CAD/CAM Methods for Reverse Engineering: A Case Study of Re-engineering Jewelry

Ioannis Fudos

University of Ioannina, fudos@cs.uoi.gr

ABSTRACT

Reverse engineering is the process of obtaining a geometric CAD model from 3D points acquired by scanning an existing physical model. It is widely used in numerous applications, such as manufacturing, industrial design and jewelry design and reproduction. We argue that for creating editable CAD models meant for manufacturing it is more appropriate to use feature-based constraint-based representations, since they capture design intent. We provide a framework for reverse engineering of small objects and in particular jewelry that combines cross section identification, feature and constraint information exploitation to attain robust, accurate and editable CAD models. First, we extract certain candidate features for describing our point cloud. These features are then reconstructed to describe the solid object. Constraints are automatically detected and maintained. Constraints capture design intent and provide robustness guaranties. Voxel inspired techniques are also employed to describe repeated patterns common to various types of traditional jewelry.

Keywords: reverse engineering, feature-based design, geometric constraints, fitting, tutorial, region growing, CAD, jewelry design.

1. INTRODUCTION

Reverse engineering is the process of obtaining a geometric CAD model from measurements acquired by scanning an existing physical model. The measurements are in the form of 3D point clouds that correspond to points on the surface of the object being re-engineered. Using CAD models to represent the scanned object is very important in various industries because they help improve the quality and efficiency of design. In addition, they speed up the manufacturing and analysis process [3], [35].

Reverse engineering is widely used for various reasons. First of all, by reverse engineering a part, we can obtain the CAD model of a part that is no longer manufactured by its manufacturer or for which only traditional blueprints exist. Also, there are cases where the original CAD model no longer corresponds to the physical part that was manufactured because of subsequent undocumented modifications that were made after the initial design stage. Furthermore, stylists and artists very often create physical models of their concepts by using clay, plaster or wood. These real-scale models should then be used to create CAD models for manufacturing the objects on an industrial scale.

A particularly interesting application of reverse engineering is that of jewelry reconstruction. Jewelry design falls under the category of conceptual and decorative design. In this paper, we present a user-interactive feature-based framework for re-engineering small objects and in particular jewelry that aims at producing robust and accurate models that can either be manufactured or modified to create new jewelry pieces. Our approach exploits skeleton extraction and region growing techniques on the point cloud to automatically extract features and constraints that exist in the scanned object. Then, an iterative process is carried out, through which the user accepts features detected in the previous phase. Finally, the system fits the features according to existing constraints and previously fitted features.

Section 2 presents techniques used for acquiring the point cloud. Section 3 presents two approaches to feature-based reverse engineering. Section 4 provides an overview of geometric constraints for feature based reverse engineering. Section 5 presents issues that arise in jewelry reengineering using the framework outlined in Sections 3 and 4. Section 6 offers conclusions.

2. ACQUIRING THE 3D POINT CLOUD

The first step in reverse engineering is to obtain an accurate and representative 3D point collection usually referred to as *point cloud*. The accuracy should be sufficient to distinguish neighboring points that provide certain structural information. The point density is determined by the sampling rate on the 3D surface. To obtain the 3D point cloud, we can use 3D laser scanning, multiple 2D snapshot reconstruction or a combination of the two.

2.1 3D Laser Scanning

By using laser scanners a large number of 3D coordinates on an object's surface is measured in a very short time. Features such as corner edges or point are not recorded. Instead we have to extract them in the reverse engineering phase.

In the case of ranging scanners, range is computed using the time of flight or a phase comparison between the outgoing and the returning signal. Ranging scanners for distances up to 100m show about the same range accuracy for any range. Triangulation scanners solve the range determination in a triangle formed by the instrument's laser signal deflector, the reflection point on the object's surface and the projection center of a camera, mounted at a certain distance from the deflector. The camera is used to determine the direction of the returning signal. In contrast to the ranging scanners, the accuracy of ranges acquired with triangulation scanners diminishes with the square of the distance between scanner and object [5]. The typical accuracy in close range laser scanning ranges from 0.4mm up to 19.7mm in terms of standard deviation from the correct values. However most systems have a standard deviation in the range of 1.5mm to 3.5mm [5].

Also laser scanning may exhibit erroneous behavior due to the following two effects:

- The *edge effect:* even when well focused the laser spot will have a certain size. When the spot hits an object edge only a part of it will be reflected there. The rest may be reflected from the adjacent surface, a different surface behind the edge or not at all.
- The *shiny surface effect*: Laser scanners have to rely on a signal reflected back from the object surface to the receiving unit in case of ranging scanners and to the camera in case of triangulation scanners. In either case, the strength of the returning signal is influenced (among other facts such as distance, atmospheric conditions, incidence angle) by the reflective abilities of the surface. White surfaces will yield strong reflections, whereas reflection from black surfaces is weak. The effects of colored surfaces depend on the spectral characteristics of the laser (green, red, near infrared). Shiny surfaces usually are not easy to record.

2.2 Obtaining 3D Points from 2D Snapshots

We may use 2D snapshots to reconstruct a 3D object. This can be accomplished by using calibrated cameras placed in exact positions and then employing photogrammetry methods for reconstruction. Alternatively, we can use plain photographs from portable cameras and then use sophisticated methods to reconstruct the 3D scene. We prefer the latter method for a number of reasons:

- It is non destructive and easy to setup, we can obtain several snapshots from museum exhibits, private collections etc, where we cannot setup a photogrammetry setting or use laser scanning.
- We do not have the shiny surface and edge effect that both occur quite frequently in jewelry.
- By using many snapshos or a video, we can obtain very accurate results.

In this case we assume that we have a portable camera by which we take pictures of a scene and we wish then to re-construct the three-dimensional object. It is important to know the way this particular camera formulates the images on the film or CCD sensor. This is equivalent to knowing the intrinsic parameters of the camera. These parameters are different for every camera and depend on the manufacturing characteristics of the camera and the topological placement of the lens when the photographs were taken. The camera parameters may be available from the manufacturer, but that is not always the case.

A camera with known intrinsic parameters is referred as a calibrated camera, and the procedure of acquiring these parameters is called camera calibration. So we first need a method for calibrating a camera. That means that we must device ways to obtain these parameters, through the process of calibration. We have developed efficient methods for estimating the intrinsic parameters of the camera. The calibration developed in our work is based on the work of Zhang [39], [40].

We use the view of a calibration pattern consisting of black and white checkerboard with squares of known length. We assume that the planar object lies on Z=0. A planar calibration object point **X** is related to its image point **x** by a 3x3 homography matrix H. The homographies can be estimated by selecting the four corners of the calibration object. We use a maximum likelihood criterion to estimate the homography. This is a nonlinear optimization problem that we solve using the Levenberg-Marquardt optimization algorithm.

We then use the epipolar geometry of the 2D snapshots to obtain 3D points by triangulation using the fundamental matrix which expresses the epipolar constraint. To estimate the fundamental matrix we employ the eight point algorithm. To obtain a more dense point cloud, we use structured light projected on the object and sophisticated image analysis techniques to detect matching points. This is a central issue in the 3D reconstruction process and there is much ongoing research in this area [17], [25].

2.3 Hybrid Methods

We can combine laser scanning results with 3D reconstruction from 2D images. By doing so, we increase the accuracy of both methods and eliminate the problems that arise from the shiny surface effect and the edge effect in laser scanning technology. To benefit from a hybrid technique we need efficient algorithms for detecting matching points in 2D snapshots and identifying their 3D point counterparts in the laser scanning outcome. Point clouds that correspond to shiny surfaces and edges are replaced from the 3D reconstruction results.

