



Identification of Ruled Surfaces in a Model Reconstruction Step

Wissam T. Joumaa¹, Ramy F. Harik² and William J. E. Derigent³

¹Lebanese American University, wissam.joumaa@lau.edu.lb

²Lebanese American University, ramy.harik@lau.edu.lb

³Research Center for Automatic Control of Nancy (CRAN), william.derigent@cran.uhp-nancy.fr

ABSTRACT

Reverse engineering is integrated in today's industry on physical models, as a solution to remove imperfections, enhance functionalities, generate a symmetrical part, or simply duplication. Weaknesses and limitations of the conventional model reconstruction methodologies, whether manual or automated, arouse when it comes to recognition and reconstruction of an extensive, yet very important type of surfaces: ruled surfaces. Our focus in this work is on the link between the numerical model resulting from digitization, and the final CAD (Computer Aided Design) model. We will extend our work by giving a perspective on the manufacturability of ruled surfaces. Our objective is to automate ruled surfaces recognition during the model reconstruction step, and enrich them with geometrical information that will simplify their manufacturability.

Keywords: reverse engineering, filtering, surface reconstruction, ruled surfaces.

DOI: 10.3722/cadaps.2009.461-470

1. INTRODUCTION

Reverse engineering is used as an enhancement solution for physical and virtual models, designed from given specifications. It is used for functionality and aesthetics improvement, imperfections removal, and other applications. It minimizes time, and enables automation. This chain starts with digitization, followed by model reconstruction to obtain a final CAD model faithful to the original product. This model could then be submitted to CAPP (Computer Aided Process Planning), CAM (Computer Aided Manufacturing), simulation of machine dynamics, prototyping, quality control and finally manufacturing. However, the clouds of points obtained from digitization have to pass filtering and noise removal, simple shape recognition, surface fitting, and beautification before proceeding with the remaining steps of the chain. Although some CAD software are capable of recognizing basic shapes, they are limited when it comes to dealing with complex and non-regular surfaces. As a result, operator interference is required to perform an optimal surface fitting. Our work is concentrated on the transition phase between the numerical model resulting from digitization, and the final CAD model. Our objective is to automate the recognition and reconstruction of ruled surfaces, and enrich them with the necessary geometrical information to simplify their manufacturability analysis.

A ruled surface can be defined as a linear surface delimited by two boundary curves (any curve), called rails, i.e. it results from the displacement of a rule (line) in space along rail curves. Ruled surfaces are present in our daily life, and they are essential surfaces of most of the mechanical parts as well. For example they are widely used in the design of aircraft structural parts. The various types of ruled surfaces can be classified into the following families: developable or non developable (quasi-developable surfaces are also considered as non developable). Developable ruled surfaces are the ones

that can be laid on a flat plane without distorting or ripping. Mathematically speaking, a developable surface has a zero Gaussian curvature at each of its points. Moreover, all the normal vectors to the surface in each point of a given surface rule are the same. For non developable surfaces, the second condition is not verifiable, i.e. all the normal vectors are not the same.

2. STATE OF THE ART

Shape recognition can be defined as finding the topological relation among points. In case of thick unorganized points cloud, this step requires both thinning (filtering) and ordering [8]. The influence of filtering depends on the model reconstruction technique used, whether is it analytical (conventional), or based on neural networks mapping [8-13]. In general, the methods involving neural networks explicitly rely on experimentation to determine their necessary parameters as done in Self Organizing Maps (SOM) in [8] and region growing in [18], which is undesirable because of error accumulation throughout the algorithm steps which will definitely deviate results within or larger to allowable tolerances. Also graphical methods are limited to certain part families or special topological shapes. In this work, our concern is in nonconventional techniques; however we will not base our algorithm on experimentation. As a result, filtering is used for point removal. Some work takes ordering into consideration and tries to overcome this issue as in [11] with regression techniques.

