Extracting Geometric Features from Clouds of Points Using Sweeping

David Yoon[1], Evrard Ohou[2], Jie Shen[3] and Huahao Shou[4]

[1]University of Michigan – Dearborn, dhyoon@mich.edu
[2]University of Michigan – Dearborn, oevfred@umd.umich.edu
[3]University of Michigan – Dearborn, shen@umich.edu
[4]University of Michigan – Dearborn, huahao@umd.umich.edu

ABSTRACT

Reverse Engineering involves data acquisition, CAD model building, and manufacturing of the parts based on the obtained models. The algorithms are presented to extract basic geometric features for mechanical parts from Clouds of Points (COP) data using sweeping techniques. Geometric features are classified into two broad categories: Basic and Advanced. The former includes polyhedra and swept models. The swept models fall into three subcategories: translational, rotational, and general. The advanced features are defined on basic features in terms of Boolean operations. This classification is based on global characteristics of mechanical parts. The **F**eature-based **R**everse **E**ngineering **S**ystem (FRES) is under development based on this theory and should be, when completed, capable of inferring the shapes of missing or worn-out parts and refurbishing legacy parts which were manufactured many decades ago and have no CAD documents nor spare parts.

1. INTRODUCTION

With advances in scanning and measuring technologies, there are a number of scanning devices available on the market, for example, CMM (coordinate measuring machine), laser sensors, moire sensors, ultrasound sensors, etc, which can scan 3D objects and generate COP data. It is not uncommon that a cloud consists of millions of 3D points. Varady et al [10] succinctly state the need for Reverse Engineering in CAD/CAM. Some mechanical parts were designed many decades ago and have no CAD database nor design documents available. Often times the original vendors are no longer in business to provide them. The only way to maintain these legacy systems is to extract CAD data from existing parts or to infer the shapes of incomplete parts due to breakage or wear. Extracting geometric features from COP data is a challenging process. In the above mentioned paper, the authors outline the procedure for reverse engineering as follows: data acquisition, preprocessing, segmentation and surface fitting and CAD model creation. They emphasize the use of global characteristics for each geometric object for the creation of its CAD model. Benko et al [2] formulate the extraction of 2-D and 3-D primitives such as lines, circles, sphere, cylinders, cones, and torus as an optimization problem. Kos et al [7] present algorithms to extract blends and fillets using the idea of a rolling ball. Varady and Rockwood [19] address the vertex blending problem. Benko and Varady [1] fit translational and rotational surfaces into COP data. Benko and Varady [3] present algorithms to segment compound objects into 3-D primitives. Mills et al [11] search symmetry in an object. With man-made entities, the existence of symmetry resolves a large number of cases. Thompson et al [17] have developed a feature-based reverse engineering system based on machining features. Hoppe [5], on the other hand, approximates the surfaces underlying COP data in terms of triangular meshes. Edelsbrunner and Mucke [4] introduce the concept of an alpha shape generalizing a polyhedral representation of an object.

In this paper we present algorithms for extracting geometric features from COP data using their global characteristics specified by sweeping parameters. The novel characteristic of our approach is that objects are considered 3-D manifolds rather than 2-D, and hence they have volumes. A feature is defined as "a geometric entity which is *inferable* from the geometry of the part and which is useful in engineering activities, e.g. design, analysis, and manufacturing of the part" [6]. As the definition indicates, the concept of a feature depends on applications. Most research activities have been on machining features. In this paper the emphasis is on *design* or *form features* of mechanical parts. Ohou [12] classified the form features of most existing solid modeling systems such as I-DEAS, UNIGRAPHICS, CATIA, Pro/Engineer, etc for shape inference as follows:

I.   Basic Features
    (i)   Polyhedra
    (ii)   Swept models
        (a) translational sweep (extrusion)
        (b) rotational sweep (revolute)
        (c) general sweep
II. Advanced Features
    Advanced features are obtained by taking Boolean operations on the basic features.

2. SWEEPING

Sweeping is the most fundamental tool in our approach to shape inference and refers to the moving of a generator, which may be a curve, surface, or solid. A simple shape may be obtained by one of the three sweeping techniques: translational, rotational, and general. Translational sweep refers to the moving of a generator along a spine curve or a trajectory. A typical example of translational sweep is the extrusion operation in solid modeling. A part obtained by a translational sweep is completely specified by a generator and a trajectory as illustrated in Figure 1. Rotational sweep may be illustrated in terms of the urn in Figure 2. It is designed by rotating a generator curve around the rotation axis. A general sweep may be specified in terms of a series of cross-section curves along a spine curve (Figure 3).

2.1. Translational Sweeping

Translational sweep refers to the moving of a generator, which may be *open* or *closed*, along a trajectory or spine curve. The generator at a particular point on the curve may be obtained by projecting points against a slice plane. Let $P(v)$ be the path curve. Then the local coordinate at the point p_i may be represented as $(p_{i,} < \mathbf{u}, \mathbf{v}, \mathbf{w} >)$ where

$$\mathbf{u} = \frac{P'}{|P'|}, \quad \mathbf{v} = \frac{P'' x P'}{|P'' x P'|}, \quad \mathbf{w} = \mathbf{u} \text{ x } \mathbf{v}, \text{ where P' and P'' are the first and second derivatives of } P(v) \text{ evaluated at}$$

point p_i.   (1)

In order to simplify computation, the generator may be represented as a profile curve which may be obtained by projecting the generator on the v-w plane which is perpendicular to the path curve. Following Piegl and Tiller [14], the surface obtained by moving a profile curve along a trajectory was given by $\mathbf{S}(u,v) = \mathbf{T}(v) + M(v)\mathbf{C}(u)$, where $\mathbf{T}(v)$ is the trajectory, $\mathbf{C}(u)$ represents the profile curve, $M(v)$ is a linear transformation, and '+' represents Boolean sum. In the current translational sweeping we will consider the case in which the profile curve does not go through shape change as it is swept and $M()$ is an identity matrix. Both the trajectory and profile curve are represented as NURBS curves.

