpenGL libraries.

Keywords: 3D visualization, splatting algorithm, tomography.

1 Introduction

The objective of a three-dimensional reconstruction is to obtain information about nature and structure of materials that conform the inside of an object. The are several applications, for example data from electron microscopes are used to reconstruct the molecular structure of proteins or to reconstruct the x-ray structure of an astronomical object such as supernova remnant, however one of the most important applications has been in medicine to obtain the density distribution within the human body from multiple x-ray projections [1]. This process is referred to as computerized tomography and is a widely used technique, which enables the reconstruction from projections and the visualization of the internal structure of objects.

A tomograph generates images, named projections, of cross sections of the scanned object, from data obtained by measuring the attenuation of x rays along a large number of lines throughout specimen under study. Despite the fact that acquisition part, in a process of computerized tomography, is tightly linked to the hardware, another components of software are needed in order to solve the

whole problem of visualizing reconstructions. Thus, we have analyzed five main parts of software:

1) projection's acquisition, 2) projection's centering, 3) specimen's reconstruction, 4) segment the different densities of the reconstructed object, and the last, 5) visualization of the surface for each segmented density. Every part is independent of the others and consists of one or several programs, but all parts must be used to resolve the whole problem of visualizing a tomographic reconstruction.

This paper is organized as follow: in section 2 are described part (1) to (4) of the developed software. In section 3 is described the visualization part developed with splatting technique and it is compared with others two visualization techniques, one with voxels faces, and the other one with deformable simplex meshes. Finally the conclusions are presented.

2 Software Parts

2.1 Acquisition

In this part we acquire projections of the specimen under study. In Fig. 1 is shown a photograph of the tomograph for which we developed the software described in this work. The X ray source generates radiation that goes throughout the specimen. X rays are invisible to human eye, therefore to detect them, their photons must hit a phosphorescent screen, by this way the intensity of the light viewed on the screen is proportional to the density and composition of the scanned object. In order to obtain several specimen's projection views, the revolving plate rotates the specimen around a fixed axis. The projections are collected with a CCD camera, it uses an optical device to get the views on the phosphorescent screen. The process geometry depends on a single rotation angle, it is known as *single axis geometry* [1,2].

All tomograph's devices are controlled by an old PC with ISA slots and Windows 98 operating system. The card's drivers to transfer the CCD's images to the PC, and to shut the X ray beam, only work in that operating system. There is not enough information about the PC's cards, therefore we were unable to migrate this program to the GNU/Linux operating system.

2.2 Reconstruction

We built a synthetic volume of mathematically described objects (it is called phantom), placed at desired positions, at desired orientations and of desired size and density, in order to separate possible problems which can not be separated physically such as acquisition's noise. In addition we tested two reconstruction methods: weighted backprojection and ART (Algebraic Reconstruction Technique), both were adapted from Xmipp collection [3], which is an old free-software for reconstructing biological macromolecules. Since backprojection's algorithm calculates the inverse process for generating a projection the result is

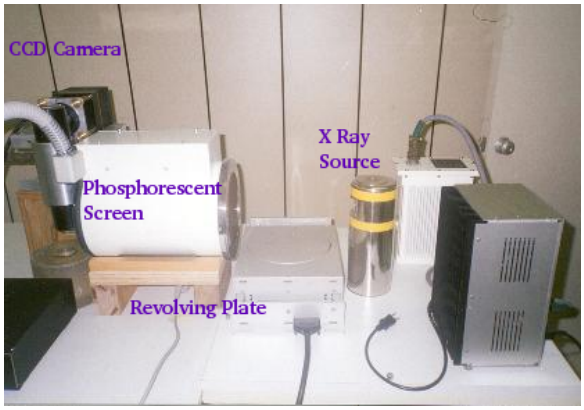


Figure 1: A tomograph's photograph.

blurred, because the most of information is concentrated in the low frequencies. Then a filter must be applied to correct this problem. We implemented the Wiener and Ramp filters and these are applied to each projection in the reciprocal space. Ramp-filter is typically used in weighted backprojection hence it produces more weight to high frequencies. On the other hand, Wiener-filter has the aim of filtering noise which it is tightly linked to the acquisition process.