Because of their simple generation, ruled surfaces arise in a variety of applications including CAD, architectural design, NC milling with cylindrical cutter, and others [3]. Even though ruled surfaces are considered to be one of the simplest objects in geometric modeling, as they are generated by basically moving a line in space as described in [14], they impose lots of difficulties when it comes to their recognition and reconstruction. We can classify them into two groups: developable and non developable surfaces. A developable surface is a surface which can be developed into a plane without stretching or tearing, mathematically this means that developable ruled surfaces have the same tangent plane at all points of the same generator (rule) [16]. Some CAD software are able to recognize ruled surfaces from points generated by the same software, and not from any imported file. [6] Proposes an algorithm with the same intentions as ours, but it will not recognize a ruled surface from a given set of points, but from a given set of parametric equations.

3. FILTERING

In this section an adaptive filtering algorithm of cloud of points is proposed, to obtain a noiseless, highly reduced cloud of points conserving the topology of the digitized part. We assume scattered and unorganized set of points. As a first step we need to define each point relative to the others by computing the Euclidean distance (Eqn.3.1) between each and every point in the data set.

$$d_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2} \quad (3.1)$$

Next, virtual spheres with a radius r having the given points as centers are defined. The radius is computed according to internal algorithmic iterations in order to find the most suitable value of r . During iterations, no points are deleted yet, the remaining points for different values of r are counted. If the number of remaining points when r_{n+1} is used, remarkably deviates from the number of remaining points when r_n is used, r_n is set as our spheres radius

Points inside spheres are considered to be neighbors. Therefore spheres centers with largest number of neighbors are to be considered first, and points inside those spheres will be deleted, except for the center, followed by the ones having fewer neighbors in a decreasing order. This filtering only conserves the centers of the spheres.

Due to possible digitization inaccuracies, noise might be included in our data set, and might not be filtered using the previous steps. Therefore single and multiple noise testing are conducted to remove existing ones. Our approach for single noise removal consists on setting at first a dimensional parameter k , such as k is a multiple of the digitization machine accuracy p (k is to be specified by the user). We define the following conditions in order to test for single and multiple noise existence within the filtered points:

$$\text{if } d(P_i, P_j) = d_{\min} > k \quad P_j \text{ is considered as noise} \quad (3.2)$$

$$\text{if } \begin{cases} d(P_i, P_l) > k \\ d(P_i, P_k) < k \end{cases} \quad \text{Multi - noise search} \quad (3.3)$$

If the condition set in Eqn.3.2 is applicable, then P_j is considered as single noise and must be deleted. If the conditions of Eqn.3.3 is applicable, this is considered to be a multi-noise removal case. Therefore the nearest points are chained, and the number of points in each chain is counted. The number of points is then compared, and the chain with the largest number of points is conserved, while the others are removed. For that purpose we define “ $Max d_{\min}$ ” which is simply the maximum of all the minimal distances. This value is set as a requirement for points to be included in a same chain or not, i.e. a point P will be considered to be in the same chain with P_j , and P_k if the conditions (except the last) in Eqn. 3.4 are applicable. The chaining step stops when the last condition of Eqn.3.4 is applicable, and we proceed with the same way with all unchained points. Only the chain with the highest number of points is kept.

$$\text{if } \begin{cases} d(P_i, P_j) < Max d_{\min} \\ d(P_i, P_k) < Max d_{\min} \\ \dots \\ d(P_i, P_l) > Max d_{\min} \end{cases} \quad (3.4)$$

This step will result in a noiseless, filtered model. The concentration of points will be higher on sharp edges than on fillets and blends. This filtering algorithm was tested on a mechanical part (Figure 1) and on random 2-D point sets and proved to be fully functional. In the example shown in Figure 1, we were able to reduce the number of points from 27000 points to approximately 9000, i.e. around 60% reduction without modifying the topology of the part. Now that our points are filtered and removed all possible noises, we can proceed with the model reconstruction. In the next section we will detail three methodologies to identify and reconstruct ruled surfaces.

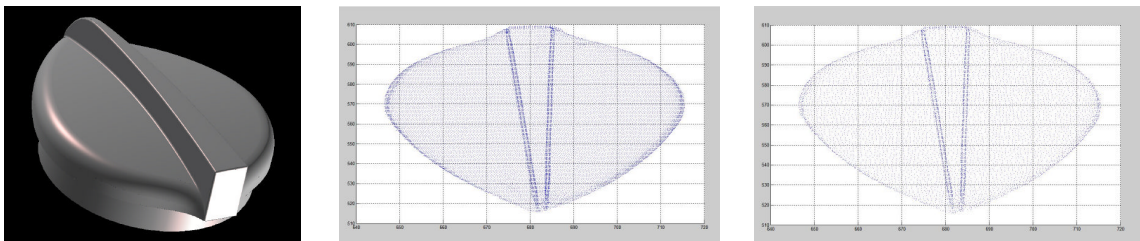


Fig. 1: “Bouchon Delahaye”: CAD model (left); before filtering (middle); after filtering (right).