There are two cases to consider: 1.The profile curve is open. In this case, translational sweeping produces an open surface. 2. When the profile curve is closed, translational sweeping produces a closed surface such as cylinder or block.

2.2. Rotational Sweeping

A rotational sweep is achieved by rotating a generator about an axis of rotation. Figure 2 illustrates a surface obtained by sweeping a generator curve around the z-axis. The first figure is a generator (profile) curve on the XZ plane, the second one the tessellation of the surface swept by the generator curve, and the last one is the shaded image of the surface. As one can see an object resulting from a rotational sweep is symmetric with respect to the axis of rotation.

Using this fact, a series of cross-section curves specifies an object of rotation. By interpolating them, one can obtain the surface of revolution.
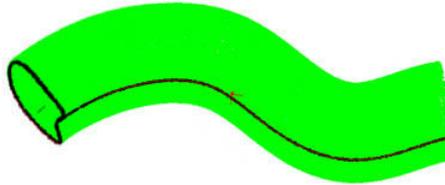

Fig. 1: A profile curve and a trajectory.

To be specific, a surface obtained by rotating a generator curve g (u) = (g_1 (u), 0 g_3 (u)) in the XZ plane about the Z-axis may be represented as follows:

$$S(u,v) = R_z(v) g(u)$$

$$= \begin{bmatrix} \cos v & -\sin v & 0 \\ \sin v & \cos v & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} g1 \ (u) \\ 0 \\ g3 \ (u) \end{bmatrix} \qquad (2)$$

Fig. 2: Rotational Sweep: A profile curve, the surface of revolution, the shaded image [13].

2.3. General Sweeping

In both translational and rotational sweeping, the generators do not go through shape change as they are swept. However, in general sweeping the generator changes its shape as it sweeps. Piegl and Tiller's [13] definition of a swept surface, $\mathbf{S}(u,v) = \mathbf{T}(v) + M(v)\mathbf{C}(u)$, where M(v) is a linear transformation, allows the shape change of the profile curve. The resulting surface may be defined as a homeomorphic image of the unit square of grids, i.e.,

$$S: I \times I \to R^3, \text{ where } I=[0,1]. \qquad (3)$$

For convenience the unit square is assumed subdivided as in Figure 3. The set of curves along one direction are called parallels and the others meridians. The cross-section curves are homeomorphic images of parallels and one of the meridians may be used as the profile curve. A surface may be obtained by interpolating cross-section curves along meridians. In some cases, it is simpler to sweep cross-section curves around the spine curve which goes through the centers of the cross-section curves.

The set of cross-section curves show the shape change of the generator curve and form a homotopy [16].

Let f: X $\to$ Y and g: X$\to$ Y be two continuous maps between topological spaces X and Y. These maps are called homotopic if there exists a family Φ_t, $0 \le t \le 1$, of continuous maps

$$\Phi_t : X \to Y \qquad (4)$$

Continuous with respect to t and x ε X simultaneously and satisfying $\Phi_0 = f$ and $\Phi_1 = g$.
It can also be regarded as a continuous map Φ: X x I $\to$ Y, where I = [0,1]. When $\Phi_t(x) = (1-t)f(x) + tg(x)$, it is called straight line homotopy.
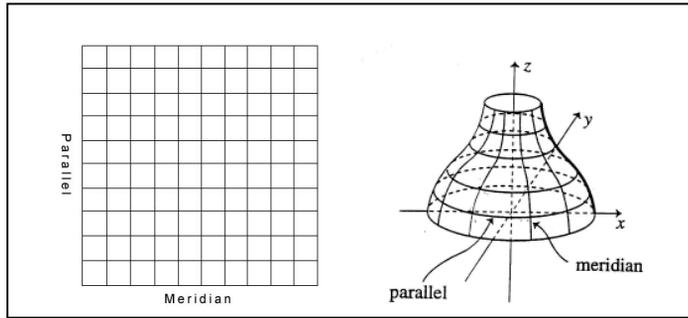
Fig. 3: The unit square and a homeomorphic image.

DEFINITION 1 [15]:

Two spaces X and Y are called homotopically equivalent if there are continuous maps f: X → Y and g : X→ Y such that the composition fg: X→ Y is homotopic to the identity map of Y, and the composition gf: X → X is homotopic to the identity map of X.

The cross-section curves to be obtained from the intersections of the free-form surface and slice-planes in Figure 4.c may illustrate the concept of a homotopy. The spine curve may be normalized to the unit interval [0,1]. The cross-section curves $\Phi_0(v)_{,....,}\Phi_1(v)$, where $v \varepsilon$ [0,1], are a subset of the family of continuous function, and hence form a homotopy.

Summing up the sweeping techniques presented in this section, a transitionally swept object may be specified in terms of a profile and a spine curve (Figure 4. a), a rotationally swept object may be specified in terms of a profile curve and its axis of rotation (Figure 4. b), and finally a general swept object may be specified in terms of a set of cross-sectional curves and a spine curve (Figure 4. c).
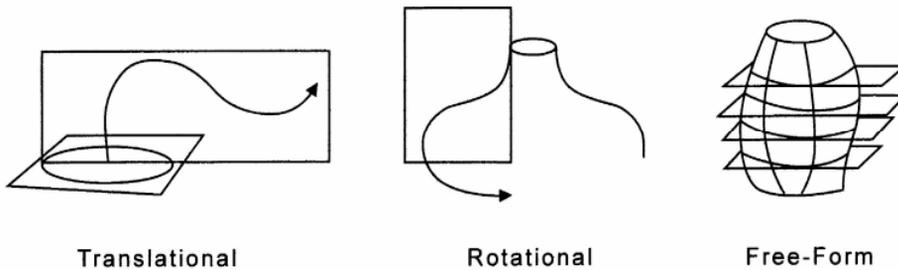

Fig. 4: (a) A translationally swept part, (b) a rotationally swept part, (c) a generally swept part.