Ramp-filter is very simple, in two dimensions it is specified as in Eq. 1. Where r is equal to the circle's radius i.e. $\sqrt{x^2+y^2}$, and k is a constant proportional to the Nyquist frequency [4].

Although the reconstruction weighted by Ramp-filter is acceptable, it become worse when noise is present since this filter amplifies high frequencies. In order to solve it Wiener-filter could be used.

$$q(r) = \begin{cases} |r|, & \text{if } r \leq k \\ 0, & \text{if } r > k \end{cases} \tag{1}$$

To design the Wiener-filter, we require knowledge about noise and object statistics. Eq. 2 express the Wiener-filter, where $f = \sqrt{(x^2+y^2)}$, and typical used values are $\alpha = 0.5$ and $SNR = 60$; the noise commonly is modeled as white additive noise [5].

$$A(f,\theta) = \frac{|f|2\pi\alpha(SNR)}{2\pi\alpha(SNR) + |f|(\alpha^2 + 4\pi^2 f^2)^{\frac{3}{2}}} \tag{2}$$

To test the developed programs we generated seventy two phantom's projections (taken every 5 degrees on the tilt angle), then we performed three reconstructions using the reconstruction algorithms: backprojection, weighted backprojection, and ART. To compare the reconstructions, we drew the changes in the density along the 15th row on the 30th slice as is shown in Fig. ref-fig:eficiencia. The simple line represents the values of the original phantom's densities, the backprojection-reconstruction is represented by the square-line, it

is improved when Ramp-filter is used (asterisk-line), however this filter decrease the dynamic amplitude, thus the best reconstruction is developed by ART, which it is represented by the cross-line.

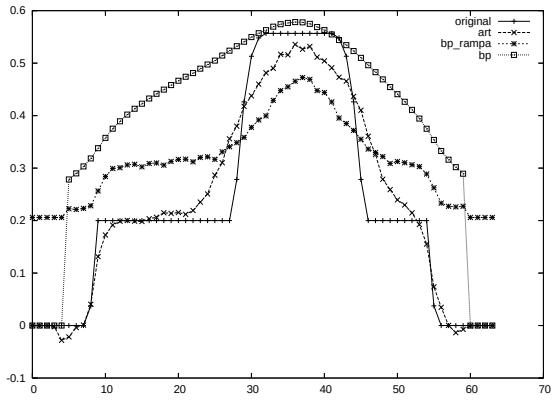


Figure 2: Graph of voxels' density values along the 15th row and 30th slice of the reconstructed volumes. The best reconstruction is performed by ART, which it is represented by the green line.

2.3 Projections centering

Before performing a 3D reconstruction, all projections must be located in a coordinate system fixed to the object, because if we take projections without being aligned, we will reconstruct a distorted volume [1,2]. It is possible to set experimentally the coordinate system origin, first we need to put the specimen on the revolving plate, after rotating the specimen we are able to verify if the still-specimen is in the plate's center. This is a boring trial-error process, which could be solved by software, then we going to show how we solved this problem.

The centering problem can be divided in two cases: (a) when object is small, and (b) when object is big. Fig. 3 shows the (a) case, a noncentered small object, the object's center is a l distance from the revolving plate center.

To solve the second case, noncentered projections of a small object, we can assumed that the coordinate system origin was in the object's center of mass, hence we could estimate it with great accuracy from the projection's centroid because these are free of noise.

The projection's center is a good reference when we can calculate it, that is, when each projection shows the whole object, however we can not calculate the centroid when the object is bigger and their projections do not show the whole object. Fig.4 is a diagram of this case: a big object is not in the revolving plate's center and the object's center is at a l distance of the plate's center. As we know

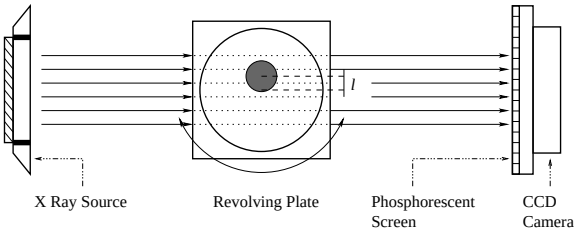


Figure 3: The object is not in the plate's center. There is a distance l from the plate's center to the specimen's coordinates origin.

the tilt angle for each projection, the displaced distance for each projections is $l \cos \phi$, where ϕ is the projection's tilt angle.

