The development and implementation of a reverse engineering method for near net shape parts

by

Niechen Chen

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Industrial Engineering

Program of Study Committee:
Matthew Frank, Major Professor
Frank Peters, Major Professor
James Oliver

Iowa State University

Ames, Iowa

2015

TABLE OF CONTENTS

ACKNOWLEDGMENTS

I would like to thank my major professor Dr. Matthew Frank. His patience and inspiring advice lead me through this two years research and study. His kindness makes my working experience enjoyable and pleasant. I would like to sincerely express my respect and thanks to him.

Next I would like to thank my major professor Dr. Frank Peters. His knowledge and experience in casting provides me with a lot of valuable ideas for this research.

I would like to thank Dr. James Oliver for serving on my committee and support me throughout the course of this research.

I would like to thank Prashant Barnawal and Nicholas Hennessy for their help through this research work in Rapid Manufacturing and Prototyping Laboratory (RMPL).

In addition, I would also like to thank my friends, colleagues, the department faculty and staff for making my time at Iowa State University a wonderful experience.

Last but not least, thanks to my parents for their encouragement. Thanks to my wife for being here to take care of me and support my study.

Funding for this research was provided by a grant from the National Science Foundation, Center for E-Design, and in collaboration with the US Army Research Laboratory.

ABSTRACT

This research presents a new method for the reverse engineering of Near Net Shape (NNS) parts that bridge the current 3D scanning and Rapid Prototyping technologies. Near Net Shape is a group of manufacturing technologies that includes forging, casting, hot isostatic pressing, and additive manufacturing. This research focuses on casting process and provides a software tool along with the new method for reverse engineering a legacy casting design to the “as was” casted state instead of the “as is” current state, and at the same time, reducing the cost and time for repairing a legacy casting part.

The three main objective for this research is to 1.Create a new reverse engineering method 2.Develop a software tool that is designed for feature free model editing 3.Validate the process through metal casting.

The Point Cloud Library is applied for assisting point cloud processing and feature free model editing. A series of algorithms is developed for draft adding and pattern generation for the process of casting. The Rapid Pattern Manufacturing system developed in Iowa State University, Rapid Manufacturing and Prototyping Lab is applied for pattern manufacturing.

This method is validated to be correct and able to reverse engineer legacy casting parts rapidly and economically through a metal casting process.

The layout of this thesis is as follows: Chapter 1: provides introduction, background, research problem statement and objective of this research. Chapter 2: a literature review for the current reverse engineering method and introduces the modules of point cloud library that are used in this research. Chapter 3: presents the overview of method and algorithms that developed for this method in detail. Chapter 4: presents the implementation of this method

and gives the analysis of the demo metal casting process. Chapter 5: provides future work and conclusions.

CHAPTER 1. INTRODUCTION

1. 1 Background

This research involves methods for the reverse engineering of Near Net Shape (NNS) parts that bridge the current 3D scanning and Rapid Prototyping technologies. Repairing a legacy part that was designed and manufactured decades ago is a difficult problem in industry. Current Reverse Engineering and Rapid Manufacturing methods only capture what is presented "today" and can then re-produce what the scanned geometry looks like. However, for a Near Net Shaping part from casting, for example, the current part being reverse engineered does not necessarily represent the original design, or even the part after the NNS process. This is due to a variety reasons; the original design was modified for casting, the part shrunk during process, and almost assuredly, the part was machined to either add features or meet critical tolerances. Thus, a method to transform the scanned model of the "as is" geometry of a legacy part to the "as was" geometry for NNS would be very advantageous; hence the topic of this research.

Feature-free model editing is the main obstacle for transforming the "as is" geometry to the "as was" geometry for near net shaping tooling. Current professional CAD tools such as Solidworks, CATIA, UG NX and etc, are designed for creating feature-based geometry models and lack the ability to handle feature free models; ones that are not generated through convention methods of *primitives*, *features* and *sketching*[1]). There are some software products intended for creating or editing feature free models such as Autodesk 3DS MAX; however, they are more focused on the needs of the art and gaming design fields as opposed to industrial component design. Although it is possible to accomplish what is proposed in this work using current CAD tools, it would require facet-to-facet or point-to-point manual editing of the reverse engineering data, it would be a very time consuming process by an expertly skilled technician.