3. FEATURE-BASED REVERSE ENGINEERING

A current trend in reverse engineering is the use of feature-based models and methods. Feature-based models are convenient for manufacturing mechanical parts, where there are well defined relationships among the different parts of the model [6]. Also, feature-based models are ideal for industrial design and manufacturing since the model produced can be easily modified. This is due to the knowledge provided by the model concerning tolerances, constraints, relationships and connectivity among features [6], [35].

Feature-based and constraint-based methods are often characterized also as knowledge-based. Their main objective is to exploit any knowledge and information that is related to design intent, functionality and construction process issues of the object being re-engineered [29]. As reported in [38], it is useful to exploit design intent and feature relationships that exist in models created for industrial use, because they justify some of the attributes of the object that are obsolete, if they are not related appropriately. Such information may be expressed through the use of geometric constraints.

There have been many projects that have focused on such approaches. The REFAB project [24], [28], [33], [34] uses a feature-based and constraint-based method to reverse engineer mechanical parts. REFAB is a human interactive system where the 3D point cloud is presented to the user, and the user selects from a predefined list a feature that may exist in the cloud, specifies the approximate location of the feature in the point cloud. The system then fits the specified feature to the actual point cloud data using a least square means method iteratively. The authors place emphasis on the fitting of pockets, where the user draws a profile of the pocket on the point cloud and the system fits the profile to the data. The profile is then extruded to create the pocket. This feature-fitting process is made more accurate by using constraints that are detected by the system, verified by the user and then exploited to achieve a better fitting of the features to the data. The system supports constraints such as parallelism, concentricity, perpendicularity and symmetry. The constraints defined and used in REFAB aim at reducing the degrees of freedom associated with the object as much as possible, so as to achieve high precision CAD models efficiently.

A feature-based reverse engineering method is used in [1] for reverse engineering a mannequin for garment design. The basic idea of this method is to create a generic mannequin model of a human torso, which is appropriately aligned with the 3D point cloud of the desired human torso model. The generic model is "fitted" to the point cloud by matching up characteristic points of the models e.g. peaks. This method creates parameterized models by exploiting the features of the object and by using them to constrain the fitting process. It provides an automated approach to reverse

engineering human torsos. This approach creates parameterized models with acceptable accuracy. However, it is difficult to be applied for reverse engineering small objects and such as jewelry because of the variety of free form designs. If the type of jewelry being reverse engineered is of a specific type, with specific features, then this method could prove useful.

In [10], another approach to reverse engineering is presented in which a priori knowledge is applied and expressed through constraints. This work mainly focuses on determining a set of regularities, and from this set choosing the subset of regularities that best describe the problem and are consistent to each other. This is also the aim of the work in [19], where consistent regularities are detected for the beautification of the object that is reverse engineered.

The main focus in all of the above work is to exploit any knowledge that is available about the initial object and the parameters, features and constraints that it contains. By using this information, we can more efficiently create and manipulate part characteristics to create more advanced models.

Nest, we present two techniques that can be used for detecting features in point clouds: skeleton extraction and boundary morphology analysis. In Section 4, we will introduce the use of geometric constraints for supporting and enhancing the techniques presented here.

3.1 Automated Skeleton Extraction

Our approach to reverse engineering solids starts from a 3D point cloud that has been processed and stripped from noisy data. Our goal is to obtain information from this 3D cloud data about the topology and shapes that exist in the initial object that has been scanned. To achieve this, we use a method for detecting and interpreting the axes of symmetry that exist in the object to determine the type of object that is to be reverse engineered. This method draws upon the concept behind the medial axis. A medial axis of a three dimensional shape is the closure of all points that have more than one closest point on the shape boundary [37]. The medial axis reflects the symmetries of a solid object. We are not interested in computing the exact medial axes or all the axes of symmetries, but only those that are useful in determining the general shape of the object. For example, in the case of reengineering jewelry, if by detecting the symmetry axes in a 3D point cloud we find that one of them is a circular surface that passes through all or a large part of the cloud points, then we can conjecture that the 3D cloud represents a ring or a bracelet.

We start by examining cross-sections of the point cloud and by determining the 2D medial axis of each cross-section. The medial axis of a two dimensional shape is the closure of the locus of the centers of all maximal inscribed discs [37]. We detect the medial axis of each cross section and then interpolate these medial axes in order to create a 3D skeleton of the point cloud which is an approximation of the medial surface. An alternative definition of the medial surface is the union of the medial axes of all cross sections.

Many methods have been proposed for computing the 3D skeleton of an object, such as [8]. However, since we are not interested in exactly computing the medial surface, we just derive a sufficient approximation of the medial surface to reproduce the general shape of the object. Computing this approximate skeleton is useful in the subsequent steps of the method, where feature and constraint fitting is introduced. This is because knowledge of the skeleton can assist the automatic feature and constraint detection phase. For instance, if the skeleton contains points that branch out and are end points, then we can assume that the shape contains some sort of end point/angle at that location. For instance in Fig. 1 the left branch of the medial axis corresponds to a sharp angle in the object form. Conclusions about the shape morphology can also be derived from the angles that are formed between the branches of the medial axis.

Another example of the usefulness of the skeleton is in the case of a simple cylindrical-shaped ring, the axis of symmetry that corresponds to the ring circumference is a circular surface that is concentric with the circle that coincides with the cross-section of the cylinder. Therefore, by detecting the symmetry axes in a 3D point cloud, we find that one of them is a circular surface that passes through a large part of the point cloud. This fact along with some other conditions, leads us to the conclusion that the 3D cloud represents a ring or a bracelet.

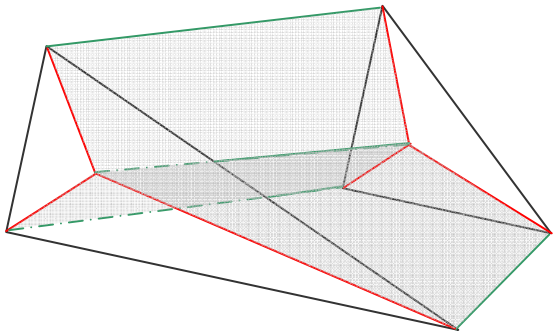

Fig. 1. Two triangular cross-sections, their medial axes and the interpolating edges used to create the medial surface.

3.2 Detecting Features by Boundary Morphology Analysis

We start with a point cloud which we assume has been preprocessed to remove noise and duplicate points (Fig. 2). The first step in the feature-based reverse engineering process is to partition the point cloud into individual components, such that each component corresponds to a single feature.

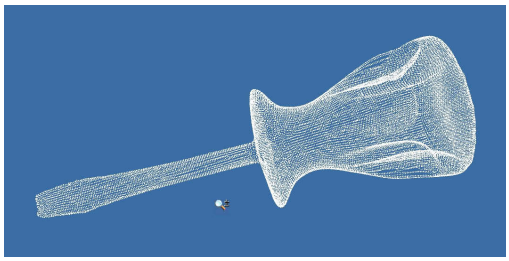

Fig. 2. The point cloud of a screwdriver (27152 points).

Shape and point cloud decomposition methods can either be carried out automatically or semi-automatically (through a user-interactive process). For instance, in [4], a direct segmentation method is proposed where the point cloud is separated into smaller subregions until no further subdivision can be performed. The subdivision is based on a special sequence of tests. In [15], a triangulated mesh is constructed to determine the surface topology and then different border detection approaches are performed to segment the point cloud. In [24] and [34], a user-defined approach to point cloud data segmentation is followed. Specifically, the user picks a feature from a list and points out its location in the point cloud. Then, the system fits the feature to the point cloud.

Regarding feature extraction and definition, in [13], features are extracted from point clouds basically as a preprocessing phase for surface reconstruction. The features detected in this context are feature lines. Feature lines are also the focus of [7].