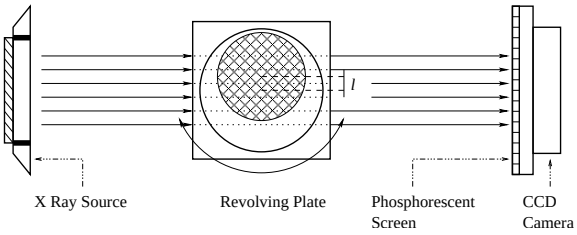


Figure 4: A projection does not show the complete object's information when the object is bigger that the width of the X ray beam.

To center the big object projections we assume not only that their projections are taken with a small increment in the tilt angle but also the displacement is only on the x axis, therefore the projection taken at 90 degrees must have a displacement of value zero, and that projection can be used as a reference to centering the others projections. Besides, we calculate, using cross-correlation, the relative displacement between consecutive projections (so they have a small displacement in the tilt angle). The increment in the tilt angle must be less than 5 degrees to get an error in the calculation of the displacement less than 1 pixel.

To automatically to know which is the reference projection from the displacement data, we need to apply the Eq. (3). In Eq. (3) x is the relative displacement between two projections, j is a index to the reference projection, d is the absolute displacement for the k projection.

$$d_k = \begin{cases} -\sum_{i=k}^{j-1} x_i & \text{if } k < j, \\ 0 & \text{if } k = j, \\ \sum_{i=j+1}^{k} x_i & \text{if } k > j, \end{cases} \tag{3}$$

In order to apply Eq. (3), we suppose that we have all projections in the same order as were acquired and we have calculated the x values, the displacement

between a projections with respect to the previous projection. The reference projection is the corresponding to the minimum value of function s in Eq. (4), that represent the sum of all absolute displacements for n projections and d_k calculated as Eq. (3).

$$s_i = \sum_{k=0}^{n-1} d_k \tag{4}$$

2.4 Segmentation

The reconstruction stage produces a three-dimensional matrix with a density value for each of its elements. To separate the different volumes with the same densities, we decide using the k-means algorithm as a clustering kernel, since it classifies n objects into k disjointed groups, based in some object's features. Thus, we use the cluster-minimal distance between a density value and a group's center. Note that we need only separate the different densities, or to separate the objects composed by the different densities; to separate the objects by their structural components it is necessary to use a clustering algorithm that takes into account the spatial coordinate (x, y, z) for each voxel.

The k-means algorithm perform the following steps:

- Step 1: Initialize the group's centers
- Step 2: Each element is assigned to a group according to the minimal distance between a group and the element.
- Step 3: For each group the center is recalculated with the new assigned elements.
- Step 4: While there are changes in the group's elements, go the step 2.

In the developed program, the number of clusters and the groups centers must be initialized by hand from the density-histogram information. The number of clusters is equal to the number of peaks. Each group's center is defined by two threshold values around the histogram peaks. The segmentation program produces binary volumes as number of groups were specified.

3 Surface Visualization Using Splatting

Splatting is a novel technique for rendering a volume based in points, it is used to render surfaces or volumes in its surface representation, that is, a 3D object is represented as a collection of samples, which they are lying over its surface [6,7]. Each sample point has position, normal, color and some others parameters, which allow to know the distance among sample points and their neighbors. The projection of a point (and its contribution around) is known as the rendering primitive *splat*. In order to represent the splats we used triangles, squares, and circles (actually a circle is a polygon of twelve sides). Each one of them seems

a small planar patch, which are assigned to a point oriented along the object's surface.

The aim of our software component is to visualize the surface of the reconstructed volumes using the splatting technique. So as to perform this part the following steps were made:

- Step 1: Extract the surface for each segmented volume using mathematical morphology. The center of each voxel will be the coordinates of the point and this point will be used to represent the surface.
- Step 2: Calculate the normal to each point over the surface.
- Step 3: Associate a splat to each surface point and set its position according with its normal.