In this work, we focus on the transformation of the "as is" geometry of a cast part into the "as was" tooling used in the mold making process. In the casting process, there are basically two types of surfaces: cast surfaces, cast and then machined surfaces. Cast surfaces are those surfaces only being created from the casting process and without any post processing like machining. They are typically freeform surfaces that cannot be easily regenerated with CAD tools, and all their geometry should be captured and maintained. In contrast, cast and then machined surfaces are those surfaces that were metal cast, but then later post-processed using machining. These surfaces are more typically critical design features such as holes, pockets, slots, bosses or profile features [1]. Cast and machined surfaces require machining allowances (extra material) to be added in order for there to be some material to machine through and create the final surface.

1. 2 Overview of Data Capture Methods for Reverse Engineering

As the first step of reverse engineering, *data capture* means digitizing an existing part to point clouds. Point clouds are composed of a group of individual points that are on the surface of the part in the 3D vision coordinate system.

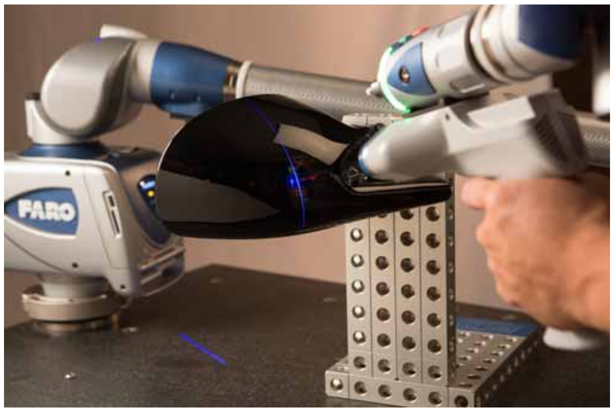


Figure 1.1. 3D Laser Scanner Working

The methods of digitization can be categorized into two large groups: *Contact* techniques and *Non-Contact* techniques. The most common contact techniques is the use of a Coordinate Measuring Machine (CMM). Non-contact techniques has developed into various forms as structured lighting, spot ranging, range from focus, range from texture and stereo scanning[2], or industrial used X-ray, magnetic resonance imaging (MRI) and computed tomography (CT) . In this research, a non-contact technique of laser scanning is used for data capture. Many devices are available on the market for non-contact data capture. Non-contact data capture device uses lasers, optics and charged couple device (CCD) sensors to capture data[3]. Figure 1.2 shows a FARO Edge ScanArm® ES device, with proposed accuracy of ±35µ (±.0014 in.) and a scan rate up to 45,120 points/sec[4]. Figure 1.2.2 shows the general method of a single camera solution triangulation scanner. The 3D coordinates of a point on the surface of the object can be calculated through a triangle that makes up the laser path and the constant base value[5]. This type of laser scanning also has limitations. The accuracy of this laser scanner decreases as the distance between the instrument and the object increases[5]. Also, because the calculation of the coordinate is based on forming a triangle, there are some occluded surfaces that the scanner cannot reach in a particular setup.

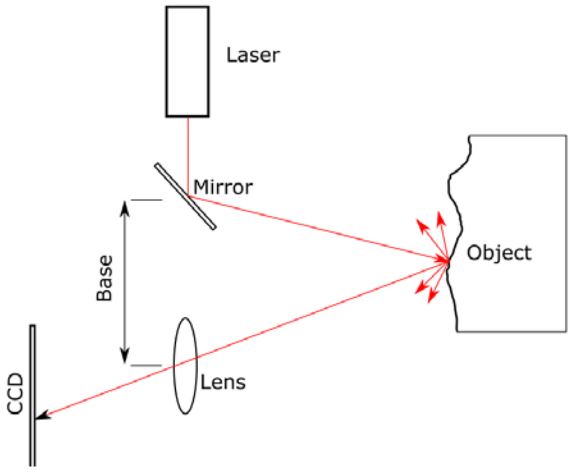


Figure 1.2. 3D Laser Scanner Principle

1. 3 Overview of sand casting process and its geometry characteristics

Sand casting, also named as sand mold casting, with the history of thousands of years, is one of the most popular manufacturing technology in industrial production because of its ability to directly create complex geometries. Sand casting process is basically conducted in five basic steps: 1. pattern making 2. core making 3. mold making 4. metal pouring 5. shake out. As the first step of the whole sand casting process, pattern making provides the final product with the designed shape. A complete pattern for sand casting is usually consists of two parts: cope pattern and drag pattern. Cope pattern and the drag pattern is split from the casting design geometry by the parting line. The mold is essentially the reverse geometry of a casting design. By pouring sand with chemical bond into the pattern box assembled from cope pattern and drag pattern, cope mold and drag mold can be created respectively. Pouring molten metal into the assembly of cope and drag mold, the desired shape will be created through the solidification process. Finally, shaking out the sand mold, the final cast part will be obtained.

