# Data Reduction Algorithm Based on Gaussian Image Estimation

E.A. Zanaty
College of Computer Science
Taif University, Saudi Arabia
zanaty22@yahoo.com

Sultan Aljahdali
College of Computer Science
Taif University, Saudi Arabia
aljahdali@tu.edu.sa

Narayan Debnath
Computer Science Department
Winona State University
ndebnath@winona.edu

***Abstract:*** *A novel 3D shape preserving data reduction technique for managing the amount of data acquired by laser scanning is presented that overcomes the shortcomings of existing filter-based methods. The technique is based on a discrete Gaussian image of the scanned points which is obtained by estimating surface normals and projecting them into a Gaussian sphere. The discrete Gaussian image is then used to partition the points into cells. In each cell, a reference point and its neighbours are used to determine the cell representative point and all other points are removed. The performance of the proposed method is illustrated using a range of point clouds scanned from typical engineering surfaces.*

*Keywords: shape preserving, filtering, data reduction, reverse engineering.*

## 1.Introduction

Reverse engineering refers to the process of creating engineering design data from existing physical parts, by acquiring its surface data using a scanning or measurement device. Improved data acquisition methods, especially using laser scanning, now make it possible to process scattered point clouds in three dimensional space with high accuracy. It has now become a realistic expectation to generate exact and continuous models, which can be directly transferred to and utilised by CAD/CAM systems. The process of reverse engineering can be divided into four phases: data acquisition, data preprocessing, segmentation and surface fitting [1].

Much research effort has been directed at managing the amount of data points acquired by laser scanners [2-8] including Fourier transform (DFT) [2] and the discrete Wavelet Transform (DWT) [3]. These techniques are designed to reduce the redundant points in 1D and 2D data sets and are typically used to speed up image transmission. In both DFT and DWT methods, reduction is achieved by filtering the high frequency component of data whilst maintaining the underlying structure of the image. Similar method has been stated in [5] for text classifiers. The techniques presented in this category do not exist in three dimensions and are practically difficult to be manipulated with very dense 3D data.

The simplest filter method of the second category is neighbourhood averaging [1]. For each point, its neighbourhood, in terms of Euclidean distance, is identified. All points within the neighbourhood are then simply replaced by the average value. Clearly, the method is simple and relatively quick but has little shape preserving properties since it will tend to smooth any shape variation. This is especially true of edge points. The median filter [4] is less affected by extreme data values and is therefore better at preserving edge points. It places a grid structure such that all cells contain the same number of points. The median value of all points within a cell is determined and all points within a cell are replaced by this single value. The approach is simple and relatively quick but, due to the grid being uniform, regions of the point cloud can be underrepresented and hence the method is insensitive to shape. The method is ideal for data scanned from planar structures and for correcting inaccuracies in a single axis direction but it is less successful for non-planar data. When the x and y coordinates are determined more accurately than z values, the method is faithful but can fail otherwise. Furthermore, to find the median point, a grid is fitted to a suitable analytic surface, i.e. planar, cylindrical, and spherical, etc., where the distances between a surface and the points are estimated and a point with median distance is selected as median point. Clearly, this introduces inaccuracies into the method.

To avoid the limitation of the median filtering methods, partitioning of the points into cells using geometric constraints has been proposed [5]. Points in a cell are used to construct a plane of best fit. If the average distance of the points in a cell to its plane of best fit is greater than some user defined threshold, the cell is subdivided. This process continues until all cells are within the threshold. The data is reduced by selecting the point with median distance. These methods work faithfully and are stable when applied to planar and non-planar data. The accuracy is dependent on the precision and distribution of the scanned data. Regions with sparse data coverage give poor reduction results as do data sets with non-homogenous sampling accuracy.

Filter-based methods that try to constrain the factor ratio of scanned data have been presented for subsampling the data, including randomized sampling, uniform sampling, normal-space sampling, and covariance sampling [6]. Randomized sampling selects points at random and uniform sampling draws equally distributed samples from the input point cloud. Normal space sampling, as proposed by Rusinkiewicz and Levoy, aims at constraining transnational sliding of input meshes, generated from the point cloud. Their algorithm tries to ensure that the normals of the selected points uniformly populate the sphere of directions. The computational expense of the normal and meshes algorithm depends mainly on the number of points. In a brute force implementation, the point pairing is $O(n^2)$.

Recently, Yan et al. [7] presented an overview of the popularly used feature extraction and selection algorithms under a unified framework. Then dimensionality reduction method is improved for large-scale and streaming data classification tasks. It can be used to improve both the efficiency and the effectiveness of classifiers and is designed under the optimization criterion for feature extraction. However, Zanaty [8] presented an efficient method for reducing a dense data. The algorithm starts by estimating the neighbourhood of the points, whereby the surface normal is obtained by fitting the best quadratic to the neighbourhood of the points. After that, the normals vectors are assigned to the points and an initial partitioning of data points into cells is obtained. A procedure is used to reduce the data in each cell. Surprisingly, little work has been done to combine real data reduction and hold only the pre-processed data for further analysis. Such work is of crucial importance since it is extremely difficult to work with dense data in reverse engineering processes for obtaining a full CAD-model. In general, filter-based methods that partition the data into cell, which contain both edge and face points as in [2-8], can distort the representation of the original surface shape especially on boundaries. If a cell contains both edge and face points, the filtered value will be mix of both point types and hence could misrepresent the true shape.