By subtracting an eroded volume (it built with a structural element which consists of a voxel and its six neighbors), to the original volume we performed step 1, the extraction of the voxels in the binary volume surface; so that we developed an program which receives a binary volume and produces another binary volume only with the surface voxels set to 1.

The step 2 is solved in very simple way: A plane is fitted to a point and its neighborhood, then the point's normal is obtained. The plane's equation is given by $Ax + By + Cz + D = 0$, in addition if we divide this equation by D, we will obtain $A'x + B'y + C'z = -1$, that is, we only need three points for obtaining the values of the three unknowns, however we could have 27 points in the neighborhood; therefore we decide to use the SVD method (Singular Value Decomposition) to solve the overdetermined system, moreover by using SVD we assure that we have gotten the best plane fitted in the least squares sense. Once we calculated the plane we know that the point's normal is (A, B, C). Now we need the correct normal direction, so we checked the values of the two neighbors voxels in the normal's position of the original volume; the correct direction is where the neighbor voxel has the value of 0 (outside the volume no density exists).

Finally, to perform the step 3, a splat is represented using OpenGL-primitives; for that reason we designed and programed an graphical user interface (GUI) made in Qt [8] and OpenGL [9] to visualize the volumes. Qt is a library to rapid prototyping developed in C++, it has an excellent documentation, and it allows to assign an OpenGL-widget to the GUI. As input to the GUI we have a file with the format: a number in a single row that represents the maximum distance between two voxels, 6 numbers per row of the three coordinates values to every point and three of their normal's values. Coordinates-values must be normalized between -0.5 and 0.5. The splat-rendering primitive was performed when the visualization is made, that is, a splat is assigned to every point, located according to its normal, and with a size of the maximum distance of two voxels in order to avoid holes in the visualization. Furthermore, we represented splats as triangles, squares, and circles, besides it allows the visualization of points as a quick visualization help.

3.1 Results

The visualization of the trial-phantom and its ART-reconstruction are presented respectively in Fig. 5 and Fig.6. The first row shows the visualization with points; we can watch splat's representations with triangles, squares, and circles corresponding to the second, third, and fourth row. In spite of rendering with triangles should be the cheapest, in the sense of computational resources, we weren't able to notice that, since we tested a relatively small phantom; in other words, we didn't felt a loss of performance with any splat primitive. However, the best splat is represented with circles, because we can see holes on images on second and third row, which they correspond to the triangle and square splat implementations.

The reconstruction's visualization is good as we can see in Fig. 6, also the phantom's size was lightly increased since we performed a morphological closing operation in order cover holes in the reconstructed surface.

We also made a comparison of the phantom's visualization between splatting, simple voxels (using Scubes program [10]) and a deformed simplex mesh [11]. As we can see in Fig. 7 the visualization with simple voxels is not good because the normals are calculated to every voxel face. The visualization with the deformable simplex mesh is not good, because it is unable to represent an object with genus grater than 0 (with one o more holes), in addition is not a simple task to assign initial values to the mesh and the deformation's program is not easy to use. The visualization with splats not only is easy to use and fast to calculate but also produces the best quality; however it is necessary for splatting that the surface is covered with a enough number of points to avoid holes in the visualization.

4 Conclusions

We developed the software for working with a x-ray tomograph. We proposed a software made of five components: acquisition, projection's centering, 3D reconstruction, isosurface's segmentation and visualization. We designed and built four of the five parts. The 3D reconstruction component was adapted from Xmipp, we used weighted backprojection and ART algorithms of this collection. In our test, ART algorithm is better than weighted backprojection: the former produces not only a better quality but also a better dynamic amplitude. We solved by software the problem of projection's centering.

For the segmentation part, we use the k-means algorithm so as to cluster the density values of the voxels around k densities. The number of densities and the initial values for the groups centers are given by hand from the 3D-reconstruction's densities-histogram.