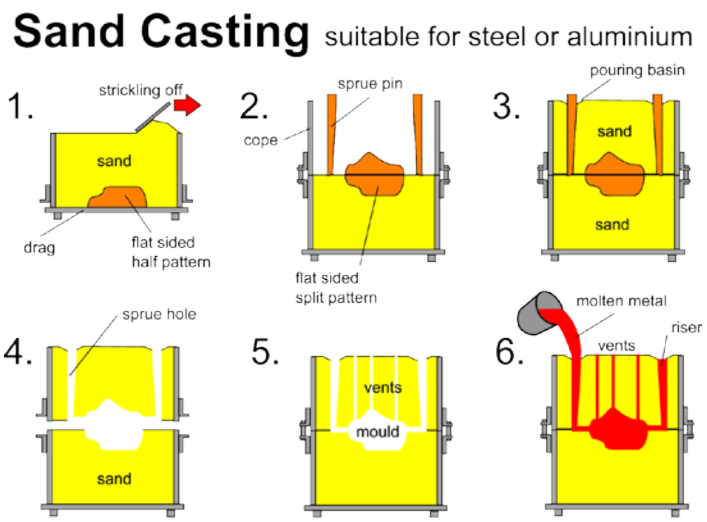


Figure 1.3. Sand Casting Process[34]

There are three types of geometry feature in sand casting design: draft, parting lines and cores[6]. First, draft is a sloped surface added to vertical surfaces that allows the pattern or core to be removed (pulled) from the mold during processing [6]. Draft feature is always removed trough post machining process on those cast then machined

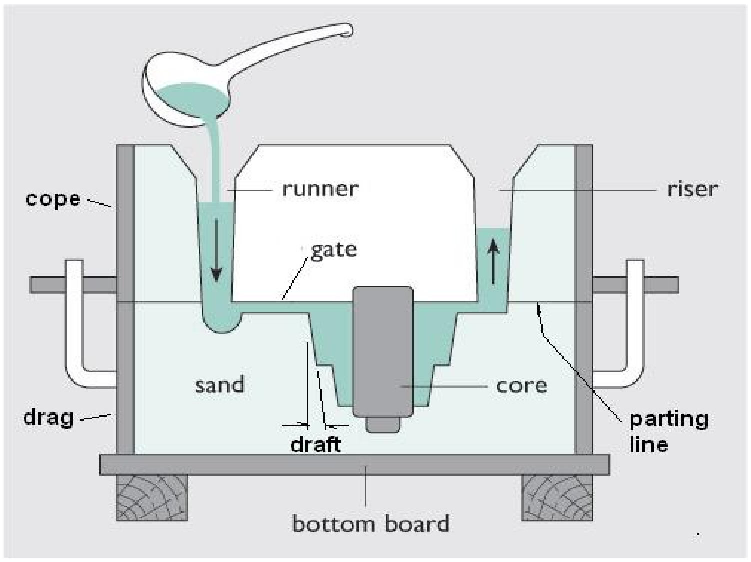


Figure 1.4. Sand Casting Geometry feature[35]

surfaces. Therefore we lack point cloud data on what the original casting tooling looked like. In this case, if we know that a surface will be oriented parallel to the parting direction, the draft surface must be recreated according to the principle of casting design. Second, the parting line is often designed as plane or set of planes that designed for splitting the pattern, so the pattern can be removed from the mold when the mold is pulled.[6]. In this work, the determination strategy of the parting line will not be specifically studied. Relevant people have already solved this problem. An assumed parting line will be used, and the parting line can be input by the user in the final implementation. Third, cored feature is the group of surfaces that are created from *core*. The *core* is separate part of the mold that is used for producing internal cavities and other nonvisible surfaces, depending on the parting direction. Cored regions are beyond the scope of this thesis work; all cored area surfaces will be excluded and not recreated for the reverse engineering process. This work will instead focus on what is conventionally called the “cope” and “drag” tooling, which are often consider the “two halves” of the mold surfaces, and are the necessary and most numerous pieces made (not all designs require cores or other ancillary mold pieces).