To address the shortcomings of filter-based methods, This paper presents an alternative method that has improved shape preserving properties and incorporates constrained filtering to help reduce data distortion. The proposed method is based on a discrete Gaussian image of the scanned points which is obtained by estimating surface normals and projecting them into a Gaussian sphere. By using the discrete Gaussian image constructed from estimated normals at all points, shape is allowed to influence the data reduction process. The techniques in [2-8] work on 3D digitized data, i.e. the points have been processed according to x, y, and z positions while it is the first time that the Gaussian image is used for data reduction. Clearly, the proposed method is more complex than existing filter-based methods and as a consequence, will require more processing time. However, since normals are of use in follow on activities, this overhead may be deemed acceptable. The proposed method is very accurate with dense and non dense data and ideally suited as a pre-process for segmentation and surface fitting.

The remainder of this paper is organized as follows: In section 2, an overview of the proposed method is given and its advantages over existing filter-based methods are highlight. Section 3 discusses the estimation of the normal vector at each point. The method of partitioning the points is given in section 4. In section 5, a reference direction is determined and used in the method of point reduction, which is detailed in section 6. The performance of the proposed method is illustrated by application to a variety of simulated and scanned data and the results are presented in section 7. The paper is concluded in section 8 by summarising the advantages of the proposed method and assessing its overall performance.

## 2. The Gaussian Image Based Method

The method begins with a set of ***n*** unordered noisy 3D points, typically captured using a laser scanner. It is assumed that the data has been pre-processed to remove gross outliers [9]. Unit normals are estimated for each data point and projected into the Gaussian sphere, resulting in a discrete version of the Gaussian image. This 2D image is then partitioned into a number of cells, based on the Euclidean distances between points, where the dense points can be partitioned based on their coordinates. Each cell is then covered by a grid such that the number of points within each cell is governed by the Gaussian image and hence the local shape of the cell. Finally, the number of points in all cells is reduced in turn. Limitations of existing filter-based methods are overcome by using shape information contained in the estimated normals and a grid structure that reflects both the shape of the data and the intended data reduction ratio.

## 3. Estimating Normal Vectors

Estimation of surface normals is a fundamental task in many reverse engineering algorithms. A number of methods to estimate the normal of a point from discrete information have been proposed. The majority estimates normals from a locally fitted least squares surface. For each point, the *k*-nearest neighbours, $\Omega_k$, computed in terms of Euclidean distance, $\nabla$, are determined. A low order surface is then fitted to the neighbourhood from which the normals are readily computed. A variety of surface forms have been proposed ranging from planar, quadratic, or cubic [9] and parametric quadratic surfaces [10]. However, the most widely used method is based on the Darboux-frame [11] that fits a local explicit surface of the form: $Z(x,y)=ax^2+bxy+cy^2$. The method is stable and easy to implement but has limited geometric shape description. To overcome this, an explicit general quadratic surface has been proposed [12] with reported improved accuracy of estimated surface characteristics. Since the Gaussian image is constructed using the estimated normals, the approach is to use the general implicit quadratic surface, ***S***, represented by: $F(x,y,z)= a_1x^2+a_2y^2+a_3z^2+a_4xy+a_5yz+a_6x\,z+a_7x+a_8y+a_9+a_{10}=0$ to approximate the local surface of $\Omega_k$, . Here, we use the computation of the normal vectors of the last method [12], as this method is found to be powerful in normal vectors computation and works very well for dense data.

## 4. Partitioning of Points

### *4.1 Gaussian image estimation*

Normal vectors, $\vec{n}: i=1,...,n$ are estimated at each data point. The Gauss map is the map from the surface ***M*** to the unit 2-sphere. Each unit normal vector on ***M*** becomes a point on $S^2$ (Fig.1). The Gaussian image of an arbitrary surface is some set on the unit 2-sphere (Figure 2). For example the Gaussian image of a developable surface is a curve (Figure 3). The Gaussian image can now be subdivided into a number of discrete cells using the uniform grid. All planar cells will have the same normal direction resulting in a dense point in the Gaussian image. Thus, whilst the scanned data set may have many faces in the same plane, the algorithm treats them as a single face for the purpose of reduction. Once the plane has been reduced, it is then mapped back onto the individual faces in the original scanned data.