4. IDENTIFICATION OF RULED SURFACES

In this section, three methodologies to recognize and reconstruct ruled surfaces from a given point cloud are detailed. We assume that our points can be delimited by a four sided “square”. The first methodology is based on using conventional geometrical methods (as dot product, cross product, planes and lines in space, and others) to identify if the given points constitute a ruled surface. The second methodology main idea is to delimit the set of given points, and then use geometrical and analytical methods on the boundary points to identify if the given points constitute a ruled surface or not. The delimiting was done using the convex hull approach, which can be found as a built-in function in MATLAB. The third methodology is inspired from the first and the second; we delimit the given set of points by using an algorithm developed for that purpose, and not using built in functions restricted to certain cases. Then use Bezier polynomials and other geometrical methods finally helps to determine the equations of the bounding curves. The degree of these curves reflects the type of the surface.

The three methodologies are coded on MATLAB, and tested on clouds of points, and each methodology contributed to convenient results. But the first and second methodology proved to be restrictive and

disadvantageous under testing. Therefore we briefly present the first two methodologies, and state their weaknesses along with their limitations. On the other hand, the third methodology, which covered a wider spectrum of ruled surfaces, is extensively described with all its assumptions. The last section is dedicated to the manufacturability analysis of ruled surfaces.

Before going into the explanations, it is important to state that our methodologies are not restricted to a given format of points or dependant of any CAD software. They are neutral algorithms that will perform the analysis of a given set of points without being dependant of the file type whether it was ASCII, Notepad, WordPad, etc. We are specifying this because some advanced CAD software are capable of automatic reconstruction of cloud of points generated by the same software; they are not general and suitable for all the cases. Also we should state that the filtering algorithm explained in Section 3 is built into the three methodologies, so any cloud of points imported is filtered before proceeding with the ruled surfaces recognition. Therefore, we are able to import, unfiltered and "noisy" set of points, without worrying about the performance of the methodologies. However each methodology has its own weaknesses and limitations that will be explored.

4.1 Methodology Based on a Geometrical/Analytical Approach

We will consider as a starting point for our first methodology a filtered set of scattered points defined in E^3 . To start, the cloud of points is meshed using Delaunay triangulation. Given a set of data points, the Delaunay triangulation is a set of lines connecting each point to its natural neighbors, i.e. for the data points defined by vectors \mathbf{v}_1 and \mathbf{v}_2 , returns a set of triangles such that no data points are contained in any triangle's circumscribed circle. For each triangle, we are able to determine:

- The unit vectors (\mathbf{u}_{ij} , \mathbf{u}_{ik} , \mathbf{u}_{jk}) of each of the sides
- The unit normal vector to the plane containing the triangle, by computing the cross product of two intersecting vectors. ($\mathbf{u}_{ij} \times \mathbf{u}_{ik}$, or $\mathbf{u}_{ij} \times \mathbf{u}_{jk}$, or $\mathbf{u}_{ik} \times \mathbf{u}_{jk}$)

The most repeated value can be tracked, and it is therefore possible to determine the dominant direction in the given set. The dominant direction is the one of the rule. Once the direction of the rule is determined, we will be able to determine tool axis position relative to the surface when studying the manufacturability.

As it can be noticed, this methodology is very limited, as it can only be applied to ruled surfaces, with a unique direction of rules. The existence of twists in the surface will lead to its failure. Also other disadvantages and weaknesses prevent this methodology from being adopted for recognition of ruled surfaces as:

- The duration due to the large number of computations. As the number of points increases the computation time also increases and the computational errors might accumulate throughout the algorithm operations, and as a consequence the final results deviate from the awaited ones, especially when the digitization was not done properly, or the digitization machine was not adequate.
- The parametric equations of the rails cannot be found, since usage of Bezier polynomials for surfaces or curves needs a special ordering of points. Each points ordering will contribute to different results. Therefore a difficulty arises when it comes to the search of the convenient points ordering, that will lead to the desired results,
- This method is very dependant of the accuracy of the point coordinates. A random set of points will lead to a deviation and inconvenience in the results because of rounding errors that will accumulate throughout the calculations.