In [21], feature definition is performed with the help of the skeleton. The authors present a shape decomposition and skeletonization method for polyhedrons that is based on approximate convex decomposition [15]. The convex hull of the polyhedron is computed and the concavity of the vertices is calculated and used as a criterion for the decomposition of the object. The concavity of a vertex is defined as the distance from the vertex to the convex hull surface of the component/polyhedron. The polyhedron is partitioned into components at locations where the concavity of the vertices is high. An iterative method is then used to simultaneously find the most efficient shape decomposition and the skeleton of each decomposed component. We have further extended these methods to decompose point clouds into components that represent candidate features.

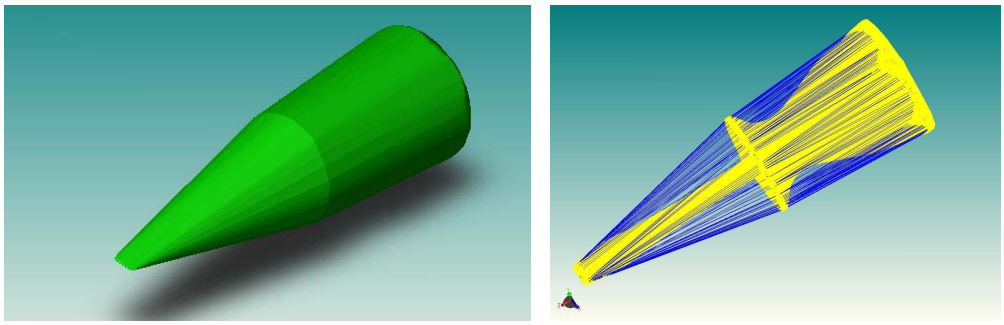

Fig. 3. The convex hull of the screwdriver.

Specifically, we compute the convex hull of a point cloud using Qhull and then calculate the intensity of each point of the point cloud (Fig. 3). We define *intensity F(P)* as the function that represents the distance of a point P to the convex hull. Fig. 4 displays the intensities of points belonging to two different point clouds, a screwdriver and the Stanford bunny. The intensity values are rendered using grey scale, where black corresponds to points belonging to the convex hull, whereas white corresponds to points that are farthest away from the convex hull (largest distance). As shown in Fig. 4, changes in the intensity value can be detected by edge-like curves and saddle points. These characteristics are then used to separate the point cloud into components that define the features of the object.

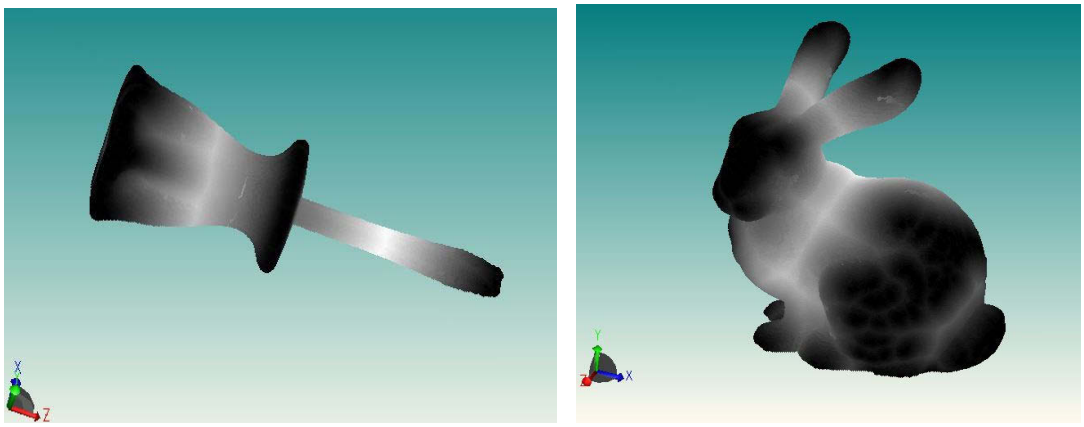

Fig. 4. Representing distance by point intensity.

Changes in the intensity can be detected by applying methods used in digital image processing and computer vision for edge detection and image segmentation. Specifically, in 2D images, segmentation algorithms are based on either discontinuity or similarity of the intensity values of neighboring pixels. In the former case, the idea is to partition an image based on abrupt changes in the levels of intensity (e.g. gradient methods) whereas, in the latter case, segmentation is based on s region growing and thresholding.

We have applied these principles to detect edges and saddle points. Points where there is a peak in intensity value are possible edge points. Saddle points are contained in an area where the variation of the intensity values is virtually zero. For each point in the point cloud, we examine the intensity of its neighbors to determine how intensity varies in this area. The neighborhood graph of the point cloud is constructed after creating a triangulation of the point cloud. A neighbor of point P is every point with which P is connected by an edge belonging to a triangle (see Fig. 5). We compute the first order gradient of each point to determine if and where we have a local maximum or minimum.

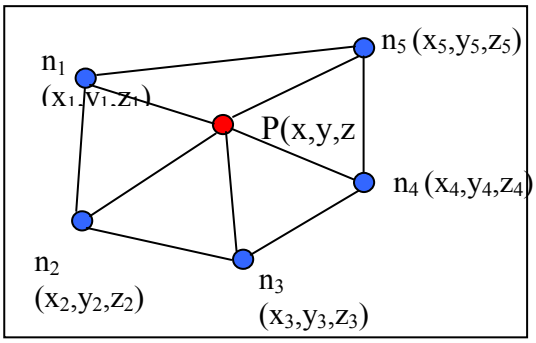

Fig. 5. A neighborhood of points.

For each point P belonging to the point cloud we compute an approximation of the gradients: ∇Fx , ∇Fy , ∇Fz , of the intensity *F*:

$$\nabla Fx = \frac{\partial F}{\partial x}, \nabla Fy = \frac{\partial F}{\partial y}, \nabla Fz = \frac{\partial F}{\partial z}$$

To compute an approximation of each of the gradient coordinates for every point P of the point cloud, we first sort the neighbors by each direction coordinate. For example, for computing an approximation of the gradient ∇Fz of point P(x,y,z), we sort all of the neighbors by their z-coordinate. Next we find the two neighbors (or at least one) that are closest to P in terms of Euclidean distance and have a z coordinate that is smaller and larger than the z coordinate of P ($P_1(x_1,y_1,z_1)$ and $P_2(x_2,y_2,z_2)$ respectively).

$$P_{1z} < P_z < P_{2z}$$

We then calculate the gradient ∇Fz for each pair (P_1,P) and (P_2,P) as follows:

$$\nabla Fz_1 = \frac{F(P) - F(P_1)}{z - z_1}, \qquad \nabla Fz_2 = \frac{F(P) - F(P_2)}{z - z_2}$$

From these two gradients, we calculate the average gradient

$$\nabla Fz = \frac{\nabla Fz_1 + \nabla Fz_2}{2}$$

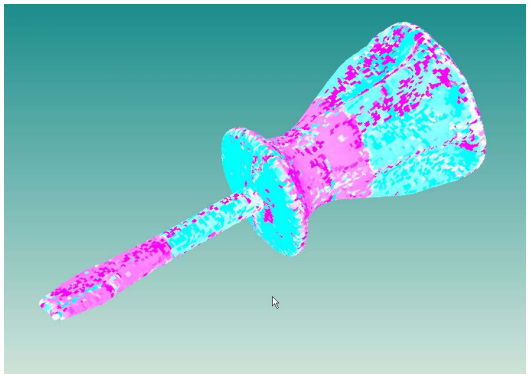

Fig. 6. ∇Fz values of the screwdriver - Magenta corresponds to negative gradient values, cyan corresponds to positive gradient values, whereas white corresponds to zero gradient.