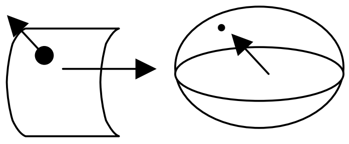


Figure1: Gauss map of a unit normal vector gives a point on $S^2$

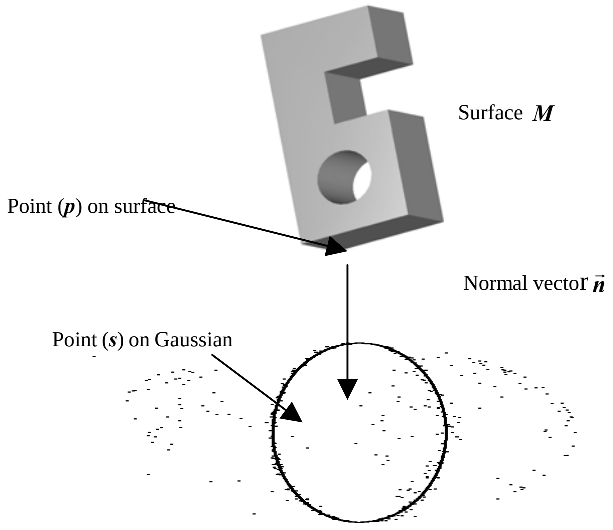


Figure 2. Gauss map of normals give points on $S^2$ .

The Gaussian image can now be subdivided into a number of discrete cells using the uniform grid. All planar cells will have the same normal direction resulting in a dense point in the Gaussian image. Thus whilst the scanned data set may have many faces in the same plane, the algorithm treats them as a single face for the purpose of reduction. Once the plane has been reduced, it is then mapped back onto the individual faces in the original scanned data.

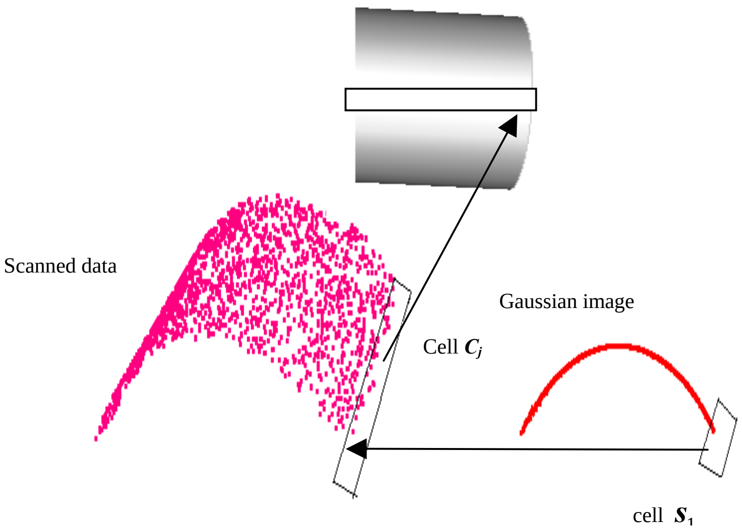


Figure 3. Construction of cells from the Gaussian image.

### *4.2 Partitioning Gauss points $s_i$.*

Let each 3D point $p \in P_i$ be assigned to the corresponding 2D points $s_i \in S$ on a 2-sphere. Further, let $s_m = n^{-1}\sum_{i=1}^{n} s_i$ be the center mass point. The next step is to subdivide the $s_i$ into some cells.

This can easily be done by sorting $s_i$ into an array *D* according to their Euclidean distances, $\nabla_i$, to the point $s_m$. To subdivide the array *D* into cells, the top point $s_j$ in *D* is selected and the Euclidean distance between this point and its $\mu$-neighbor points from the top of the array is computed. $\mu$ is a user prescribed value depending on the size of the given data set, $n$, that can be used to optimize the processing time. The results are not dependent on $\mu$., however larger values result in more processing time. For the data sets considered when $n<20000$ a suitable value is $\mu=200$, for $20000<n<40000$; $\mu=1000$ **and for** $n>40000$, $\mu=10000$. A point is extracted from *D* if it has Euclidean distance smaller than a threshold $\xi$. For instance, let $T_j = \{s_1, s_2, \ldots, s_3, S_{\beta_j}\}$ be a cell of *D* if $|s_j - s_t| \le \xi \, \forall \; t = 1,\ldots,\beta_j$, where $\xi$ is a user defined parameter and $\beta_j$ is the number of points in the *j*-th cell which varies with the parts's shape. The partitioning of Gauss points into cells is outlined in Algorithm 1. If a $T_j$ contains more Gauss points than a user defined limit $\upsilon$, it is further subdivided using Algorithm 2. Clearly the parameter $\upsilon$ determines the maximum number of points in each cell. Experimental evidence suggests using $\upsilon=k$ although the algorithm appears to be stable with regards to choice of $\upsilon$. This process is repeated by selecting the next top point from *D*. When all the Gauss points, $s_i$, have been partitioned into cells $T_j$, the corresponding data points $p_i$, $i=1,\ldots,\beta_j$ are placed into the corresponding cell $C_j$ ( Figure 3).