3. PREPROCESSING OF COP DATA

Extracting geometric features from COP data is by no means a trivial task. The object frame comprising the center of the object and the object coordinates will be of great help in the extraction of the necessary information from COP data. Hence in the next section the computation of the object center, object coordinates and the minimal bounding box is illustrated.

3.1. Object Center, Object Coordinates, and Bounding Box

Let $\{p_i = 1,,.,n\}$ be COP points from the part. Define the center o of the object as:

$$o = 1/n \Sigma p_i \tag{5}$$

The covariance matrix is given by

$$C = 1/n \Sigma (p_i - o)^T (p_i - o) \tag{6}$$

The eigenvalues of the covariance C is calculated as follows:

$$\text{Det } (C - \lambda I) = 0 \tag{7}$$

Eigenvectors X should be singular and computed as follows:

$$CX = \lambda X \quad \text{or} \quad (C - \lambda I)X = 0. \tag{8}$$

Arranging the eigenvectors **r**, **s**, and **t** in descending order of magnitude, one obtains the object coordinate system $\{o, < \mathbf{r}, \mathbf{s}, \mathbf{t} >\}$, where **r** being the largest. The object coordinate system $\{o, < \mathbf{r}, \mathbf{s}, \mathbf{t} >\}$ is also known as a *frame* for the object.

The dimensions of the bounding box of a part are computed as follows:

$$a = \tfrac{1}{2} \{< \mathbf{r}, -\min \{ p_i \bullet \mathbf{r}\} > + <-\mathbf{r}, \max \{ p_i \bullet \mathbf{r}\} >\}$$
$$b = \tfrac{1}{2} \{< \mathbf{s}, -\min \{ p_i \bullet \mathbf{s}\} > + <-\mathbf{s}, \max \{ p_i \bullet \mathbf{s} \} >\}$$
$$c = \tfrac{1}{2}\{< \mathbf{t}, -\min \{ p_i \bullet \mathbf{t} > + <-\mathbf{t},\max\{ p_i \bullet \mathbf{t}\} >\} \tag{9}$$

The bounding box bd may be expressed as follows: bd $= a\mathbf{r} + b\mathbf{s} + c\mathbf{t}.$

The bounding box obtained this way is minimal in the sense that any other bounding box will be larger than this one. Figure 5 is an image of a set of COP data collected from a block using a Minolta Vivid 910. The computed object frame $\{o, < \mathbf{r}, \mathbf{s}, \mathbf{t} >\}$ and the bounding box are illustrated.
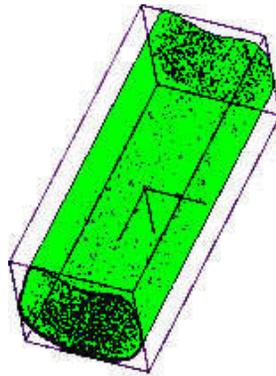

Fig. 5: The COP data from a block, the object frame and its bounding box.

4. SLICE PLANE SELECTION

The process of selecting a slice plane is not trivial. Given a set of COP data representing the 3-dimension object, we have calculated its center and local coordinates in 3-D space. The minimal bounding box has been computed using the eigenvalues and corresponding eigenvectors of the covariance matrix over all of the COP data representing the object. The minimal bounding box is such that it is the smallest bounding box that encloses the object without intersecting with it. Both the object and its minimal bounding box share the same local coordinate system. In fact the slice planes are derived from projecting each side of the minimal bounding box onto the center of the object.

The slice planes may be derived from the local coordinate system of the object, as calculated earlier. There are three principal slice planes. The first slice plane contains the **rs**-plane, the second one contains the **rt**-plane, and the third one contains the **st**-plane. One can choose one from the three principal slice planes to define the trajectory such that the remaining two are orthogonal to it and their normal vectors follow the right-handed rule orientation respectively. The case of the rotational sweep does not require a trajectory plane. The curve created by intersecting the slice plane and the scanned part is called the profile curve. The curve that is the intersection of the trajectory plane and the scanned part is called either the trajectory curve or the spine curve. The difference that we make between a trajectory curve and a spine curve is that a trajectory curve does not usually go through the center of the scanned part, whereas a spine curve is a curve that goes through the center of the object. Both trajectory and spine curves describe the path taken by the slice plane in the technique presented in Ohou [12].