4.2 Methodology Based on Analysis of 2D Convex Hull points

We consider as a starting point for this methodology a filtered, scattered set of points defined in E^3 , and we assume that the set of points belong to a ruled surface. Triangulation is needed to connect the scattered points of the given set to each others. Our purpose is to find the corners points of the given set. We will use for that purpose the 2D convex hull. The term "convex hull" is used to describe the boundary of the minimal convex set containing a given non-empty finite set of points in the plane. Unless the points are collinear, the convex hull in this sense is a simple closed polygonal chain.

Now our concern will be concentrated on the convex hull points, i.e. the corner points. We will determine, using the data obtained from triangulation of the points, the triangles having as one of their summits, a convex hull point. Having the bounding triangles of our set, and their summits, we will

determine the normal vectors to each plan containing a bounding triangle, by computing the cross product (Eqn.4.1, Eqn.4.2) of two intersecting vectors in each of the bounding triangles.

$$\vec{v}_{i1}, \vec{v}_{i2}, \vec{v}_{i3} \quad \left\{ \begin{array}{l} i = 0 \text{ to } n \\ n \text{ is the number of bounding triangles} \\ \vec{v} \text{ is the vector defined by the triangle summits} \Leftrightarrow \text{defining a plane} \end{array} \right. \quad (4.1)$$

$$\vec{n}_{triangles} = \vec{v}_{i1} \times \vec{v}_{i2} \quad (4.2)$$

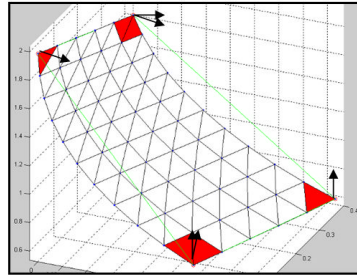


Fig. 2: The triangulated set with normal vectors at convex hull points.

In order to determine the type of the edges of the given set, whether a straight line, or higher order curves, we need to determine the points defining the edges of the set (the bounding points of our set):

- Mathematically we can define a plane by a normal vector and a point. We determine the vector joining two consecutive convex hull points,
- We already computed the normal vectors at the convex hull points,
- We compute the cross product of the vector defined by two consecutive convex hull points and the normal vectors to bounding triangles convex hull points, i.e. we determine the equations of the planes delimiting our point cloud,
- We check which points from the given set verifies the equations of the bounding planes,
- As a final step in order to determine the type of the edges of the given set, we use Bezier polynomials. If we don't obtain at least one linear equation in one of the planes then the surface is not ruled.

This methodology was coded on MATLAB. After testing the weaknesses and limitations of this methodology can be summarized by the following:

- The points should not be very close to each other, or else we will not obtain the inquired boundaries. We will obtain points not belonging to edges.
- In case of two or more triangles at a single convex hull point, we have to consider two normal vectors and re-compute everything, or we –have to determine the resultant from the two normal vectors and continue as if we have only one normal vector. We will be deviating from our main purpose. This problem tends to increase dramatically when the number of triangles at convex hull points increase (\Leftrightarrow when distance between points decrease).
- The 2D convex hull function is not reliable when it comes to concave set of points. It doesn't take into consideration the change in curvature of the surface.
- The search for profile points using bounding planes equations will be affected by computational and rounding errors. We might not obtain the real bounding planes equations.

4.3 Methodology Based on Classification of Points

The approach that will be detailed in this sub-section is inspired from the two previous methodologies that were limited under testing. We will use the idea of delimiting the given set of points of the second methodology, but now using an algorithm we developed for that purpose, and not using built in functions restricted to certain cases. Then use Bezier polynomials and other geometrical methods to determine the equations of the bounding curves, as for the first and second methodologies, to determine the equations necessary for recognizing if the surface is ruled or not.