**<u>Algorithm 1: Partition $s_i$</u>**

```
1-Compute the center mass point, $s_m$
2-Sort the points $s_i$, $i = 1,2,\ldots,n$ in an array $D$ according
   to the Euclidean distance, $\nabla_i$, to $s_m$
3-Select the first/next point in $D$; compute $\nabla_i$'s between
   this point and its $\mu$ neighbours
4-Count the number neighbour points with $\nabla_i \le \xi$.
5-If the counter<$\upsilon$ then apply algorithm 2 to further
   partition the cell
6-Place corresponding data points $p_i$ and corresponding
   normals, $n_i$, into cell $C_j$
7-Remove the extracted points from $D$
8-Repeat steps 3-6 until all the points in $D$ have been
   processed.
```

**<u>Algorithm2: Subdivision of cell $T_j$:</u>**

```
1- Sort the points into a 1D array and subdivide the data into
   two equal sized cells according to their positions in x-
   direction (or y-,z-).
2- The data is subdivided according to their x-positions.
3- If the number of the points in a cell<$\upsilon$ goto step 6 in
   Algorithm 1; else subdivide the points again according to
   their y-positions, repeating for z if necessary.
```

## 5. Reference Direction Selection

At this stage, all 3D data points have been partitioned into cells containing $\beta_j$ points, and corresponding estimated normal vectors, $\vec{n} : i = 1,\ldots,n$. An arbitrary direction is now chosen as a reference direction. Since the reduction process is independent of this direction, a suitable choice is the unit vector along the *x*-axis i.e. $\vec{N} = (1,0,0)$. The angles between the reference direction and all the normal vectors within a cell are determined along with the extreme values $\theta_{\min}, \theta_{\max}$.

Three types of Gauss points are identified. Firstly, points lying on the edge are called edge points. Points that lie inside the boundaries are face points and points that lie on the sharp edges are called sharp edge points.

## 6. 3D Points Reduction

The Gaussian points $s_i$ in each cell are assigned their corresponding estimated normal vector $\vec{n} : i = 1,\ldots,\beta_j$. It is noted that as a consequence of the algorithm 1, cells either contain all edge points or all face points. Edge and face points are readily differentiated since the relative distance between these points in the Gaussian image is large due to the nature their normals. This is especially true for sharp edges [13]. Since cells are treated independently, edges are better preserved during the point reduction process.

The points are placed in different cells based on angles. The angle criterion guarantees that the angle between two normal vectors in a cell is smaller than a prescribe value, $\eta$, i.e. let $\theta_{\max} = \theta_{\min} + \Delta$, then $\theta_{\max} - \theta_{\min} = \Delta$, with $|\Delta| \le \eta$ for each cell. If $|\Delta| \le \eta$, the cell is partitioned again. The angle, $|\Delta|$ characterises the difference between the two normals; if $|\Delta|$ is large, two points have different geometric properties, when $|\Delta|=0$ the two points have identical geometric properties, and when $|\Delta|$ is small, the two points have similar geometric properties.

The average of the angles, $\theta = \beta^{-1}\sum_{i=1}^{\beta_j}\theta_i$, in cell $C_j$ is computed and the nearest angle, to within some user defined tolerance, $\lambda$, to $\theta_i$ and its neighbours is chosen. The corresponding 3D point (and normal) is chosen as representative of the cell whilst all other points are removed.

There are three user defined parameters within the data reduction process, $\lambda$, $\xi$, $\eta$. The two threshold parameters $\xi$ and $\eta$ control the reduction ratio defined as the number of points removed to the original number of points. Decreasing both $\xi$ and $\eta$, increases the reduction ratio. $\lambda$ is a user defined tolerance, which determines how close to the cell average angle a point has to be selected as representative of that cell. Arbitrary values of $\xi = \eta = 0.1$; and $\upsilon = k$ have been used throughout. These parameters can be used to control the overall sensitivity of the reduction process.

**<u>Algorithm 3:</u>**

```
1-Set $j$=1.
2-Define $\vec{N}$ .
3-Select normal vectors $\vec{n} : i = 1,\ldots,\beta_j$ in cell $C_j$.
4-Compute angles, $\theta_i$, between $\vec{N}$ and $\vec{n}$ .
```

5-If $|\theta_{\max} - \theta_{\min}| > \eta$, apply algorithm 1 to further partition the cell.

6-Else compute reference angle $\theta = \beta^{-1}\sum_{i=1}^{\beta_j}\theta_i$

7-Select $\theta_i$, s.t. $|\theta - \theta_i| < \lambda$ $\iota = 1,...,\beta_j$, where $\lambda$ is a user defined value.

8-Find corresponding unit normals of $\theta_i$.

9-Select corresponding original points.

10-Remove residual points in $C_j$.

11-$j$=$j$+1.

12-Repeat steps 3 through 6 until all the cells have been processed.