1. 4 Overview of Rapid Pattern Manufacturing System[7]

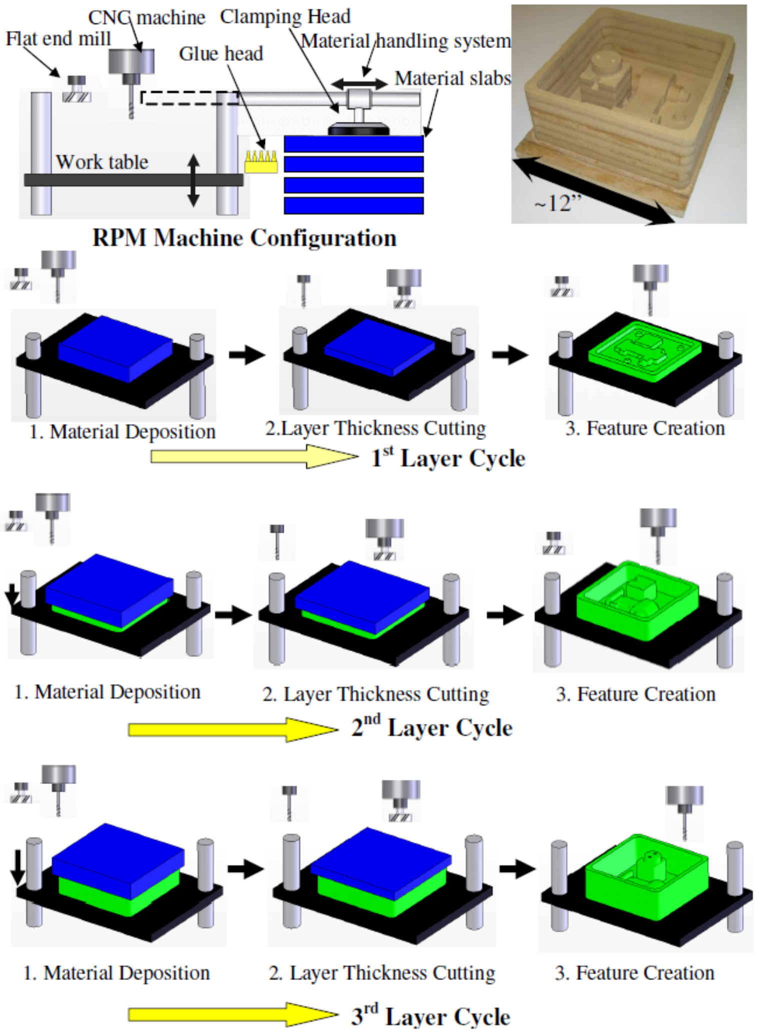


Figure 1.5. Rapid Pattern Machine Working Cycle

The Rapid Pattern Manufacturing (RPM) system is a machine that rapidly creates wood patterns for sand casting; which was developed in the Rapid Manufacturing and Prototyping Laboratory (RMPL) at Iowa State University. This system can automatically generate pattern tooling directly from a CAD model. It uses a concept of *hybrid* additive/subtractive manufacturing to create sand casting patterns from layers of wood slabs.

Figure 1.3.1 shows a step-by-step overview of the RPM process. The entire machine is composed of a material handling system, gluing system and a CNC router machine. The material handling system feeds wood sheets into the work space. The gluing system selectively sprays glue on the bottom of the wood sheet in the shape of the part cross section. After gluing, the wood sheet is placed on the work table and machined. This iterative process continues, where each layer is added and machined, and a complete wood pattern for sand casting will be delivered in hours.

1. 5 Research Problem and Objectives

A part after manufacturing often differs from its original design, especially for near net shape processes like casting. When dealing with legacy parts, the CAD models or even blueprints are likely not available or inaccurate. Replacing this type of decades-old legacy part cannot be justified when production level volumes are not needed. A method to reproducing a legacy part with low cost and short time is needed. In addition, current reverse engineering methods and rapid manufacturing systems insufficiently handle this problem. Even when the system can capture geometry, as stated above, it is the “as is” geometry, not what we truly need. Therefore, a reverse engineering method is needed that can re-create the part shape of the “as-was” design. This research focuses on the revere engineering of legacy parts that were manufactured through casting processes, specifically sand casting. This overall objective of this research is to integrate advanced reverse engineering through laser scanning with a hybrid method for the additive/subtractive manufacturing of wood pattern tooling for sand casting. In order to meet this, the following sub-objectives are defined:

Create a new reverse engineering method: this work will create a new reverse engineering method that can recreate the geometry not just as what the given part looks like, but as what it looked like prior to near net shape processing (the pattern geometry).