The slice planes may be derived from the bounding box and the local coordinate system of the object, as calculated earlier. The following recursive algorithm computes the slice planes (Figure 6):
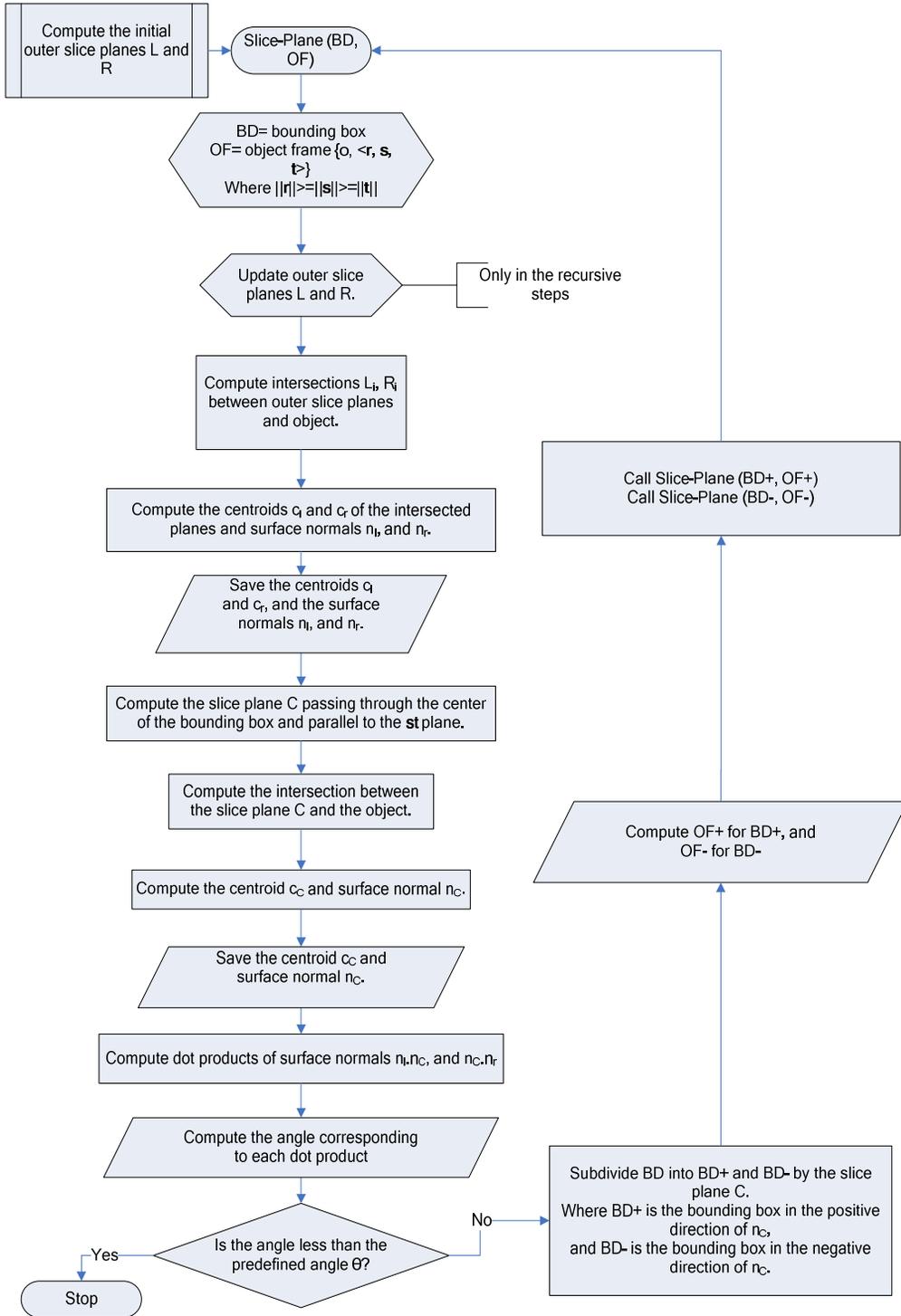
Fig. 6: The Slice Plane Algorithm.

5. PROJECTING 3D POINTS TO THE SLICE PLANE

Once a slice plane is obtained, the next step is to project a subset of COP data onto the plane. The projected points, now in 2D space, are to be thinned out through a process called *moving least square approximation*. The thinned points are then gathered and sorted into an ordered set of control points that are interpolated to form either a closed or open NURBS curve.
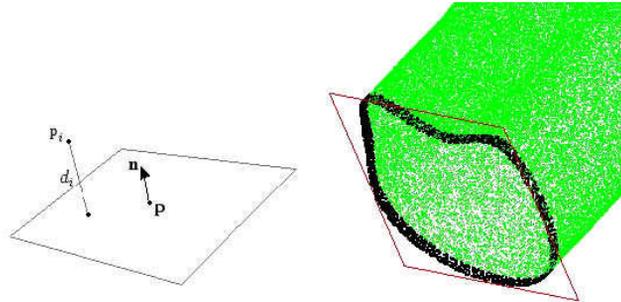

Fig. 7: (a) Representation of a plane   (b) Selected sample COP data.

The slice plane is represented by its point-normal form $\{\mathbf{n}, p\}$, where $\mathbf{n}$ is the plane unit normal vector, and p is a point on the plane (Figure 7.a). The slice plane approximates the position and orientation of an original plane containing the desired curve (Figure 7.b). We sample a subset of the COP data points to project onto the slice plane by selecting points from the COP data that are within a tolerance, *t*, the distance from the slice plane. The distance function used is not the usual Euclidean point-to-plane distance. The distance function is called the *signed distance*, d_i, from some point p_i to the slice plane

$$d_i = \mathbf{n} \cdot p_i + d, \text{ where } d = -\mathbf{n} \cdot p \tag{10}$$