To visualize the isosurfaces on the reconstructed volumes we designed a simple algorithm to calculate the surface's normals: a plane is fitted using SVD to the neighborhood to each surface voxel. We used the voxel's center as the point that sampling the surface. In addition, we used OpenGL primitives to represent

a splat, we tested the primitives of triangles, squares, and circles. The best visualization in provided by the circles (actually a circle is a polygon of twelve sides). We show that splatting is easy to use give us the best quality visualization.

We think that all the programs developed must be tunned according to a specific application, therefore they must be changed to adjust the visualization to an specific reconstruction application task.

References

1. Gabor T. Herman. *Image Reconstruction From Projections*. ACADEMIC PRESS,INC, Orlando, Florida 32887, 1980.
2. John C. Russ. *The Image Processing Handbook*, chapter 9, 10. CRC PRESS & IEEE PRESS, 3. edition, 1999.
3. R. Marabini, I.M. Masegosa, M. C. San Martin, S. Marco, J.J. Fernandez, L.G. de la Fraga, C. Vaquerizo, and J. M. Carazo. Xmipp: An image processing package for electron microscopy. *Journal of Structural Biology*, pages 237–240, 1996.
4. Edwin L. Dove. Notes on computerized tomography. In *Bioimaging Fundamentals*. Dove Â Physics of Medical Imaging, 2001.
5. Image projections and the radon transform. http://www.owlnet.rice.edu/ elec539/Projects97/cult/node4.html.
6. Matthias Zwicker, Hanspeter Pfister, Jeroen van Baar, and Markus Gross. Surface splatting. In Eugene Fiume, editor, *SIGGRAPH 2001, Computer Graphics Proceedings*, pages 371–378. ACM Press / ACM SIGGRAPH, 2001.
7. Christopher S. Co, Bernd Hamann, and Kenneth I. Joy. Iso-splatting: A point-based alternative to isosurface visualization. In *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*. Computer Society, 2003.
8. Qt. http://doc.trolltech.com/3.0/index.html.
9. Opengl. http://www.opengl.org/.
10. Luis Gerardo de la Fraga and Feliú Sagols Troncoso. Scubes: A program to visualize vox-solids. In *VII Conferencia de Ingeniería Eléctrica*. CINVESTAV, 2001.
11. Jorge Eduardo Ramírez Flores and Luis Gerardo de la Fraga. Basic three-dimensional objects constructed with simplex meshes. In *Electrical and Electronics Engineering, 2004. (ICEEE). 1st International Conference*. CINVESTAV, 2004.