## 7. Experimental Results

To assess the performance of the proposed reduction algorithm, it is applied to simulated and actual scanned data. Before applying the reduction algorithm, the initial point cloud is pre-processed to remove any gross outliers. For surface normal estimation, experimental evidence suggests that data sets with noise greater than sampling tolerances or that consists of scan lines with large distance variability, a neighbourhood size of $k$=16 is acceptable [10] and this is used in all test cases.

| $\xi = \eta$ | Planar set (1654 points) | | | Cylinder set (12000) | | |
|---|---|---|---|---|---|---|
| | N | M | R Ratio(%) | N | Mean Error | R |
| 0.5 | 1377 | 0.9543 | 17 | 10266 | 1.543 | 14 |
| 0.4 | 1105 | 0.8754 | 33 | 8978 | 1.823 | 25 |
| 0.3 | 979 | 0.3476 | 40 | 6234 | 0.942 | 48 |
| 0.2 | 879 | 0.1011 | 47 | 5143 | 0.321 | 57 |
| 0.1 | 805 | 0.01004 | 51 | 3543 | 0.034 | 70 |
| 0.05 | 577 | 0.01007 | 65 | 1087 | 0.154 | 90 |
| 0.09 | 388 | 0.31004 | 77 | 834 | 1.397 | 93 |
| 0.001 | 210 | 0.31004 | 87 | 654 | 4.674 | 94 |

(a)

| $\xi = \eta$ | Sphere set (12786 points) | | | Cone set (20000) | | |
|---|---|---|---|---|---|---|
| | N | Mean Error | R Ratio(%) | N | Mean Error | R Ratio(%) |
| 0.5 | 11799 | 1.576 | 7 | 18543 | 2.586 | 7 |
| 0.4 | 10347 | 0.7071 | 19 | 15965 | 2.642 | 20 |
| 0.3 | 8634 | 0.088 | 32 | 11987 | 1.734 | 40 |
| 0.2 | 5721 | 0.0517 | 55 | 9643 | 0.678 | 51 |
| 0.1 | 3265 | 0.0034 | 73 | 5932 | 0.983 | 70 |
| 0.05 | 1087 | 1.00326 | 91 | 2314 | 1.546 | 88 |
| 0.09 | 876 | 2.0321 | 93 | 1324 | 3.543 | 93 |
| 0.001 | 670 | 4.1839 | 95 | 1076 | 7.914 | 95 |

(b)

Table (1): (a), (b)Various reduced data sets under changing the parameter $\xi = \eta$

In a typical reverse engineering process such as segmentation, fitting, surface continuity, integration, and reconstruction, once a point cloud has been reduced it would primarily be used to determine the most likely data segmentation, followed by surface fitting, integration, and finally, surface reconstruction(more discussion can be shown in [9]). Thus three tests have been performed on different data types based on these processes. First of all, the performance of the reduction algorithm is assessed by fitting a surface to the data points pre- and post-reduction using standard least squares (LS)[10]. The expectation here is that if the method is performing well, the resulting LS fit should improve as the data is reduced. The error metric is taken as the average of the orthogonal distances between the fitted surface and the points. The second test is to apply the segmentation processes pre- and post-reduction using the proposed method. Finally, the effects of the data reduction algorithm on the process of surface reconstruction are considered.

There are three user defined parameters within the data reduction process, $\lambda$, $\xi$, $\eta$. To find a suitable value of these parameters, we test the proposed algorithm using different data sets under changing the parameters $\xi = \eta$. The reduction ratio of the given data is changed, decreasing both $\xi$ and $\eta$, increases the reduction ratio (R ratio(%)) (see Table (1) (a, b)). For example, when $\xi$=0.5, the reduction ratio (17% of planar set, 14% of cylinder set, 7% of sphere set, and 7% of cone set of points) is smaller than when $\xi$ =0.1 (51% of planar set, 70% of cylinder set, 73% of sphere set, and 70% of cone set of points), but the fitting process becomes unstable for small $\xi$. Here the thresholds for the experiments were fixed at $\xi = \eta$ =0.1, where these values give always accurate results and are stable in all tested cases as shown in Table (1) (a, b). However $\lambda$, and $\mu$ are fixed at 0.025 and 200 respectively

| Surface | N | mse | R degree(D) | R Ratio(%) |
|---|---|---|---|---|
| Plane | 1654 | 0.9670 | 0 | |
| | 803 | 0.010043 | 1 | 51 |
| Cylinder A | 2676 | 1.0094 | 0 | |
| | 1900 | 0.002376 | 1 | 29 |
| | 500 | 0.002029 | 2 | 82 |
| Cylinder B | 1854 | 2.0054 | 0 | |
| | 783 | 0.000092 | 1 | 58 |
| Sphere | 12786 | 2.0054 | 0 | |
| | 3265 | 0.0034 | 1 | 73 |
| Cone | 20000 | 2.7824 | 0 | |
| | 5432 | 0.9832 | 1 | 73 |
| Cylinder C | 12000 | 1.8743 | 0 | |
| | 3543 | 0.0346 | 1 | 70 |

Table (2): Scanned analytic surfaces with simulate noise.

### *7.1 Surface fitting:*

Simulated data sets are considered first, i.e. point clouds are taken from analytic object and then segmented into planar and cylindrical components. To assess the performance of the algorithm in the presence of noise, the point clouds are taken from analytic surface to which a number of randomly distributed noise points are superimposed (1.0% of original data size). The level of noise is generally larger than those typically expected in real data (0.1%) so that the reduction method can be assessed in extreme cases. Due to a scanner measurement, high speed data acquisition can be achieved. These devices can generate thousands of points per second, and now it often becomes a problem to work with these points, because these machines can scan parts with very high speed and much noise points. The reducing points here has two

advantages; first decreasing the noise effect on the reverse engineering processes such as fitting, for instance mse=0.967 in case of 1654 points, but mse becomes 0.010043 in case of 803 points. If the scanner captures the surface data of excellent accuracy, then mse becomes stable in all reduction steps.
Second, to decrease the process time in all reverse engineering process, as an instance, we will work with 5432 instead of 20000 points in all reverse engineering algorithms, especially many algorithms are demanded for creating CAD model from these data. Also computer RAM can not be able to work with these algorithms in case of large data (such as object has several thousand or million points).