The sign of the distance is positive if the point is in the direction of the normal vector $\mathbf{n}$ and is negative if the point lies in the opposite direction of the normal vector $\mathbf{n}$ from the slice plane. The sampled COP data points $\{ p_i\}$ are such that $|d_i| \leq t$. They are translated to the slice plane and form a new set of points $\{ p_i ' \}$ such that $p_i ' = p_i - d_i \cdot \mathbf{n}$. The slice plane, together with the set $\{ p_i ' \}$, is moved to the world coordinate origin and is rotated so that the slice plane's normal vector coincides with the world coordinate *z*-axis, transforming the translated 3D points into 2D points in the slice plane. The transformed 2D point cloud forms an unorganized band of points around the desired curve. We need now to select a subset of points to create an ordered organized set that approximates the desired curve.
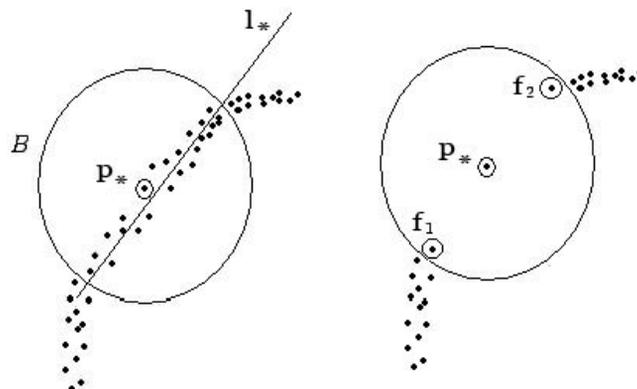

Fig. 8: Eliminating neighboring points within a radius.

6. OBTAINING CURVES FROM COP DATA

Obtaining profile and trajectory (or spine) curves is a crucial step in the inference of shapes. The algorithms presented below will play the major role in the reduction of millions of COP data points to a couple of curves.

6.1. Thinning and Smoothing of COP Data

The banded 2D points approximate the desired curve. This approximation requires the selection of a target point that better fits the desired curve within the band, when the latter is thin enough. In order to assure that we have a thin enough band of COP data approximating the desired curve, we utilize a method called the *moving least squares* method by Levin [9] to thin out the point cloud. McLain [18] developed the moving least squares method, and Lee [8] presented algorithms using the moving least squares method applied to COP data with varying thickness.

The theory of the method resides in the statistics of regressions. COP data are thinned and smoothed by projecting each point in the data onto its local regression curve. That is for each point p_* in {p_i}, there is a local quadratic regression curve, $q_* : y = a_0 + a_1 x + a_2 x^2$. We compute the local regression curve by minimizing

$$d_q = \sum_{i=1}^{N} \left(y_i - \left(a_0 + a_1 x_i + a_2 x_i^2 \right) \right)^2 w_i \tag{11}$$

where d_q is the square distance in the y direction from a neighboring point $p_i = (x_i, y_i)$ to the local quadratic regression curve, and w_i is a nonnegative weight for each p_i. In order to make the approximation local to p_*, a weighting function that penalizes points whose distance from p_* is larger than some chosen r

$$w_i = \begin{cases} 2\dfrac{h^3}{r^3} - 3\dfrac{h^2}{r^2} + 1 & \text{if } h \le r \\ 0 & \text{if } h > r \end{cases} \quad , \text{where } h = \left\| p_i - p_* \right\|^2 \tag{12}$$

The projection of a point onto its local regression curve can be simplified by computing a local regression line $l_* : y = a_0 + a_1 x$, where a_0 and a_1 are values that minimize the squared y-distance to l_* from points in {p_i} near p_*. All the points within the ball B or radius r are eliminated except p_*, f$_1$, and f$_2$ (Figure 8). We apply a transformation, M, to all of the points in {p_i} that moves p_* to the origin, of the transformed slice plane, and rotates l_* so that it is parallel to the x-axis, producing a new set $\left\{ \hat{p}_i = (\hat{x}_i, \hat{y}_i), i = 1, ..., N \right\}$. A quadratic regression curve $\hat{q}_*$ is also computed for $\left\{ \hat{p}_i \right\}$ by minimizing the following equation

$$d_q = \sum_{i=1}^{N} \left(\hat{y}_i - \left(a_0 + a_1 \hat{x}_i + a_2 \hat{x}_i^2 \right) \right)^2 w_i \tag{13}$$

The projection of the points in $\left\{ \hat{p}_i \right\}$ onto $\hat{q}_*$ is simply (0,a$_0$), and the projection of p_* onto q_* is found by applying the inverse transformation M^{-1} of M to (0,a$_0$).

One or more successive applications of the moving least square method to thinning and smoothing a banded COP data in a 2D plane might be required to get a desired thin enough band that can be approximated by a curve.

6.2. Points Selection for Curve Approximation

We adopt the technique as presented by Piegl and Tiller [15] to the selection of points approximating the desired curve. For a point p_* of a 2D point cloud $\left\{ p_i \right\}, 0 \le i \le N$, we need to find a subset of $\left\{ p_i \right\}$ such that $\left\| p_i - p_* \right\| < r$, for some chosen radius r. A rudimentary search by comparison of the Euclidean distance for all the

points in the set $\{p_i\}$ has a time complexity of $O\left(N^2\right)$. Piegl and Tiller [20] find the k nearest neighborhood points of p_* that avoids the use of sophisticated preprocessing techniques as that of the Voronoi diagram of point, and has a linear time complexity of $O\left(N\right)$. The technique uses a grid based data structure. A slight modification is applied to find neighboring points within a chosen radius.