After importing the filtered cloud, we will order the points in an ascending order. The sort will be based on the first column of the matrix of points coordinates ($m \times 3$ matrix such as m is the number of points). For any rows that have equal elements in a particular column, sorting is based on the column immediately to the right.

The next step is the classification of points. When we scan an object, optically or mechanically, to obtain its numerical model, we do not vary the three dimensions at the same time. The scanning head is fixed for a specific value on one of the three dimensions X , Y , or Z , and scan in the plane defined by the two other dimensions. For example, when digitizing for a fixed value of Y we obtain values of X and Z that generate a certain profile, then for another value of Y we obtain other values of X and Z that generate another profiles, and etc. So in the overall given set, representing the merged sets of scanned points, one dimension will be varying much less than the others. The later is due to the merging techniques used when merging two or more views of the same digitized object. But the assumption made is restrictive, because it will not contribute to a convenient result, in case the scanning was done with an angled scanning head. So for the rest of the explanation, assume a scanning head aligned with one of the three dimensions, as well as a face belonging to the considered part. The goal of classifying the points according to their least varying dimension is to delimit the given set of points; in fact the first and last class obtained constitutes two of the boundaries. The other two boundaries are constituted from putting the first points of each class in an array, and the last points of each class into another array respectively. This approach for delimiting the given set of points is not limited only to convex set of points but also to concave sets as well.

Since our points are now ordered in an ascending order and classified in well defined classes, we can use now Bezier polynomials (Eqn. 4.3) to determine the parametric equations of the profiles generated by the classes of points obtained in the previous step.

$$\vec{P}(t) = \sum_{i=0}^N B_{i,N}(t) \cdot \vec{S}_i$$

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i} \tag{4.3}$$

$$\binom{n}{i} t^i (1-t)^{n-i} = \begin{cases} \frac{n!}{i!(n-i)!} & \text{if } 0 \leq i \leq n \\ 0 & \text{else} \end{cases}$$

The probable appearance of high order equations is due to non rounding operations along the various steps of the algorithm. Now we will need to determine whether the surface is ruled or not. A rule as defined in the beginning of Section 1 is a line. Therefore if the surface is ruled the unit vectors joining points of the same level, on different profile curves should be the same (in Figure 3, if the unit vectors $(X1, X2)$, $(X1, X3)$, $(X1, X4)$, $(X1, X5)$, $(X1, X6)$ are the same then the curve joining the points $X1, X2, X3, X4, X5, X6$ is a line. We will proceed in the same way for $X1', X2', X3', X4', X5', X6'$ and all the other points, following the same methodology.

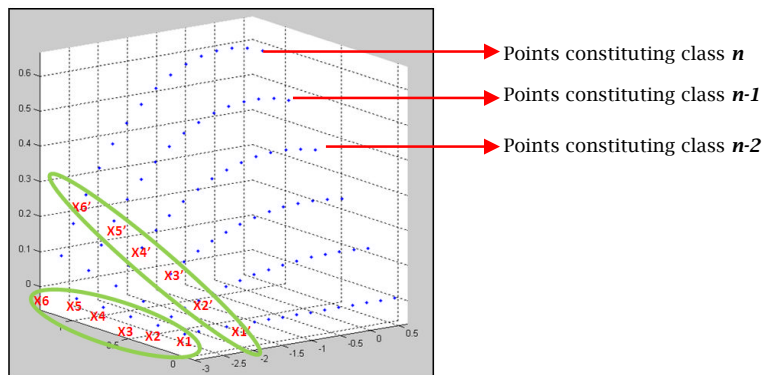


Fig. 3: The given set with points on the same level, on different profile curves.

Now that we have the unit vector of each rule, if the surface is ruled, we will determine the parametric equation for each line using the parametric equation for a line in space (Eqn. 4.4).

$$\begin{aligned} x &= x_0 + at \\ y &= y_0 + bt \\ z &= z_0 + ct \end{aligned} \tag{4.4}$$

4.3.1 Implementation and Results

The previously detailed methodology was coded on MATLAB, and contributed to convenient results for different test cloud of points as shown in Figure 4 and Figure.5. The test points cloud presented in Figure 4 represents a non developable ruled surface. As we can see in Figure 5, the test points cloud is a complex due to the presence of a twist. We can state that the presented methodology overcame the following limitations, which were the main weaknesses of methodology 1 and 2:

- Existence of a twist in the given surface.
- It is not only applicable to developable ruled surfaces, which give it wider fields of application.
- It was not affected by the high number of points, or by the curvature changes
- The computation time was highly decreased, since long and repetitive calculations were not required.
- The number of points and the distance between points was a highly affective parameter in the previous methodologies.
- As stated before, this methodology is applicable to concave and convex set of points.