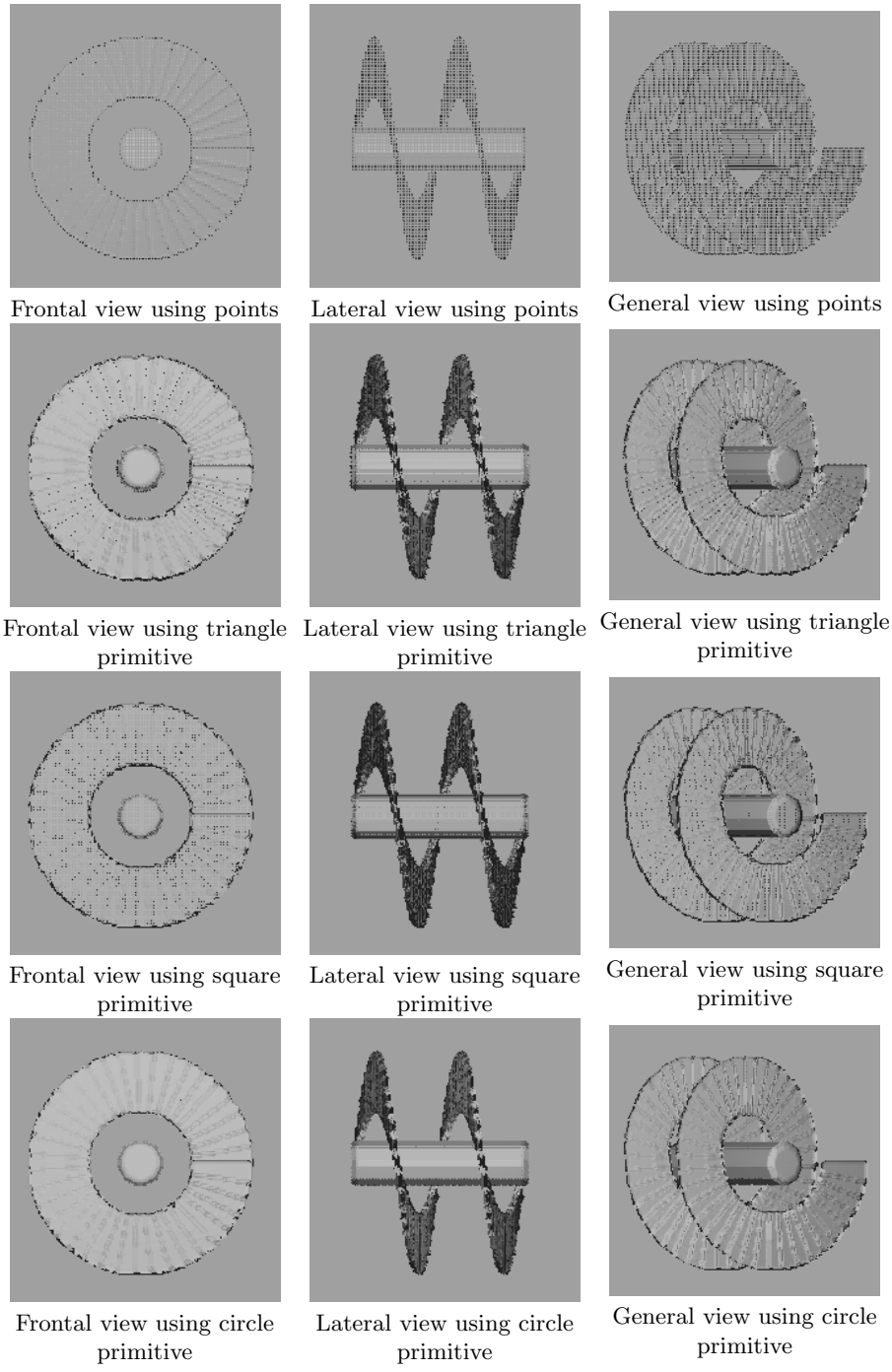
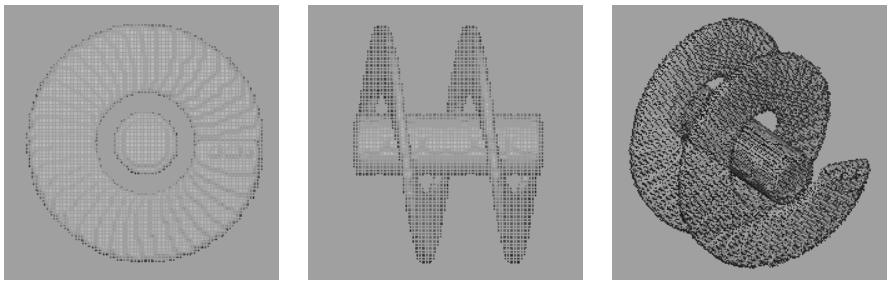
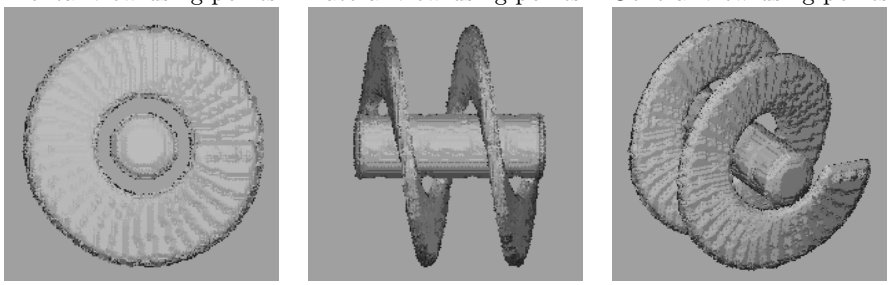