The original segmentations are shown in Figures (4a-5a). Applying the reduction algorithm ($k$=16, $\eta = \xi = 0.1$, $\mu = 10000$; $\lambda$ =0.025, $\upsilon$=$k$) gives 65875 points (28% reduction) in the RevolutionBlock set and 42231 points (23% reduction) in the model set. The resulting segmentations are shown in Figures 4(b)-5(b). Comparing Figures 4(a), 4(b), and 5(a) and 5(b) indicates that the data reduction has not degraded the information and gives further evidence that the data reduction algorithm is behaving sensibly. Also, it is noted that the number of the segments is unaffected by the reduction process: RevolutionBlock has 18 regular segments containing 90898 points (of 90974) before reduction, and 18 regular segments containing 64363 points (of 65875) after reduction. For the model data: 11 segments containing 52931 points (of 54854) before reduction and 11 segments containing 42031 points (of 42231) after reduction.

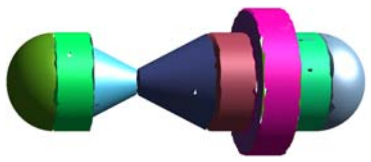
The results of applying the reduction agorithm a number of times (the order) and 9c and the resulting mean errors after performing a LS fit are given in Table (2). For example, 2676 points were sampled from analytic surface: Cylinder part A. Fitting a cylinder using LS resulted in a mean square error (mse)=1.0094. The point set was then reduced twice. For the first reduction D=1, the resulting number of points was 1900 (R ratio of 29%), with mse=0.002376. A second reduction, D=2 gave 500 points (R ratio 82%) with mse=0.002029. The results for all the test cases exhibit the same improved LS fit as it can be seen from Table (2). These results indicate that the proposed method reduces the point set whilst maintaining the integrity of the reduced point cloud, i.e enough data is retained to accurately describe the underlying analytic surface.

### *7.2 Segmenting data sets:*

To assess the affect of the proposed data reduction algorithm on the segmentation process, two different benchmark data sets: RevolutionBlock (90974 points) and model (54854 points) are segmented [10] pre- and post-reduction and the resulting segmentations are visually compared. Here, we use the segmentation algorithm presented in [10] because it is fast and more accurate.
This algorithm will segment the given object into known geometric shapes such as plane, cone, cylinder, and sphere. This gives further evidence that the proposed data reduction method is behaving sensibly.

### 7.3 Surface reconstruction:

As a final check on the integrity of the data reduction algorithm, its effect on surface reconstruction is considered. From previous section, the object is segmented from whole data and reduced data, therefore, each segment has been fitted to a suitable surface.
We will try to intersect the fitted two surfaces (sphere and cylinder) of the model data in Figure 6. If we fail to find the interestion curve in the case of the two surfaces that have been fitted to segments of points before reduction as in Figure 6(b), we then will not able to create the CAD model of a given object (the algorithms will fail).


(a)

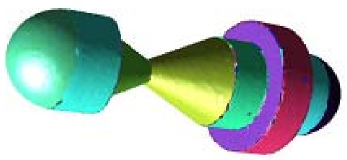
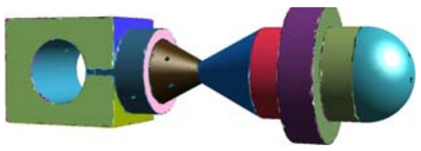

(b)

Figure 4: Segmentation of model data (a) pre- (b) post reduction.


(a)

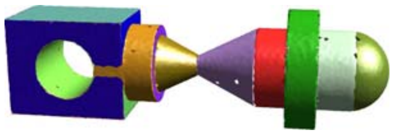
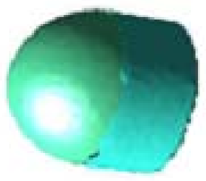

(b)

Figure 5: Segmentation of RevolutionBlock data (a) pre- (b) post-reduction.


(a)

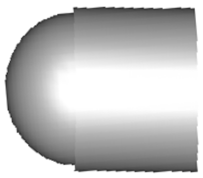
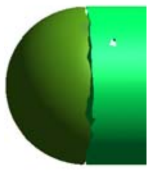
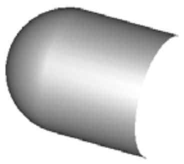

(b)

Figure 6: Before reduction (a) the data set has been segmented (b) fail to reconstruct the two faces.



(a)



(b)

Figure 7: After reduction (a) the data set has been segmented (b) are two faces reconstructed.