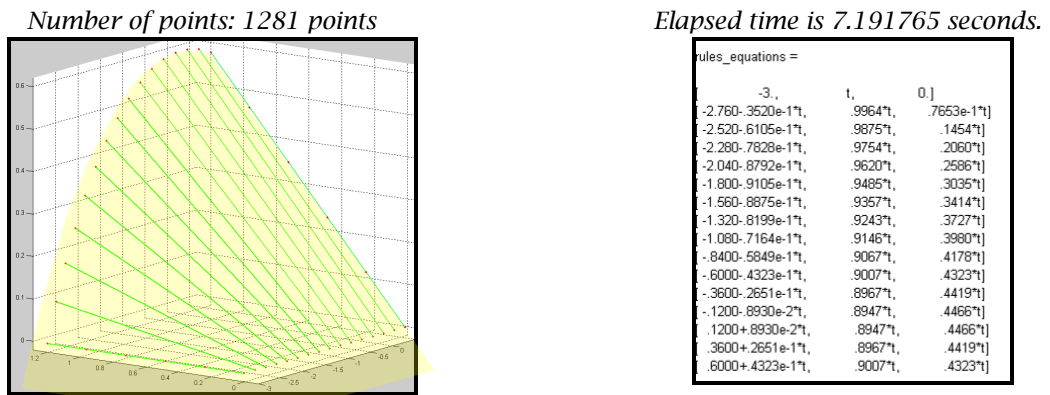


Fig. 4: Results obtained for test cloud 1.

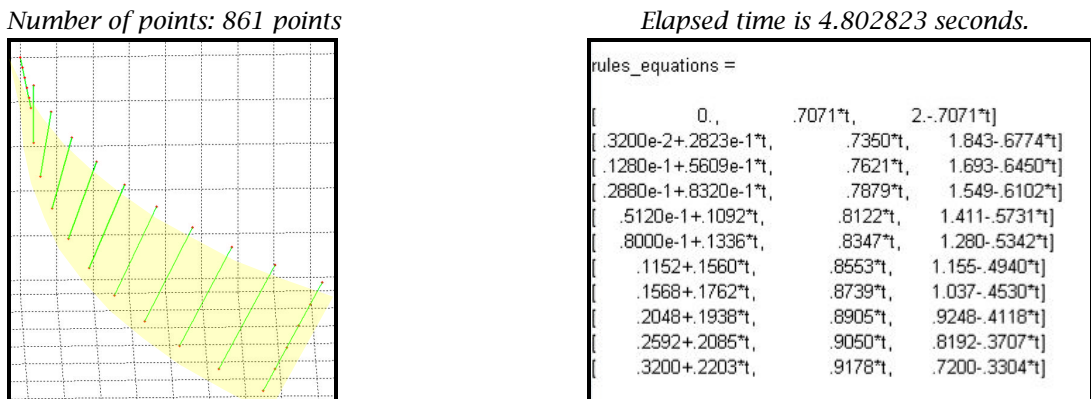
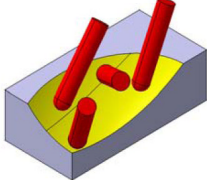
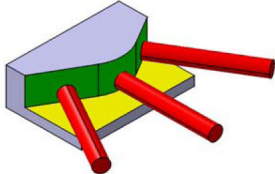


Fig. 5: Results obtained for test cloud 2.

5. MANUFACTURABILITY

Now that we have explored the ruled surfaces from the geometrical aspect, and presented methodologies that can be used to identify and reconstruct ruled surfaces if they exist in a given set of points, we will give in this section a perspective on the phase that succeeds model reconstruction, and present some the complexities and requirements for its success.