To render the calculated gradient intensity values and to determine the behavior of the gradient in a specific direction, we map each point's gradient value to an rgb color. We map the maximum (positive) gradient value to cyan and the minimum (negative) gradient value to magenta (see Fig. 6). Intermediate gradient values are mapped to a tone of the above colors (a tone of cyan for positive values and a tone of magenta for negative values). White points correspond to points of gradient value zero or almost zero. The gradient values of the points form a Gaussian distribution and we calculate the standard deviation, in order to determine which gradient values will be used as maximum and minimum to perform the mapping.

This approach gives us an initial decomposition of the point cloud. For instance, in the above ∇Fz , we can determine the following initial decomposition of the point cloud into features (Fig. 7):

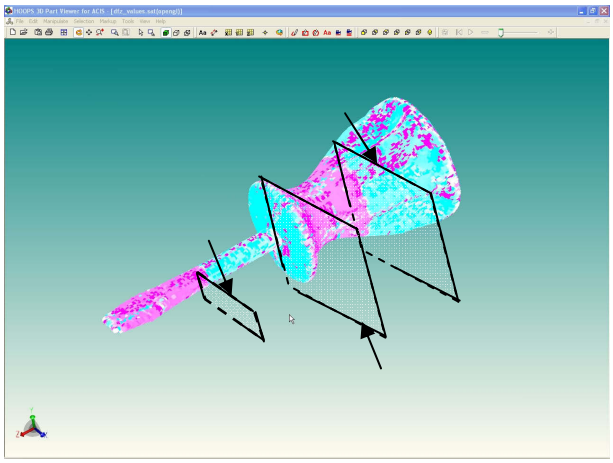

Fig. 7. A decomposition of the point cloud into 4 basic point sets, based on the changes of ∇Fz.

We are currently enhancing the region growing method to find saddle points in the intensity values, where the intensity remains almost steady over a considerable area.

4. EXTRACTING AND MAINTAINING GEOMETRIC CONSTRAINTS

The importance of being able to accurately describe a piece of geometry in a clear and unambiguous way has been realized since ancient times. For simple 2D shapes, Euclidean geometry showed us precisely what information was needed to completely define them. However, the difficulties of describing more complex geometries remained unresolved for many years until the use of computers became common. With the use of CAD/CAM, it was soon realized that the increased ease of entering more and more complex geometries further necessitated the invention of novel formalisms capable of capturing these geometries in a simple but rigorous manner. A number of authors have conducted research in the field of geometric constraints and regularities for reverse engineered models.

Feature based design has been approached as an extension of the CSG (Constructive Solid Geometry) paradigm [30]. In a CSG construction, a solid is built from standard primitives by regularized Boolean operations. The solid then, is represented by a tree structure in which the leaves are solid primitives and the interior nodes are Boolean operations and rigid-body transformations. Although limited, this approach provides well-defined design semantics.

In a more recent work [6], semantics for the creation of generated features are defined. This work is based on a neutral, high-level design representation, called *Erep (editable representation*), which allows design modifications based on a general design paradigm. This framework considers constrained generated features based on a planar profile and then revolved, swept, extruded, etc in 3D shape.

When using the term *constraint* in CAD we refer to the dimensions and relations (lengths, angles, tangency etc.) used to define a particular solid geometry. Geometric constraints and regularities have been commonly characterized for the primitive surfaces (plane, sphere, cylinder, cone, and torus) [22]. In this section we describe the most common

constraints used in the current research to capture design intent. By introducing constraints we enforce exact geometry as opposed to the inherent approximate Brep that is built solely on the point cloud.

One of the central research directions in computer vision is 3D object recognition. In 3D object recognition the term of surface characterization refers to the computational process of partitioning surfaces into regions with equivalent characteristics. Many algorithms have been developed for surface characterization that make use of differential geometry. The local shape of a surface is central to object recognition. It may be determined with the use of Gaussian and mean curvatures which combine the first and second fundamental forms of the surface [18] to obtain scalar surface features which are invariant to rotation, translation and re-parameterization. Surface shapes may be characterized by the sign of the mean curvature and Gaussian curvature. These curvatures may be computed directly from the point set acquired from the data acquisition process.

In [26], a laser projection system and an image processor [27] are used for determining a fixed set of horizontal cross sections of the recognized object which is placed on a turntable in a stable vertical orientation. For each horizontal cross section boundary based Fourier shape representations are computed. Constraints between two cross sections may be defined such as horizontal strain, section shape, torsion, and displacement.

Constraints are often imposed to enhance the surface fitting step. Then a simultaneous (as opposed to sequential) fitting is attempted using the constraints as a set of side conditions that must be satisfied by the surface parameters [2].

Another approach [34] is to drive the segmentation and surface fitting phases using pre-defined features like slots and pockets whose abstract location and type has been determined by the user.

[22], [38] describe the importance of a post-processing step, often called *beautification*, which adjusts the model to reflect more closely the intended object. This step involves the analysis of the model to find geometric regularities, the selection of an appropriate consistent set of regularities which renders the original design intent, and finally, the reconstruction of an improved model using geometric constraints without further reference to the point data which avoids the computational expense of constrained fitting.

The set of constraints associated with a given object may be divided into two categories: the first one is associated with the detection and properties of certain features. These constraints that enforce the specific shapes of the surfaces are called *intra-feature topology*. The second is associated with the overall geometric and topological relationships among the object features and is called *inter-feature topology*.

4.1 Intra-feature Topology

The model may suffer from inaccuracies caused by sensing errors from the data acquisition phase, approximation and numerical errors arising from the successive algorithmic steps, or possible wear of the original object. The resulting inaccuracies may be categorized as follows: gaps in a single face, gaps crossing an edge, gaps spanning multiple faces, pinched faces, chains of small faces, sliver faces, chains of short edges, adjacent faces with same geometry, isolated small faces, adjacent edges with same geometry, isolated short edges.

In [23], [12] the authors use the notion of tolerance to repair local topological problems. Tolerance may be thought of as a scalar value that is user-defined or automatically derived by an algorithm. The rule that may determine the tolerance is that the length tolerance should be larger than the size of any small face or short edge which is to be deleted, but smaller than any part of the model that is to be retained. When possible a single global tolerance value is defined for the entire model. But in many cases this is not feasible since different regions of the object may have been scanned at different resolutions due to data acquisition technology restrictions. In such cases a more sophisticated approach with adaptive tolerance is adopted.

In a model obtained by reverse engineering many topological deficiencies of different types coexist. To eliminate such problematic structures efficiently we need to detect and correct deficiencies in a specific order so as to avoid an iterative correction process.

To maintain topological consistency [23], [12] for certain primitive surfaces a process of repairing certain topological deficiencies of the model is used. [38]uses analytical geometry laws where the five primitive surfaces may

be defined by a number of equations. As a result the constraints that may be imposed on a particular primitive surface will be in terms of these equations. For instance we may detect/enforce the following:

- the circularity of a cylinder: all points on a surface of a cylinder intersected by a plane that is perpendicular to the cylinder axis are equidistant to that axis.
- the circularity of a cone: all points on a surface of a cone intersected by a plane that is perpendicular to the cone axis are equidistant to that axis.
- the sphere constraint: An ellipsoidal cross section represents a perfect sphere.

The above topological deficiencies have to be identified and repaired prior to imposing geometric constraints on the object. The process of gaining topological intra-feature consistency may be performed in the following steps:

- Detect topological deficiencies inside features.
- Adjust the topological structure of features to repair the topological deficiencies.
- Constrain the feature's geometry to ensure that in the resulting structure, the faces, the edges and the vertices have the desired connectivity and contact.
- Regenerate the model's geometry by solving the system of constraints derived in the previous step.