Develop a software tool that is designed for feature free model editing: a software tool will be developed that can do geometry editing on feature free models and automatically generates the pattern tooling geometry for casting.

Validate the process through metal casting; using the developed software, a metal cast component will be reverse engineering, tooling made, mold produced and metal cast in order to recreate the part.

CHAPTER 2. LITERATURE REVIEW

2. 1 Previous approach for reverse engineering

In the past twenty years, there has been a considerable amount of knowledge based reverse engineering approaches targeting at providing a design feature database that allows the recreation of a CAD model of a design from the scanned point cloud.

Since 1980s, with the development of computational geometry, a lot of research work has focused on recognition, reconstruction of 3D shapes[8][9][10] and related algorithms. A possible concept for reconstructing the CAD structure from a physical part was gradually brought to reality starting from 1997[11]. This method is summarized as four steps: 1. Data capture, 2. Preprocessing, 3. Segmentation and surface fitting, 4. CAD model creation. The step 3 and step 4 are the major steps. Segmentation, generally dividing the scanned point clouds into subset clusters of points, such that each individual cluster represents a natural surface. Surface fitting then find the optimized surface type that can fit the cluster of points. Based on the surface fitting process, a CAD model that is defined by surface primitives or solid primitives can be created.

In 1999, a research group from the University of Utah created a Reverse Engineering-FeAture-Based (REFAB) prototype system[12]. Their approach used geometric primitives to fit scanned data. Geometric primitives includes lines, arcs, polygons parametric curves and surfaces[13]. From geometric primitives, basic machining manufacturing features such as stock, holes, slots, pockets, facing features, groves, bosses and profile features can be created. From this essential idea, this group used an interactive approach that allows the user to assign five types of 2½D features: stocks, simple holes, profile pockets, profiles islands, and profile sides to the segmented 3D position points. Then from these five 2½ D features to recreate the design. Although it is only limited to 2½D features it shows the possibility of recreating a feature based CAD model from the scanned point cloud.

In 2009, a research group from France developed a Knowledge-Based Reverse Engineering (KBRE) approach[14]. This KBRE approach evolves from geometry feature based CAD model reverse engineering to a manufacturing knowledge and function knowledge based approach. First, a knowledge base of geometric shapes that can be created from a manufacturing process is developed. Then, by studying the part, the manufacturing process for this part can be identified. Finally, fitting all the geometric shapes related to the manufacturing process onto the segmented point cloud to recreate the functional and structural skeleton (FFS) of a part which is an assembly of manufacturing features. This KBRE approach is highly related to the nature of an industrial design, and more closely represents the ultimate goal of recreating the CAD model from a physical part.

Although researchers from all over the world have made considerable progresses on Reverse Engineering technology, a perfect reverse engineering tool that allows us recreate a complete CAD model from a physical part automatically or effortlessly does not exist. Thus, the current reverse engineering process can only re-produce the “as-is” geometry of the scanned part.

2. 2 Shape recognition

Shape recognition is an important process for reverse engineering. The point cloud is the direct raw geometry data that can be captured from a physical part. Shape recognition can extract design or manufacturing features from the point cloud. Generally, shape recognition is conducted in two steps: segmentation and shape fitting. Segmentation can also be referred as border detection. This process is used for extracting surface patches that are potential geometric features, so the part can then be proceeded to the next step of shape fitting. Currently, the existing segmentation method can be summarized as two main approaches: a border based approach and a region based approach[15]. The border based approach starts by identifying those points that are on the edges of a model and then linking those points to form a continuous wire frame of the edges of the model[16]. The region based approach starts from randomly assign seed points on the point cloud. It starts the region growing from the seed point to its neighboring points, then take the new points and seed points and iterate over and over until it detects that the new neighboring points reach the threshold of the boundary criteria[17].