The grid data structure is set up over the enclosing rectangular space $\left[x_l, x_r\right]\times\left[y_b, y_t\right]$ of the 2D point cloud, where subscripts l,r,b, and t refer to left, right, bottom and top, respectively. We estimate the size of the grid by

$$size = \alpha\sqrt{\frac{\left(x_r - x_l\right)\left(y_t - y_b\right)}{N}} \tag{14}$$

where $\alpha = 1.0$ originally and is changed to adjust the grid size. The resolution of the grid in both x and y directions is given by

$$x_{res} = \left\lfloor\frac{x_r - x_l}{size}\right\rfloor \qquad y_{res} = \left\lfloor\frac{y_t - y_b}{size}\right\rfloor,\text{ where } x_{res}\text{ and }y_{res}\text{ floor values, i.e. integers.} \tag{15}$$

A cell structure is created in row-major using a 2-dimensional array of a linked list as follows:

$$cell\left[j\right]\left[k\right] \quad j = 0,\ldots, y_{res} - 1 \quad k = 0,\ldots, x_{res} - 1 \tag{16}$$

The index of each point in $\{p_i\}$ is placed in a cell. Indexes of points falling in the same cell are pushed onto the end of the list. The appropriate cell is found given x or y values as follows:

$$cell_j\left(y\right) = \begin{cases} y_{res} - 1 & \text{if } y = y_b \\ \left\lfloor\dfrac{y_t - y}{size}\right\rfloor & \text{otherwise} \end{cases} \tag{17}$$

$$cell_k\left(x\right) = \begin{cases} x_{res} - 1 & \text{if } x = x_r \\ \left\lfloor\dfrac{x - x_l}{size}\right\rfloor & \text{otherwise} \end{cases}$$

We search for neighboring points of p_* by starting with the cell that contains p_*. The distances from p_* to each point within the same cell are checked. We denote by d_{sh} the maximum radius searched within the cell, which is the shortest distance from p_* to a cell wall. If d_{sh} is less than the chosen radius r, we extend the search to the surrounding cells. Each extension to a new surrounding cell is referred to as a layer, which is incremented each time a new layer is (Figure 9). The search proceeds layer by layer until a layer is reached where $d_{sh} + layer\times size > r$. Figure 10 illustrates a set of thinned points.

7. BASIC FEATURES
Most mechanical parts are designed using solid modeling systems such as I-DEAS, UNIGRAPHICS, CATIA, Pro/Engineer, etc, which allow designers to design new products in terms of primitive shapes. For the inference of the shapes of missing components, the primitive shapes may be classified into two major categories as follows:
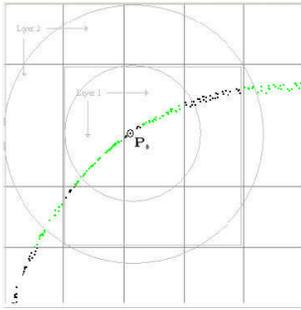
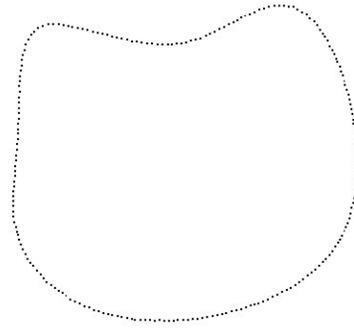Fig. 9: Grid over points and maximum searchable radius within layers.

Fig. 10: Thinned and ordered points approximating desired curve.

*Polygonal Models*: tetrahedron, cube, dodecahedron
Swept Models:
- Translational : cylinder family
- Rotational: cone, sphere, torus, ellipsoid,
- General : hyperbolic paraboloid, helicoid:

An Example: Based on the theory presented above, FRES is under development. It receives, as input, a set of COP data from an incomplete physical part due to breakage or wear, infers the shape of the broken-off piece, and remakes the incomplete part. As an example, we will demonstrate the major steps of the reverse engineering process using a broken screw (Figure 11.a). The basic assumption is that there is no CAD data nor documentation available. The shape of the broken-off piece must be inferred and remanufactured. The part is scanned using the Minolta Vivid 910 laser scanner and a set of COP data is obtained (Figure 11.b). The object frame consisting of the center of the object, the coordinate system and the minimal bounding box is computed (Figure 11.b).
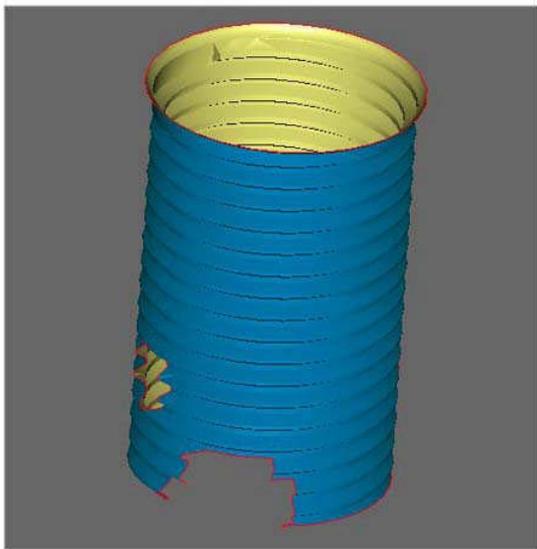
(a)
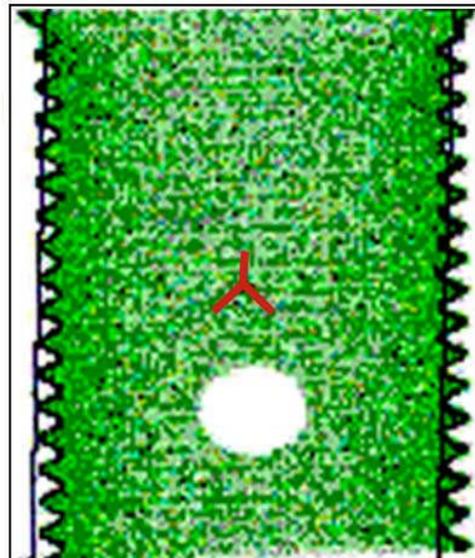
(b)

Fig. 11: (a) A broken screw.   (b) The COP data, the object frame, and the bounding box.
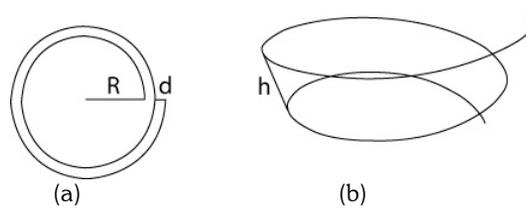
Fig. 12: (a) The radius R and distance d in a projected plane (b) The height h in a 3D space [14].