The most suitable manufacturing mode of ruled surfaces is flank milling, i.e. by the cutting length of the tool. This mode minimizes manufacturing time, and results in smooth surfaces faithful to the product CAD model. A simple comparison between end milling and flank milling is presented in Tab.1.

<i>End Milling</i>	<i>Flank Milling</i>
 <ul style="list-style-type: none"> ▪ Require lots of time, due to the necessity of multiple passes over the same area to guarantee smoothness ▪ Expensive due to need of manual operations. ▪ Resulted surface is not smooth 	 <ul style="list-style-type: none"> ▪ Reduction of number of passes over the same area by 14 times ▪ Reduction of costs by 75% ▪ Best smoothness that can be achieved ⇔ this will have a “snowball” effect on the following operations

Tab. 1: End milling vs. Flank milling [7].

- Determining if a surface is ruled or not, is not enough for manufacturing, we should also define whether it is developable or non developable. This classification will determine whether overcuts or undercuts will exist while manufacturing the part or not. (Figure 6)
- To manufacture ruled surfaces, the axis of the manufacturing tool should have the same direction as the rule of the surface and should move along its rails with an offset “e”. The value of “e” will be a function of the optimal position of the tool to minimize overcuts and undercuts if non developable surface or a constant distance between the tool axis and surface rule (radius of the tool) if developable surface. (Figure 7)
- If non developable surface, the interpolation of the curve that allow the manufacturing of the surface without overcuts and undercuts is needed. The margin of errors of the approximation (tolerances) can be set by the user based on his knowledge. A minimal margin for overcuts and undercuts must be accepted, since manufacturing cannot supply 100% products faithful to the CAD model. We propose that a methodology based on “confidence and prediction interval” for linear regression can be used to determine the value to offset the surface, in order to minimize resources and errors relative to CAD model.

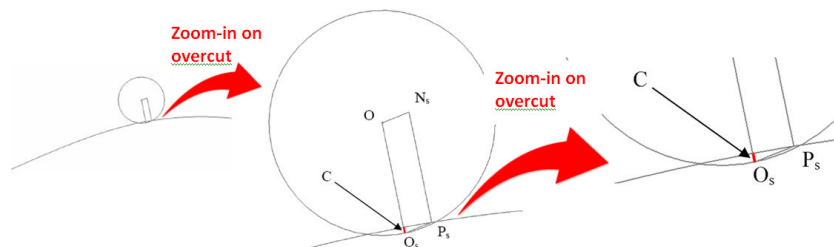


Fig. 6: Overcut in a quasi-developable ruled surface [6].

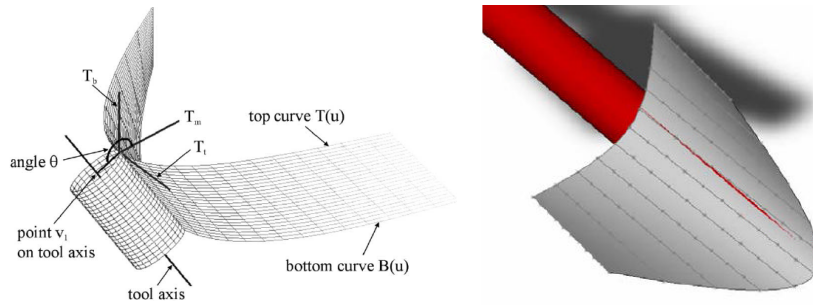


Fig. 7 (from left to right): (a) Position of the tool on the surface; (b) Overcut example [6].

6. CONCLUSION

The reverse engineering chain is used as enhancement solutions for CAD models started from specifications. Due to this chain, it is possible to remove imperfections, enhance functionality, and improve esthetics. It requires less time, and enables automation. The above detailed methodologies constitute a step towards automation of model reconstruction. From an unorganized, scattered cloud of points, exported from any CAD software we recognized and reconstructed a ruled surface if it exists, due to the filtering algorithm and the three methodologies presented. The above completed work will enrich the CAD model with geometrical information related to ruled surfaces as rule direction at each class of points, the rails equations, and others. This will simplify the process planning of the part because it will allow the operator or the CAM software to determine the tool position, tolerances, and path to trace in order to overcome design difficulties while manufacturing the ruled surfaces. As stated previously, the restrictions of the assumptions made limit our methodologies. However, they contributed to convenient results for when applied to complex cloud of points.