After the whole point cloud is segmented and divided into separated point clusters. The process of shape recognition can begin. The most popular shape recognition method is the Random Sample Consensus (RANSAC) paradigm, introduced by Martin A. Fischler and Robert C. Bolles[18] in 1981 and then improved by R. Schnabel, R. Wahl and R. Klein[19] in 2007 for point cloud shape detection. The RANSAC paradigm construct shape primitives by randomly reading the minimum number of points that are required to define a shape primitive. It then compares the resulting candidate shape primitive to determine the best fit shape primitive for a given point group[19]. This method has a simple concept and a robust algorithm to deal with point cloud with noises.

2. 3 Point Cloud Library (PCL)[20]

Point Cloud Library (PCL) is an open source library for point cloud processing[21]. The whole library contains 12 modules: filters, features, keypoints, registration, kdtree, octree, segmentation, sample consensus, surface, recognition, IO and visualization. The module segmentation and module sample consensus is used in this thesis work. The

PCL_Segmentation library provides algorithms for segmenting a point cloud into distinct clusters[22]. The PCL_sample_consensus library provides sample consensus algorithms like RANSAC and modules such as plane, cylinder, sphere and more geometry primitives in development[23].

CHAPTER 3. METHOD AND ALGORITHMS

3. 1 Method Overview

This chapter presents a new method for the reverse engineering of legacy casting parts. A flowchart describing the process steps of this method is presented in Figure 3.1.

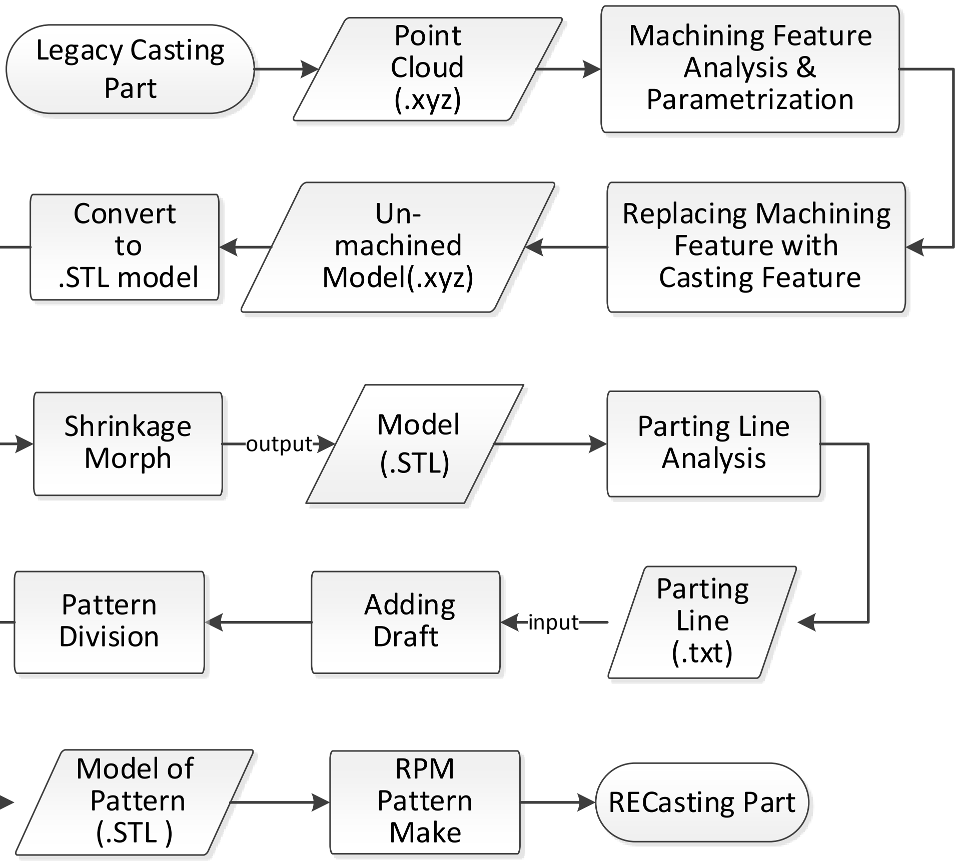


Figure 3.1. Method Flowchart

The entire method begins with similar steps seen in conventional reverse engineering, from data acquisition to preprocessing and then segmentation and fitting[11]. However, after segmentation and fitting, the proposed method focuses on modifying the model instead of directly converting it into a CAD model. Replacing the machining features with casting