An object comprising position continuous, non-intersecting cylinder (1080 points) and sphere (3200 points) segments taken from model data set (Figure 6a) and is shown in Figure 7 (a).
A cylinder segment is fitted to the cylindrical data points resulting in the equation:

$$(x+0.0002249)^2+(y-0.00244)^2+(-0.000015)^2-r^2$$
$$-\left[0.0004615(x+0.0002249)+0.0002543(y-0.00244)+(z-0.000015)\right]^2=0$$

where $(-0.0002249, 0.00244, 0.000015)$ is an arbitrary point on the axis vector $(0.0004615, 0.0002543, 1.0)$ and $r$=2.002 is the radius of the cylinder. Similarly, a sphere segment is fitted to the spherical points, with center $(0.0001, 0.01, -0.00003)$ and radius $r$=2.0. The intersection curve between the two reconstructed surfaces cannot be computed due to the errors induced in the fitting of the original points, as shown in Figure 7(b). The two reconstructed surfaces are not well connected and hence the intersection curve between the surfaces is not available [13].
The two point data sets were reduced using the proposed method ($\eta$ =0.1; $\mu = 200$ ; $\lambda$ =0.025, $\upsilon$=$k$) resulting in a cylinder segment of 578 and a spherical segment of 721 points respectively (Figure 7(a)).
On refitting the cylinder and sphere centers are changed slightly to $(-0.00009, 0.0000056, 0.0)$, $(0.000001, 0.00007, 0.000004)$, the radii to $r$=2.0006 and $r$=2.003. The intersection between the two surfaces can now be computed, for example, using ACIS software package ( Figure 7 (b)) since the reconstructed surfaces are well connected (see[13] for more discussion).

## 8. Comparative Results

In order to evaluate the preformance of the proposed algorithm, we carried out some experiments with different data sets (see section 7). In this section, we compare the performance of the proposed algorithm and Zanaty method [8] that is stable and more accurate than others (more discussion can be seen in [8]). Thus, two tests have been performed on different data types. In the first test, we fit a suitable surface to the reduced points of both methods using standard least squares. The second test is to apply the segmentation processes to both reduced data using the proposed method [10]. The number of neighbourhood is slected $k$=15, and the reduction factor is governed by parameter $\eta$=0.1 in case of Zanaty method [8], while the prameters $\xi$ = $\eta$=0.1; $\mu = 200$; $k$=15 and $\lambda$ =0.025 are given to the proposed algorithm. We denote the resultant points number after appling the proposed method by (NP), and (SP) is the number of points that results after applying established methods [8].

### *8.1 Fitting results*

Mse is estimated in each fitting process. Table (3) presents the comparative fitting results and reduction ratio (%) in each data set when the proposed method (P) and Zanaty (S)[8] methods are applied to different data sets. The results show that our method is more accurate than [8] with factor data reduction of over 50%.

### *8.2 Segmentation results*

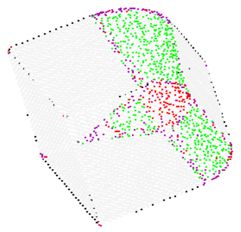
Two different data sets: Bajaj and CurvedBox-curve data set have 16172 and 27792 points respectively. Bajaj and CurvedBox-curve data sets consist of 16 and 12 segments containing 16172 and 27792 points respectively. These sets are reduced using both methods. Then, the reduced sets are segmented [10] and compare the resulting segmentation. Applying the proposed reduction algorithm gives 16 segments containing 10543 points (34% reduction) in the Bajaj set case and 12 segments 15879 points (43% reduction) in the CurvedBox-curve set as shown in Figures 8(a) and 9(a). While the number of segments is changed in [8] to be 15 segments containing 12445 points (23% reduction) in the Bajaj set case, and 13 segments containing 19454 points (30% reduction) in the CurvedBox-curve set as shown in Figures 8(b) and 9(b). Comparing Figures 8(a) and 8(b), and also 9(a) and 9(b) (in case of the number of segments) indicates that our data reduction method improves its shape preserving nature.

## 9. Conclusion and Future Work

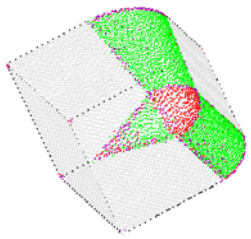
A filter-based 3D shape preserving technique for data reduction has been presented. The approach is designed to handle data sets of various reverse engineering activities, including scanned (laser) of varying density. It can be used to filter data as a post process to data segmentation or surface fitting in reverse engineering. The algorithm uses the discrete Gaussian image which requires the estimation of surface normals. This increases the computation time compared to existing filter-based methods, but this should be deemed an acceptable overhead for a shape preserving reduction algorithm. The representative point of a cell is geometrically selected rather than using a mean or median and is therefore more influenced by the local shape of the point set. The algorithm also differentiates between interior and edge points.
Thus the reduction process is constrained so that edges can be better maintained. This is particularly useful when the density of the scanned data is low along boundaries.