4.2 Inter-feature Topology

Features can be revolved, extruded, or swept to become 3D solids. For instance in [20], [22] the common notion of generating primitive surfaces by rotating a planar generating feature about an axis in that plane is used. For instance, the generator of a sphere is a circle with axis through the center, and the generator of a cone is a line that intersects the axis.

Surface	**Generator**	**Characteristics**	**Numericals**
Plane	Line orthogonal to the axis	Normal, Axis	Distance
Cylinder	Line parallel to the axis	Axis	Radius
Cone	Line intersecting the axis	Axis, Apex	Angle
Sphere	Circle with axis through the center	Centroid	Radius
Torus	Circle with axis outside the circle	Axis, Circle of Rotation, Centroid	Major Minor radii

Tab. 1 - Primitive surfaces and generators.

Since the rotation axis of each cylinder, cone, torus is unique, the axis is a characteristic of these surfaces. Toroidal surfaces have also the characteristic circle of rotation. A sphere has the characteristic centroid. A cone has the characteristic apex, which is at the intersection of the line and the axis. For the case of a plane, we introduce an origin on the plane and an orthogonal line passing through the origin (normal). The above characteristics are fundamental but not enough to specify a particular primitive surface. For instance, to specify a cylinder we need the axis but we also need the radius of the cylinder. Table 1 summarizes some primitive surfaces and their generators.

Inter-feature topology may be now enforced using the above characteristics. In particular, we look for approximately equal shape parameters from surfaces and edges such as radii, lengths, and angles. Table 2 summarizes shape regularities that may be imposed on shapes.

	Constraint Type
Shape Parameters	Equality of shape parameters (radii, edge lengths, Congruent faces)
	Special values for shape parameters

	Simple integer relations between shape parameters (2:1 radius ratio, 2:1 edge length ratio)
Axis Directions	Parallel axis directions
	Sets of axis that have the same direction
	Symmetrical arrangements of axis directions
Axes	Intersection of axes in a point
	Aligned axes
	Parallel axes arranged along lines with regular distances between them
	Parallel axes arranged symmetrically on cylinders
Positions	Equal positions
	Regular distances between positions on a line, 2D or 3D grid
	Equal positions under projection

Tab. 2 - Shape Regularities.

In Table 2 we distinguish a few constraint types where the notion of regular distance is introduced in [22](in [38] there is a similar notion called *relative separation*). Therefore, a collection of coplanar parallel lines with constant distance between adjacent lines may be used in cases where some pair of object feature axes should be parallel and in constant distance between them. A change in distance between a pair of adjacent lines implies a change in distance of all pairs of adjacent lines. The relative separation between features is the distance between parallel features. The same notion may also be used in a collection of parallel circles on cylinders. Similarly to regular distances, the notion of regular angles is introduced in [22] (in [38] there is a similar notion called r*elative orientation*) and it may be used as global constraint. Coplanar lines with a common point and constant angle between adjacent lines are said to have regular angles between them.

A part may contain features which are associated with the same geometric entity or which coincide at the same position. In other words two edges may belong to the same infinite line and two faces may lie on the same infinite plane. This type of constraint is called *coincidence constraint.*

A specific feature point may lie on an object feature (line, plane, etc). This constraint requires that the point satisfies the feature equation. This type of constraint is called *inclusion constraint*.

For the containment of a line in a cylinder surface the line and the cylinder must have the same orientation and an arbitrary point of the line must satisfy the equation of the cylinder. Similarly, for the containment of a line in a cone the orientation vector of the line must satisfy the homogeneous equation of the cone and the line must pass through the cone summit.

The process for extracting inter-feature constraints is outlined below:

- Based on consecutive features determined in previous phase, we extract constraints that bind features in forming a certain 3D surface. We usually work with low degree surfaces (planar and conics).
- We then detect relationship among surface characteristics. For example, we may detect concentricity of a torus and a sphere.
- We Regenerate of the model's geometry by solving the system of constraints derived in the previous step.

4.3 Maintaining Topology through Constraints

The last step in this process is always the maintenance of constraints. There are various methods that we may use for enforcing geometric entities to conform to a system of geometric constraints. See for example [11] and [14].

The intra-feature constraints are local and usually 2D. In this context we use a graph constructive method enhanced by randomized optimization techniques. The regeneration of the placement of geometric entities to conform to the set

of geometric constraint is usually successful and most importantly it maintains the relative topology of geometry to a large extent [11].

In inter-feature topology, the resulting 3D system of geometric constraints is technically more difficult to solve. We may use the frontier algorithm presented in [14] to detect a minimal dense constraint subgraph. We then employee randomized optimization methods to solve the subsystems. The problem of maintaining the relative topology of geometry in 3D scales disgracefully and even interactive methods are inefficient. A rule of thumb is to minimize the number of inter-cluster constraints that cannot be expressed in a sequential construction to bind features together.

5. RE-ENGINEERING JEWELRY

Jewelry design falls under the category of conceptual and decorative design. We can distinguish two categories of jewelry: free form jewelry and jewelry that conforms to certain patterns and constraints e.g. repeated patterns or specific gem cuts. Re-engineering of objects that fall under the first category is difficult to be automated, whereas the second type of jewelry can be approached using combinations of voxel-based and feature-based techniques.

Reverse engineering jewelry requires that the CAD models created are accurate and robust. These models should be parameterizable to support custom jewelry design. Furthermore, the user-designers should have the capability to modify the re-engineered CAD model according to their preferences, to create novel designs. For instance, in the case where a ring is re-engineered, we would like to be able to modify its dimensions to produce rings of larger sizes, or we would like to be able to choose certain parts of the object to use them to create other pieces of jewelry as sets of pieces, e.g. a matching set of earrings. To this end, one needs to exploit the features of the original jewelry model and the relationships and constraints that hold between them. By applying constraints and fitting features to the point cloud we can enhance the semantics model and achieve better accuracy and parameterization.

It is very difficult to develop fully automated reverse engineering systems where there is no human interaction. It is more appropriate to design systems where the user interacts with the system and provides information that can be used to acquire a more accurate and complete CAD representation of the object. Our goal is to make the system as automated as possible, by minimizing user interaction, without sacrificing real time response and high accuracy. We do so by optimizing the usage of the knowledge of the application domain. This approach has been adopted by many systems, such as [10], [34].

Most work in the literature refers to reverse engineering of mechanical parts and objects of industrial design. In [16] a reverse engineering system is developed for re-engineering rings. A 3D point cloud is generated from the coordinates captured from a CMM (co-ordinate measuring machine) system and the CATIA CAD system is employed to interactively concatenate the points to obtain curves that fit the data. From these curves (polynomial curves) surface patches are created, and these surfaces are then used to generate a Brep representation of a solid model. This method creates basic generic models of the initial ring object. Then, to manufacture the ring, the 3D model is transformed into a 2D representation on which the engraving of the ring design will be performed. The authors use radial, planar cuts to obtain a flat representation of the ring. This method is appropriate for creating blank generic models of rings that a designer will then use to create his/her own ring model, since the system also offers a library of precious stones and popular ring settings that the designer can use. However, this is not a complete reverse engineering process in the sense that the system basically creates blanks on which engraving is performed based on engineering drawings. This method is not useful in the case where the drawings are not available.

By observing different pieces of jewelry we notice that in most cases there are symmetries present in the object. These symmetries may concern either the whole piece of jewelry that we express with inter-feature constraints, or local areas which we express as intra-feature constraints. These symmetries can be exploited to produce more accurate and robust models and to reduce the time needed to re-engineer the initial model.