The next step involves obtaining sample points for the screw path and interpolating them in terms of NURBS curves. The path of the screw motions is given by

$$
P(\Theta) = \begin{pmatrix} (R + \dfrac{d}{2\pi})\cos\vartheta \\ (R + \dfrac{d}{2\pi})\sin\vartheta \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \dfrac{pt}{2\pi}\vartheta \end{pmatrix} \tag{18}
$$

where R= the initial radius, d= the distance between two points reached by the curve before and after a 2л turn in the projected plane (Figure 12. a.), pt = the pitch = h/(2 л), h = the distance between two points reached by the curve before and after a 2л turn in the 3D space [14]. Figure 13.a. shows the sample points for the screw path obtained by using equation 18 and Figure 13.b. is the NUBRS representation of the screw path.

(a)   (b)
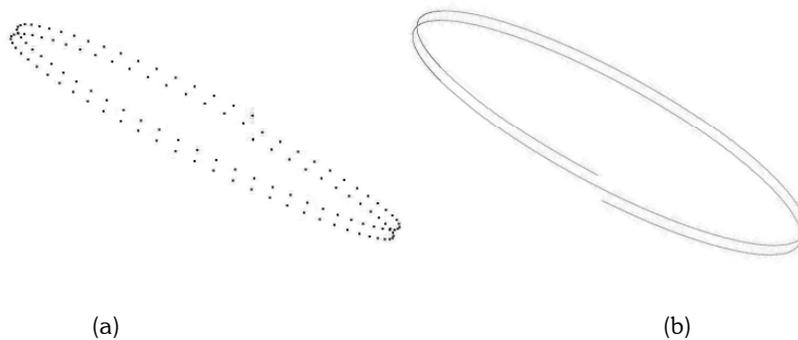
Fig. 13: (a) The sample points for the screw path. (b) The NURBS interpolated path.

Similarly, Figure 14.a. illustrates the sample points of the profile of the cutter which cuts out the groove along the screw path. For this, a scan plane is set to pass through the center of the screw and contain one of the local coordinate axes. The boundary curve of the screw is projected onto the scan plane and interpolated in terms of NURBS as discussed in sections 5 and 6. Figure 14 (a) illustrates the sample points of the profile curve of the cutter and Figure 14.b. shows its NURBS interpolation.

8. CONCLUSION
We have introduced sweeping as a way to specify objects. In translational sweeping an object is specified in terms of a spine curve (or trajectory) and a profile curve, in rotational sweeping in terms of the axis of rotation and a profile curve, and in general sweeping in terms of a spine curve and a set of cross-section curves. As illustrated in the above example, the technique is so robust that it enables one to infer the shape of a missing component from the remaining part. In addition to shape inference, it may easily be employed for shape reconstruction and matching. In the follow-

up paper, we plan to unify the swept models into the homotopy model and use it to segment compound objects into primitive shapes.

(a)   (b)
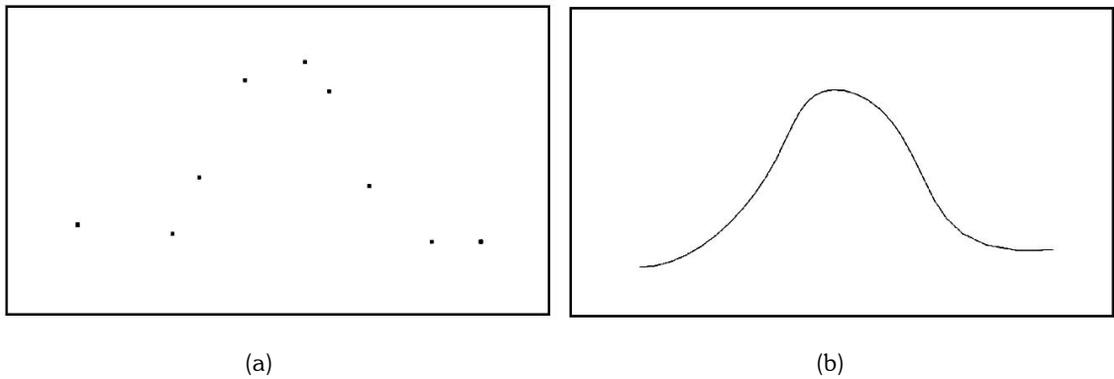
Fig. 14: (a) the Sample points for the profile of the Cutter.   (b) The NURBS interpolated profile curve.

By sweeping the profile curve (Figure 14.b.) along the path (Figure 13.b.), the screw is generated (Figure 15.a). The shape of the broken-off screw piece (Figure 15.b) is regenerated by taking Boolean difference between the regenerated screw (Figure15.a.) and the damaged screw (Figure 11.a.). Further details may be found in [21].