Finally, the reduction process is governed by three threshold values $\xi$, $\eta$ , $\lambda$. $\xi$ controls the amount of data partitioning and $\eta$ controls the density of points within each cell. Thus, the user is able to specify the amount of data to be reduced and control the algorithm to suit specific data sets. For example, a large value of $\xi$ will suffice for planar data, reducing the computational time for the reduction process. For freeform shapes, the selection of values $\eta,\xi$ is governed by the required level of data reduction. However, if the noise level in the data is high, the thresholds can be reduced, effectively increasing the number of partitions of the data and reducing the number of points and hence the variability within each cell. $\lambda$ controls how closely a point has to be to the cell average normal before it can be selected as the cell representative point. A smaller value should be used for extremely noisy data, whilst a more relaxed valued should be used for non dense data. The algorithm has been demonstrated to perform as expected; reducing the data whilst maintaining enough shape information to reproduce the original shape. Further experimentations on data sets varying the threshold values have produced equally encouraging results. It is therefore worthy of further consideration.

As future research, we will extend the algorithm to recognise curvature information which should further improve its shape preserving nature.

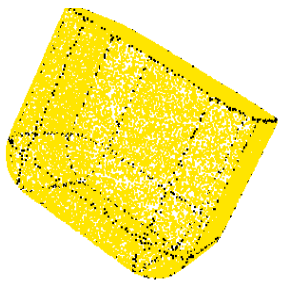


(a)

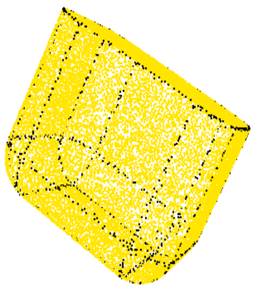


(b)

Figure 8: The segmentation of the reduced points of Bajaj data set using: (a) proposed method (b) Zanaty method[11].



(a)



(b)

Figure 9: The segmentation of the reduced points of model data set using: (a) proposed method (b) Zanaty method[11].

| Surface | NP | Mse of P | R Ratio(%) | NS | Mse of S | R Ratio(%) |
|---|---|---|---|---|---|---|
| Plane | 803 | 0.010043 | 51 | 805 | 0.097 | 51 |
| Cylinder A | 500 | 0.002029 | 78 | 955 | 0.0048 | 65 |
| Cylinder B | 783 | 0.000092 | 59 | 853 | 0.0087 | 54 |
| Sphere | 3265 | 0.0034 | 73 | 8312 | 0.0760 | 35 |
| Cone | 5432 | 0.9832 | 73 | 6400 | 0.9870 | 30 |
| Cylinder C | 3543 | 0.0346 | 70 | 8342 | 0.5637 | 59 |

Table (3): Shows the comparative fitting results and reduction ratio (%) in each data set when applied the proposed method and Zanaty[11] methods to different data sets.

## References

[1] Marshall, A.D. and Martin, R.R., "Computer vision, models and inspection", World Scientific Series in Robotics and Automated Systems-Vol.4, 1992.

[2] Guo, G., Wang H. and Bell D. A., "Data reduction and noise filtering for predicting times series". WAIM, pp. 421-429, 2002.

[3] Uk Jung, "Wavelet-based data reduction and mining for multiple functional data", PhD, Engineering, Georgia Institute of Technology, USA, 2005.

[4] Martin, R.R., Stroud, I.A. and Marshall, A.D., "Data reduction for reverse engineering", RECCAD, Deliverable Document 1 COPERUNICUS project, No 10 1068, Computer and Automation Institute of Hungarian Academy of Science, January 1996.

[5] Lee K.H., Woo H. and Suk T., "Data reduction methods for reverse engineering", International Journal of Advanced Manufacturing Technology, pp.735-743, 2001.

[6] S. Rusinkiewicz, N. Gelfand, and M. Levoy, "Geometrically stable sampling for the ICP algorithm", In Proc. IEEE 3DIM, Canada, 2003.

[7] Yan J., Zhang B., Liu N., Yan S., Cheng Q., Fan W., Yang Q., Xi W., Chen Z. "Effective and efficient dimensionality reduction for large-scale and streaming data preprocessing", IEEE Transcations on knowledge and data engineering Vol.18, No. 3, pp. 1215-1230, March 2006.

[8] Zanaty E., "An efficient method for data reduction in reverse engineering", IJICIS, Vol.6, No.2, pp.75-85, July 2006.

[9] Uhercik M., "Implicit surface reconstruction using local approximation from unorganized 3D points", Msc., thesis, Comenius University, Bratislava, Slovakia, 2005.

[10] Vanco M., Brunnett G., "Direct segmentation of algebraic model for reverse engineering", Computing 72, 1-2, pp. 207-220, 2004.

[11] Ferrie, F. P., Lagarde, J. and Whait, P., "Darboux frames, snakes, and super-quadrics: geometry from the bottom up", IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 15, No. 8, pp. 771-783, 1993.

[12] Cripps, R. J. and Li, X., "Data Segmentation using implicit surfaces", Proceedings of the 34th International MATADOR Conference, S. Hinduja(Ed.), Manchester, pp.75-80, 2004.

[13] Zanaty E.A., Girgis M.R., "Collect the fitted surface into complex based on $C^0$continuity", Journal of universal computer, Vol.11, 6, pp. 1102-1114, 2005.