In summary, the aim of our work is to create robust CAD models from 3D point clouds of jewelry. We do not necessarily intend to create exact replicas of the original pieces. Our goal is to create models resembling the original piece as far as the craftsmanship is concerned. The resulting models should also be fully parameterizable and robust, so that they can be modified and remanufactured.

We propose a method based on extracting a skeleton of the point cloud enhanced by user intervention and then detecting some initial features and constraints. These features and constraints are then used in an iterative user-

interactive process where the user defines features and constraints that are then fitted to the point cloud. The features,are represented using voxel-based elements that are combined together to form more complex shapes.

5.1 Skeleton Construction in Jewelry

The skeleton of an object provides information regarding its shape morphology. The skeleton of an object can be used for automatically extracting features and constraints from the point cloud. Small objects and in particular jewelry have a certain morphology of a small genus as opposed to mechanical parts. We present two different interactions to calculating the skeleton of the point cloud depending on the genus of the object, which represent the number of handles that exist in the object.

Initially, we orient the point cloud by placing it in a convenient direction in the coordinate system. We obtain the diameter of the point cloud, i.e. the point pair with the largest Euclidean distance. We place the point cloud in such a way so that the one point of the pair is on the origin of the coordinate system (0, 0, 0) and the other is placed on the z axis. Following this, we find the correct orientation for the point cloud around the z axis. We determine which pair of points, when projected on the xy plane, form the largest distance and rotate the cloud around the z axis until the alignment is such that the largest projection pair is located on the y axis. After this, our point cloud is properly orientated and we create a bounding parallelepiped around the point cloud (Fig. 8(d)).

We then define a planar surface parallel to the top and bottom surfaces of the bounding parallelepiped. We use this surface to "cut" the point cloud into slices by sweeping it along a path and we examine if each slice contains one or more closed shapes. If there is a large "space" between pairs of points then it is very possible that a hole is formed at that location. This can be further looked into by deriving more slices around that area and by examining the formation of the points. If we determine that there is a hole in the point cloud then we can also determine its diameter and consequently its center. If there is a hole present in the point cloud then the object is of genus at least 1, otherwise it is of genus 0.

After determining the genus of the point cloud, we continue on to construct the skeleton of the object depending on the genus. The skeleton is constructed by combining the medial axes calculated on individual cross-sections of the point cloud as outlined in Section 3. Therefore, it is important to determine a method for deriving the cross-sections of the point cloud that are going to be used for the skeleton construction phase.

In the case that the point cloud is of genus 0, we consider a sphere that is created at the center of the bounding parallelepiped by the intersection of the diagonals of the bounding parallelepiped. We enlarge this sphere until it "touches" a point of the cloud. Around this area we form a local surface and calculate a normal for it. We consider a planar surface that passes through the center of the sphere and contains the normal of the point cloud at the local boundary surface. We then rotate the surface around the center of the sphere and determine the smallest cross-section that is obtained. The smallest cross-section is used as a starting point for "scanning" the point cloud and deriving cross-sections that are to be used in the skeleton construction phase. From the initial cross-section we "walk" on a parallel path to the faces of the bounding parallelepiped.

In the case where the object is of genus 1, then it is best to exploit the knowledge of where the hole in the object is. When determining the genus of the object we obtain information about where in the point cloud the hole is located and which is its diameter and center coordinates. We use the center of the hole to create a sphere which we continuously enlarge until it intersects the point cloud at a certain area of points. At this small surface we consider a normal on the local surface and we create the planar surface that passes through the center of the sphere and contains the normal. The intersection of the surface and the point cloud gives us one of the cross-sections. All other cross-sections are derived by sweeping the surface by angle a around the axis that goes through the center point and is perpendicular to axis z, by which the point cloud was oriented (Fig. 9).

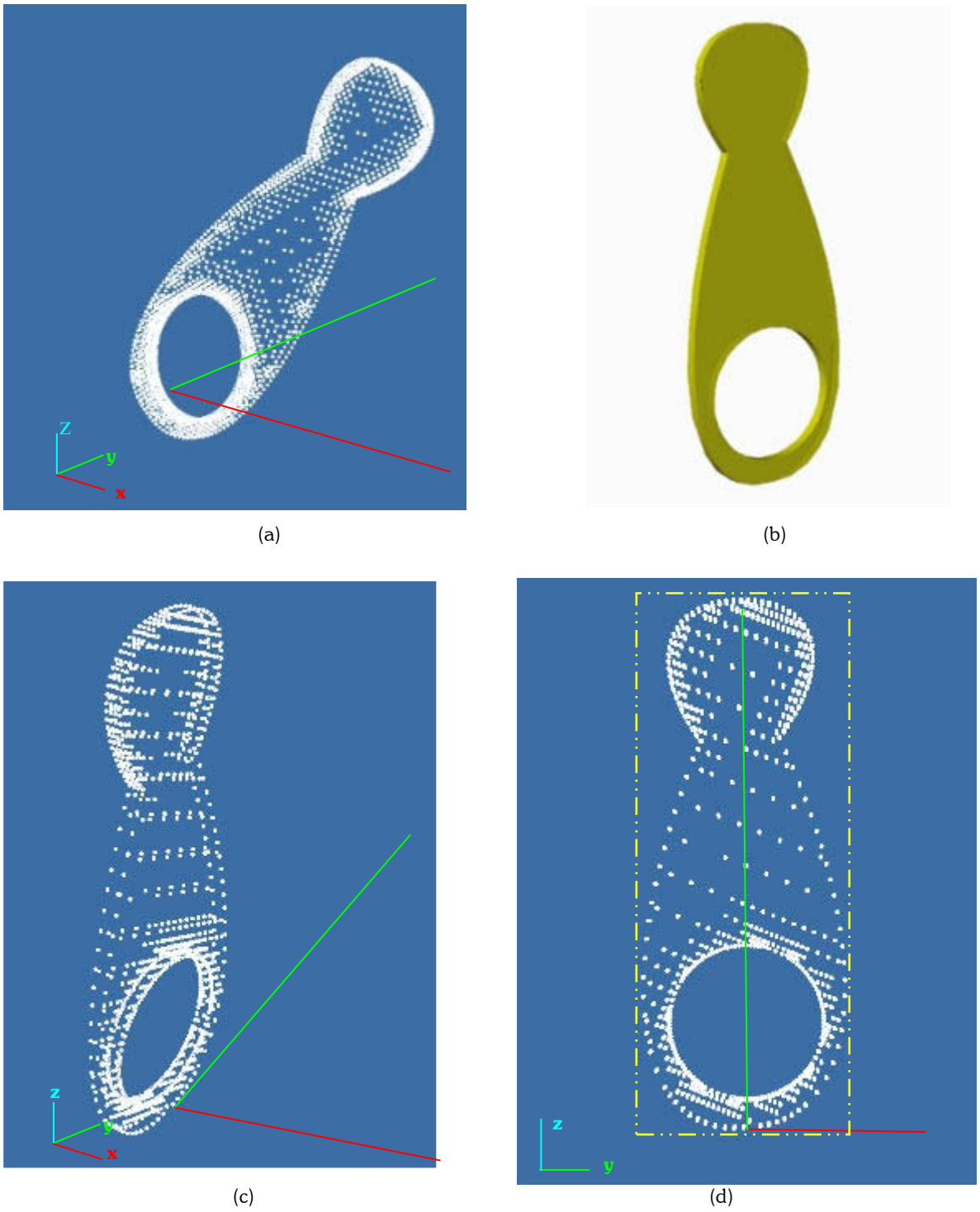

Fig. 8. (a) The point cloud before its reorientation, (b) the object whose point cloud is displayed in (a), (c) the point cloud after it has been reoriented, (d) bounding parallelepiped of the point cloud.