7. PERSPECTIVES

The above stated work constitutes a step towards automation, but more work need to be done to fully automate model reconstruction. We will seek to expand our assumption made in Section 4.3, and as result our algorithm will be applicable to all possible positions of the scanning head, and all possible object orientations.

8. REFERENCES

- [1] Benko, P; Martin, R. R; Varady, T.: Algorithms for reverse engineering boundary representation models, *Computer Aided Design*, 33, 2001, 839-851.
- [2] Chan, C. K; Tan, S. T.: Extreme points of a large 3D point set along multiple directions, *Computer Aided Design*, 37, 2005, 17-34.
- [3] Chen, H. Y; Pottmann, H.: Approximation by ruled surfaces, *Journal of Computational and Applied Mathematics*, 102, 1999, 143-156.
- [4] Demarsin, K; Vanderstraeten, D; Voldine, T; Roose, D.: Detection of closed sharp edges in point clouds using normal estimation and graph theory, *Computer Aided Design*, 39, 2006, 276-283.
- [5] Gong, H.; Cao, L. X.; Liu, J. L.: Improved positioning of cylindrical cutter for flank milling ruled surfaces, *Computer Aided Design*, 37, 2005, 1205-1213.
- [6] Harik, R.; Herviou, D.; Lombard, M.; Ris, G.: Etude d'usinabilité en flanc d'une face en ingénierie intégrée, 17eme Congrès Français de Mécanique, 2005.
- [7] Harik, R.; Derigent, W.; Ris, G.: Computer Aided Process Planning in Aircraft Manufacturing, *Computer-Aided Design and Applications*, 5(1-4), 2008, 953-962.
- [8] Kumar, S. G.; Kumar Kalra, P.; Dhande, S.G.: Curve and surface reconstruction from points: an approach based on self organizing maps, *Applied Soft Computing*, 5, 2004, 55-66.
- [9] Lartigua, C.; Duc, E.; Affouard, A.: Tool path deformation in 5-axis milling using envelope surface, *Computer-Aided Design*, 35, 2003, 375-382
- [10] Larue, A.; Anselmetti, B.: Deviation of a machined surface in flank milling, *International Journal of Machine Tools & Manufacture*, 2003, 129-138.

- [11] Lee, I. K.: Curve reconstruction from unorganized points, *Computer Aided Geometric Design*, 17, 2000, 161-177.
- [12] Maekawa, T.: An overview of offset curves and surface, *Computer Aided Design*, 31, 1999, 165-173.
- [13] Peternell, M.: Developable surface fitting to point cloud, *Computer Aided Geometric Design*, 21, 2004, 785-803.
- [14] Peternell, M.; Pottmann, H.; Ravani, B.: On the computational geometry of ruled surfaces, *Computer Aided Design*, 31, 1999, 17-32.
- [15] Peternell, M.; Steiner, T.: Reconstruction of piecewise planar objects from point cloud, *Computer Aided Design*, 36, 2003, 333-342.
- [16] Pottmann, H.; Wallner, J.: Approximation Algorithms for Developable Surfaces, *Computer-Aided Geometric Design*, 16(6), 539-556, 1999.
- [17] Senatore, J.; Monies, F.; Redonnet, J. M.; Rubio, W.: Analysis of improves positioning in five-axis ruled surface milling using envelope surface, *Computer Aided Design*, 37, 2005, 989-998.
- [18] Vieira, M.; Shimada, K.: Surface mesh segmentation and smooth surface extraction through region growing, *Computer Aided Geometric Design*, 22, 2005, 771-792.
- [19] Wang, J.; Oliveira, M. M: Filling holes on locally smooth surfaces reconstructed from point clouds, *Image and Vision Computing*, 25, 2007, 103-113.
- [20] Zeid, M: *Mastering CAD/CAM*, McGraw-Hill Science/Engineering/Math, 978-0072976816, 2004.
- [21] Zhang, L.Y.; Zhou, R. R.; Zhou, L. S: Model reconstruction from cloud data, *Journal of Materials Processing Technology*, 138, 2003, 494-498.