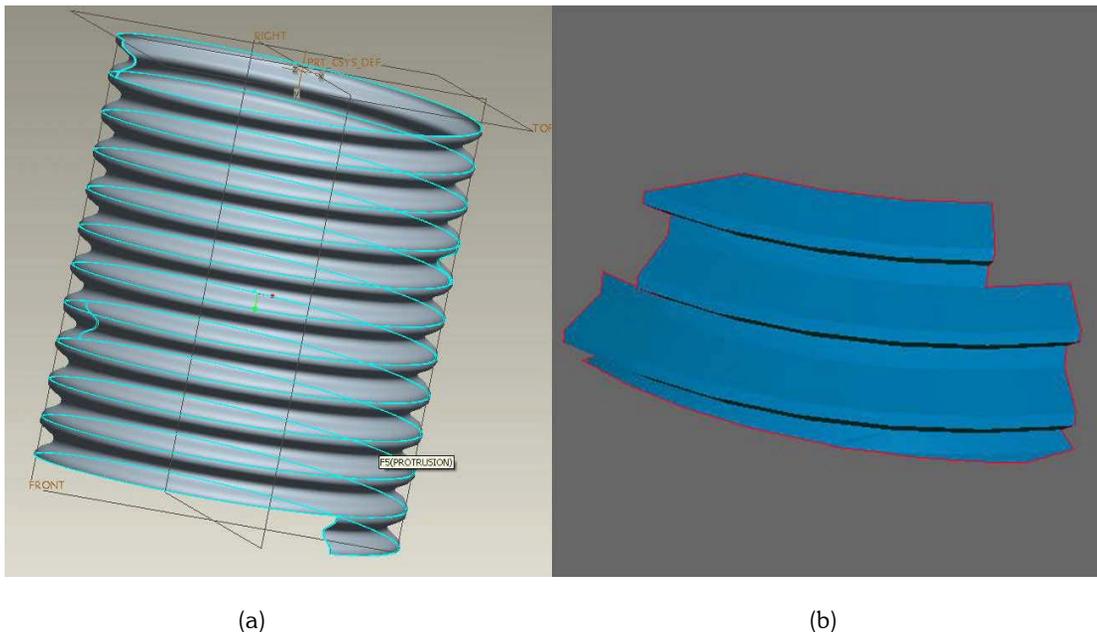(a)   (b)

Fig. 15: (a) The original screw is obtained by sweeping the profile curve along the spine curve. (b) The shaded image of the broken-off piece obtained by taking Boolean difference between the regenerated shape in (Figure 15.a) and the broken screw (Fig. 11. a).

9. REFERENCES

[1]   Benko, P.; Varady T.: Best Fit Translational and rotational surfaces for reverse engineering shapes, In: Cipolla, R.; Martin, R. R., editors, The Mathematics of Surfaces IX, New York, Springer, 2000, 71-81.

[2]   Benko, P.; Kos G.; Varady T.; Ando L.; Martin R.: Constrained fitting in reverse engineering, Computer Aided Geometric Design, 19, 2002, 173-205.

[3]   Benko, P.; Varady, T.: Segmentation methods for smooth point regions of conventional engineering objects, Computer-Aided Design, 36, 2004, 511-523.

[4]   Edelsbrummer, H.; Mucke, E. P.: Three-dimensional alpha Shapes, ACM Trans. on Graphics, 13, 1994, 43-72.

[5]   Hoppe, H.: Surface Reconstruction from Unorganized Points, University of Washington, PhD Thesis, 1994.

[6]   Kim, Y. S.: Volumetric Feature Recognition Using Convex Decomposition, In: Shah J.; Mantyla, M., Nau, D. S., editors, Advances in Feature Based Manufacturing, New York, Elsevier, 1994.

[7]   Kos, G.; Martin, R. R.; Varady, T.: Methods to recover constraint radius rolling ball blends in reverse engineering, Computer Aided Geometric Design, 17, 2000, 127-160.

[8]   Lee, I. K.: Curve Reconstruction from Unorganized Points, Computer-Aided Geometric Design, 17, 2000, 161-177.

[9]   Levin, D.: The Approximation Power of Moving Least Squares, Mathematics of Computation, 67, 1998, 1517-1531.

[10]   McLain, D.: Two Dimensional Interpolation from Range Data, The Computer Journal, 19, 1976, 178-181.

[11]   Mills, B. I.; Langbein, F. G.; Marshall, A. D,; Martin, R. R.: Approximate Symmetry Detection for Reverse Engineering. Solid Modeling 01, Ann Arbor, Michigan, USA, Copyright, ACM 2001, 1-581113-3669/01/06

[12]   Ohou, E. F.: Design and Implementation of a Feature-based Reverse Engineering System, University of Michigan – Dearborn, MS-Thesis, 2005.

[13]   Paoluzzi, A.: Geometric Programming for Computer-Aided Design, West Sussex, England, John Wiley & Sons, 2003.

[14]   Piegl, L.; Tiller, W.: The NURBS Book, New York, Springer, 1997.

[15]   Piegl, L.; Tiller, W.: Algorithms to Finding all k nearest Neighbors, Computer-Aided Design, 34, 2000, 593-603.

[16]   Shinagawa, Y.; Kunii,T. L.: The Homotopy Model: A Generalized Model for smooth surface generation from cross sectional data, The Visual Computer, 7, 1991, 72-86.

[17]   Thompson, W. B.; Owen, J. C.; de St. Germain, H. J.; Stark, S. R,; Henderson, T. C.: Feature-Based Reverse Engineering, IEEE Trans on Robotics and Automation, 15(1), 1999, 1-10.

[18]   Varady, T.; Martin, R. R.; Cox, J.: Reverse Engineering of Geometric Models: An Introduction, Computer-Aided Design, 29, 1997, 255-268.

[19]   Varady, T.; Rockwood, A.: Geometric Construction for setback vertex blending. Computer-Aided Design, 29, 1997, 413-425.

[20]   Yoon, D.; Shen, J.; Mohanty, P.; Kantz, J.: Application of Sweeping Techniques to Reverse Engineering, International Journal of Modelling and Simulation, 25, 2005, 278-284.

[21]   Yoon, D.; Ohou, E.; Mohanty, P.: Capturing Screw Motions from COP Data. Proc of 6th Int. Conf. Computer-Aided Industrial Design and Conceptual Design, Netherlands, May 30-June 1, 2005, 1-4.