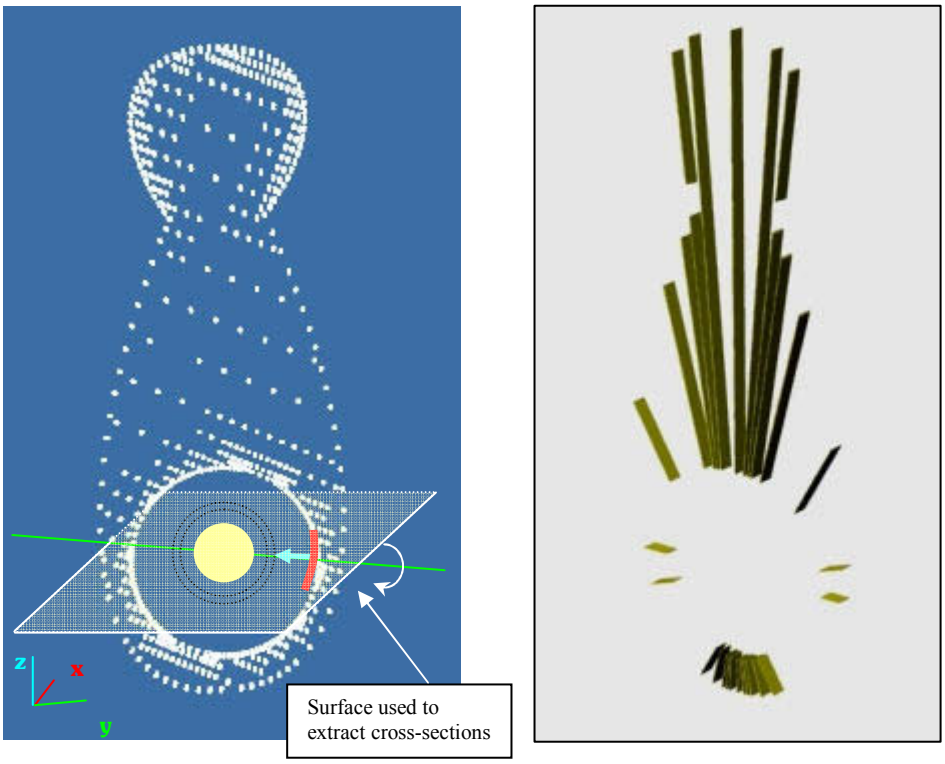

Fig. 9. (left) An example displaying how the sphere is augmented and how the planar surface cuts through the point cloud, (right) an example of the cross-sections obtained from the object by intersecting it with planar surfaces.

For each cross-section obtained by the above method, we compute the 2D medial axis. Then, we examine the cross-sections in adjacent pairs and compare the medial axis of each cross-section with the next and determine if they are similar enough. In the case where they are similar enough, we interpolate the axes by connecting corresponding points and by creating connecting surfaces. If the medial axes are very different we consider a new cross-section between the two, and after computing its medial axis, we compare it with the previous cross-sections. This process is iteratively carried out until the whole point cloud is scanned and the skeleton of the object is determined. From this information we determine the type of jewelry that is to be reconstructed.

5.2 Fitting Features and Imposing Constraints

By knowing the type of jewelry represented by the point cloud we can then continue building the model by exploiting certain features and constraints that exist. Feature detection is first performed automatically to detect an initial repertoire of feature detected by the method outlined in Section 3.2. We then go through an interactive process where feature refinement and fitting is performed iteratively.

The 3D cloud is reverse engineered using a voxel-based approach, in which each feature is represented and modeled through a voxel-based element. Voxel-based elements are used as building blocks to construct more complex shapes and designs by unioning them together to form solid CAD models. Each voxel-based element has features that make it differ from other elements, for example, a through hole, a pocket, a component forming an angle etc. By changing the parameters of the element, we can modify it to obtain an appropriate piece according to the jewelry that is being reconstructed. Each voxel-based element also has a set of constraints that refer to its morphology, dimensions and behavior, in reference to itself and to other elements. The voxel-based elements are configured and combined using Boolean operations so as to create the CAD model.

An example of such a feature-based and voxel-based approach is presented in [31] and [32], in the case of traditional pierced Byzantine jewellery. In this case, we want to reproduce pierced jewellery, which are jewellery representing designs created by combinations of through holes and carvings around the holes. We consider voxel-based elements named "poxels" as the building blocks for creating this type of jewellery. A poxel is a solid rectangular parallelepiped containing a through hole and carvings. The number of carvings and their directions are what makes each poxel differ from the next. The poxels are placed side-by-side and unioned together so as to create more complex pierced designs and shapes (Fig. 10). By combining poxels we create pierced plates, that are then manipulated and transformed to create specific forms of jewellery. For instance, in the case of a ring, the pierced plate is created and bended along the appropriate axis.

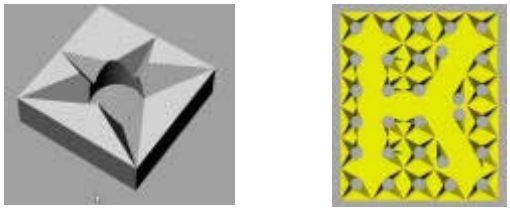

Fig. 10. (left) a pierced voxel, (right) a pierced plate displaying the letter k created by combining pierced voxels

Some constraints are embedded in the system but the user has the ability to define its own features [6]. Constraints are defined for the features found on each voxel element and for the relationships between the various "voxels" combined together to create the CAD model.

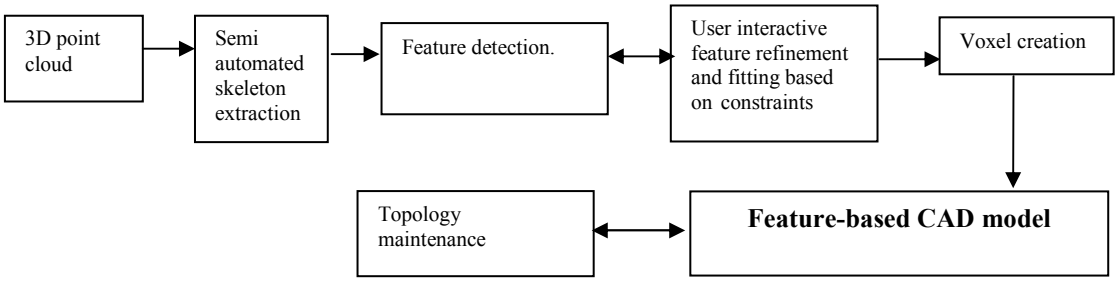

Fig. 11- A diagram of the feature-based constraint-based method for re-engineering jewelry.

We summarize our approach in Fig. 11.

6. CONCLUSIONS

In this paper we presented several aspects of a feature-based approach to re-engineering small objects. Reproducing jewelry is a challenge because of the complexity and size of the objects. In general, for a reverse engineering system to create CAD jewelry models that are both accurate and robust, certain requirements must be met. We propose an interactive system which exploits application specific knowledge to create feature and constraint based parametric models that can be prototyped or further modified and adapted to conform with user requirements.

6. REFERENCES

[1] Au, C. K. and Yuen, M. M. F., Feature-based reverse engineering of mannequin for garment design, *Computer-Aided Design*, Vol. 31, 1999, pp 751-759.

[2] Benko, P., Kos, G., Andor, L. and Martin, R. R., Constrained Fitting in Reverse Engineering. *Computer-Aided Geometric Design*, Vol. 19, 2002, pp 173-205.