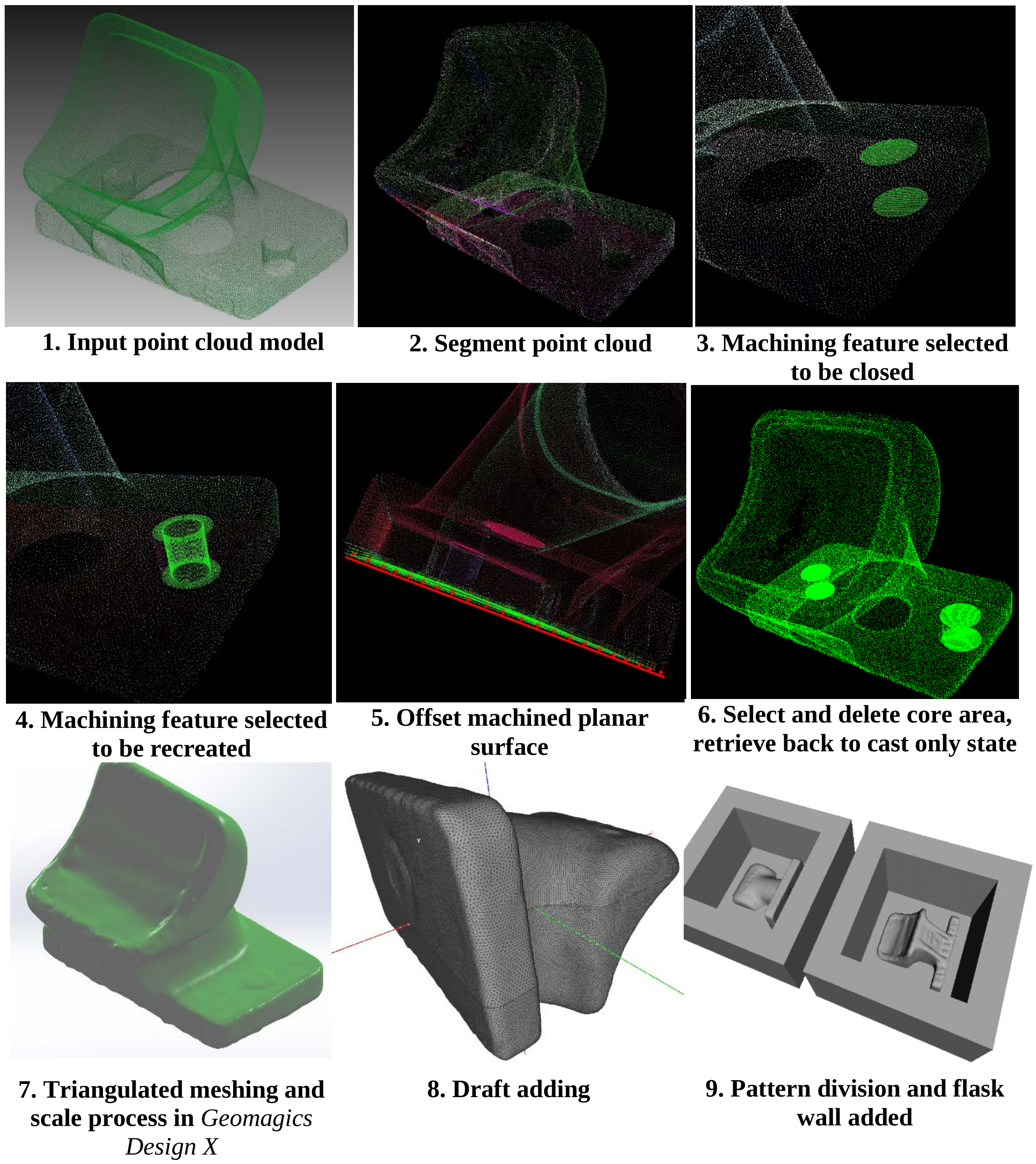

1. Input point cloud model
2. Segment point cloud
3. Machining feature selected to be closed
4. Machining feature selected to be recreated
5. Offset machined planar surface
6. Select and delete core area, retrieve back to cast only state
7. Triangulated meshing and scale process in *Geomagics Design X*
8. Draft adding
9. Pattern division and flask wall added