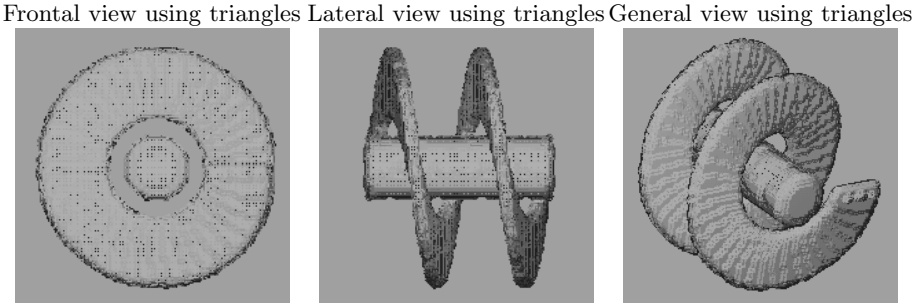
Figure 5: Comparison of the visualization with points and three primitives, triangle, square and circle, to render the splats.



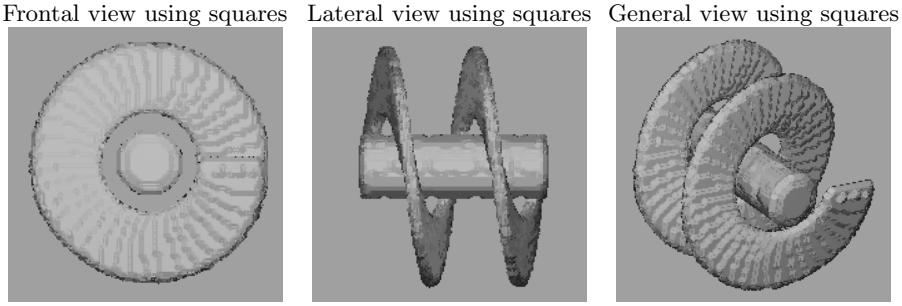
Frontal view using points | Lateral view using points | General view using points



Frontal view using triangles | Lateral view using triangles | General view using triangles

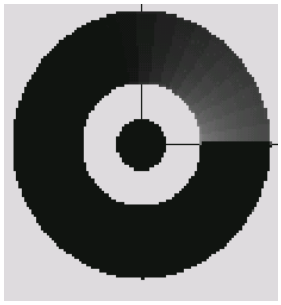


Frontal view using squares | Lateral view using squares | General view using squares

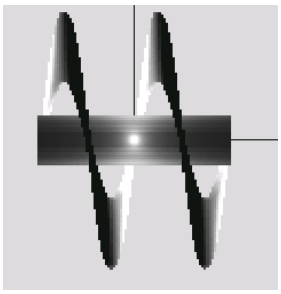


Frontal view using circles | Lateral view using circles | General view using circles

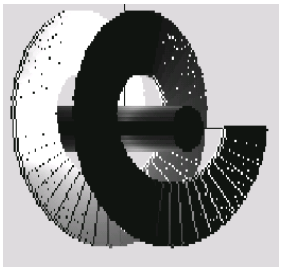
Figure 6: Visualization using different splats of the reconstructed phantom.



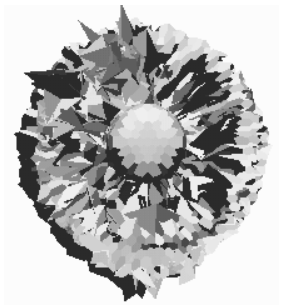
Frontal view using simple voxels



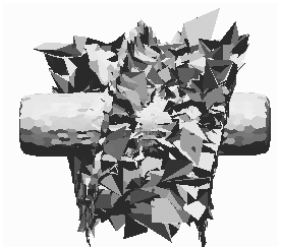
Lateral view using simple voxels



General view using simple voxels



Frontal view using a deformable simplex mesh



Lateral view using a deformable simplex mesh

General view using a deformable simplex mesh

Figure 7: Comparison of the visualization with other two techniques: using simple voxels and a simplex mesh deformed to cover the phantom surface.