[3] Benko, P., Martin, R. R. and Varady, T., Algorithms for Reverse Engineering Boundary Representation Models, *Computer-Aided Design*, Vol. 33, No. 11, 2001, pp 839-851.

[4] Benko, P. and Varady, T., Segmenting large point clouds in reverse engineering conventional engineering objects, In Proc. First Hungarian Conference on Computer Graphics and Geometry, pp 3-69, L. Szirmay-Kalos and G. Renner (eds.), 2002.

[5] Bohler, W. and Marbs, A., Investigating Laser Scanner Accuracy, White paper,Institute for Spatial Information and Surveying Technology, FH Mainz, University of Applied Sciences, Mainz, Germany

[6] Chen, X. and Hoffmann, C. M., On Editability of Feature-based Design, *Computer-Aided Design*, Vol. 27, No. 12, 1995, pp 905-914.

[7] Demarsin K., Vanderstraeten D., Volodine T. and Roose D., Detection of feature lines in a point cloud by combination of first order segmentation and graph theory, Proceedings SIAM conference on Geometric Design and Computing, Phoenix, Arizona, USA, 2005.

[8] Dey, T., Woo, H. and Zhao, W., Approximate Medial Axis for CAD Models, In Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications, Seattle, Washington, USA, 2003, pp 280 – 285.

[9] Dutta, D. and Hoffmann, C. M., A Geometric Investigation of the Skeleton of CSG Objects, Tech. Report Cer-90-10, 1990.

[10] Fisher, R. B., Applying knowledge to reverse engineering problems, *Computer-Aided Design*, Vol. 36, No. 6, 2004, pp 501-510.

[11] Fudos, I. and Hoffmann, C. M., A Graph-constructive Method to Solving systems of Geometric Constraints, *ACM Transactions of Graphics*, Vol. 16, No. 2, pp 179-216.

[12] Gao, C. H., Langbein, F. C., Marshall, A. D. and Martin R. R., Local topological beautification of reverse engineered models, *Computer-Aided Design*, Vol. 36, No. 13, 2004, pp 1337-1355.

[13] Gumhold, S., Wand X. and MacLeod R., Feature Extraction from point clouds, Proc. 10th Int. Meshing Roundtable, 2001

[14] Hoffman, C. M., Lomonosov, A., and Sitharam, M., Decomposition of Geometric Constraints Part II: new algorithms. *JSC*, Vol. 31, 2001, pp 409-428.

[15] Huang J. and Menq C., Automatic data segmentation for geometric feature extraction from unorganized 3-D coordinate points, *IEEE Transactions on Robotics and Automation*, Vol. 17, No. 3, 2001, pp 268-279.

[16] Kai, C. C., and Gay, R., CAD/CAM/CAE for ring design and manufacture, *Computer-Aided Engineering Journal*, February 1991, pp 13-24.

[17] Kanade, T. and Okutomi, M., A stereo matching algorithm with an adaptive window: theory and experiment, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, No. 9, 2004, pp 920-932.

[18] Lipschutz, M. M., *Differential Geometry*. McGraw Hill, New York 1969.

[19] Langbein, F. C., Marshall, A. D., and Martin, R. R., Choosing consistent constraints for beautification of reverse engineered geometric models, *Computer-Aided Design*, Vol. 36, 2004, pp 261-278.

[20] Langbein, F. C., Mills, B. I., Marshall A. D. and Martin R. R., Finding Approximate Shape Regularities In Reverse Engineered Solid Models Bounded By Simple Surfaces, D. C. Anderson , K. Lee (eds). Proc. 6th ACM Symp. Solid Modelling and Applications. ACM Press, 2001, pp 206-215.

[21] Lien J. and A. N.M., Simultaneous shape decomposition and skeletonization using approximate convex decomposition, 2005, Texas A&M University.

[22] Mills, B. I., Langbein, F. C., Marshall, A. D. and Martin R. R., Estimate Frequencies of Geometric Regularities for Use in Reverse Engineering of Simple Mechanical Components, Technical Report GVG 2001 – 1

[23] Mills, B. I., Langbein, F. C., Marshall, A. D. and Martin R. R., Approximate Symmetry Detection for Reverse Engineering, D. C. Anderson, K. Lee (eds). Proc. 6th ACM Symp. Solid Modelling and Applications. ACM Press, 2001, pp 241-248.

[24] Owen, J. C., Sloan P. J. and Thompson, W. B., Interactive Feature-based Reverse Engineering of Mechanical Parts, In Proc. ARPA Image Understanding Workshop, November 1994, pp 1115-1124.

[25] Roy, S. and Cox J., Stereo without epipolar lines: a maximum-flow formulation, *International Journal of Computer Vision*, Volume 34, Issue 2-3, 1999, pp 147 – 161.

[26] Sato, Y. and Honda, I., Pseudodistance measures for recognition of curved objects. *IEEE Trans. Pattern Anal. Machine Intell. PAMI-5*, Vol. 4 (July), pp 362-373.

[27] Sato, Y., Kitagawa, H. and Fujita, H., Shape measurement of curved objects using multiple slit ray projections. *IEEE Trans Pattern Anal. Machine Intell. PAMI-4*, Vol. 6 (Nov), pp 641-646.

[28] de St. Germain, H. J., Stark, S. R., Thompson, W. B. and Henderson, T. C., Constraint Optimization and Feature-based Model Construction for Reverse Engineering, In Proc. ARPA Image Understanding Workshop, 1997.

[29] de St. Germain, H. J., Reverse Engineering Utilizing Domain Specific Knowledge, Doctoral Dissertation, School of Computing, University of Utah, 2002.

[30] Solano, L. and Brunet, P., A system for constructive constraint – based modeling, In B. Falcidieno and T. Kunii, editors, Modelling in Computer Graphics, pp 61-84, Springer Verlag, 1993

[31] Stamati, V., Fudos, I., Theodoridou, S., Edipidi, C. and Avramidis, D., Using Poxels for reproducing traditional pierced Byzantine jewellery, In Computer Graphics International 2004, Crete, Greece, June 16-19.
[32] Stamati, V., and Fudos, I. A Parametric Feature-based CAD System for Reproducing Traditional Pierced Jewellery, *Computer-Aided Design*, 2004 (In press)
[33] Thompson, W. B., de St. Germain, H. J., Henderson, T. C. and Owen, J .C. Constructing High-Precision Geometric Models from Sensed Position Data, In Proc. ARPA Image Understanding Workshop, February 1996.
[34] Thompson, W. B., Owen, J. C., de St Germain, H. J., Stark, S. R. and Henderson, T.C., Feature-based reverse engineering of mechanical parts, *IEEE Transactions on Robotics and Automation*, February 1999, 15, No. 1
[35] Varady, T., Martin, R. R. and Cox, J., Reverse Engineering of Geometric Models – An Introduction, *Computer-Aided Design*, Vol. 29, No. 4, 1997, pp 255-268.
[36] Vergeest, J. S. M., Reverse Engineering for Shape Synthesis in Industrial Engineering, 26th Int. Conf. of Computers in Industrial Engineering, January 1999.
[37] Vermeer, P., Medial Axis Transform to Boundary Representation Conversion, PhD thesis, Purdue University, 1994
[38] Werghi, N., Fisher, R., Robertson, C. and Ashbrook A., Object reconstruction by incorporating geometric constraints in reverse engineering, *Computer Aided Design*, Vol. 31, 1999, pp 363-399.
[39] Zhang, Z., A Flexible New Technique for Camera Calibration, Technical Report MSR-TR-98-71, Microsoft Research, December 1998.
[40] Zhang, Z., Flexible Camera Calibration by Viewing a Plane from Unknown Orientations, In Proc. 7th International Conference on Computer Vision, Kerkyra, Greece, pp 666-673, September 1999.