Figure 3.2. Method Overview with Data flow

features is essentially an extension of feature recognition and fitting. Basic machining features are often primitive shapes, such as planes, cylinders or combinations of them. Through the segmentation and fitting process, machining features can be isolated and removed or recreated into a shape that can be designed. Since we do not consider the surfaces of any cored areas, all core areas are simply selected and removed. After this point, the model is transformed from a point cloud model to triangle mesh model before proceeding. The step designated as “shrinkage morph” on the mesh model is a function to scale up the model to compensate for the solid- solid shrinkage that occurs during the casting process. The next step involves the parting line definition, wherein a polyline shape parting line can be defined based on user input, and this line will be defined in 2D coordinates and input in text format. From the parting line defined by the user, a draft adding function will automatically add draft on the polygon mesh according to the degree value that the user inputs. Now the model is nearly derived back to the “as-cast” state. In the next step, a pattern division will split the model into two parts according to the parting line, and add walls for the pattern tooling flask on each of them to provide a complete pair of cope and drag pattern model in triangle mesh form. Finally, a physical wood pattern can be manufactured using the Rapid Pattern Maker, a sand mold pulled from the wood pattern, and a re-cast product will be created.

The next sections will provide additional detail on the additional steps of the process, from point cloud generation through pattern model division.

3. 2 Point Cloud processing

The point cloud processing step allows us to analyze and edit point cloud based models. The point cloud model is the direct geometry information that can be acquired from a physical part. Properly handling this raw position information would provide a lot of convenience for later work. In this thesis, the point cloud processing work is based on the segmentation module and the sample consensus module of Point Cloud Library (PCL)[21].

A region growing segmentation method[24] is used in this thesis for the point cloud segmentation purpose. This algorithm begins with sorting all the points according to their curvature value from low to high. The curvature value of each point is calculated through a number of k nearest neighbor points or the neighbor points in a certain radius[25]. When all the points are sorted by their curvature value from low to high, the algorithm starts from the point with the lowest curvature value. The region grow process proceeds from this point as the seed point, which will be the seed point. First, a neighbor normal test process is conducted. When it finds that the angle of the neighbor point's normal between the normal of the seed point, this point is added to the region. Second, all the neighbor points that have a smaller curvature value than the threshold value are added to the seeds set for region growing. Third, the current seed is removed from the seeds set. Then, iterate this process on the rest of the seeds in the seeds set until the seeds set is empty. That is the complete steps for one region growing process. Now, for the rest of the points in the points set, find the point with the least curvature value and conduct the region growing process again, until there are no points left ungrouped. The result of point clouds segmentation is as figure 3.3 shows:

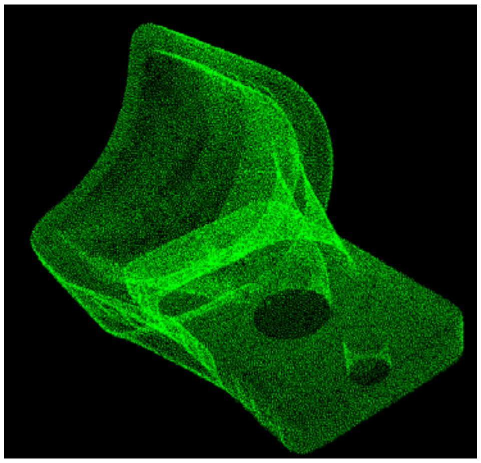


Figure 3.3(a). 3D Point Cloud

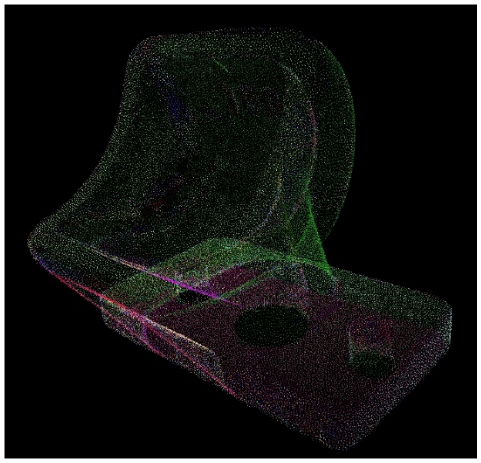


Figure 3.3(b). Segmented Point Cloud

Once the point cloud is segmented, each region of points forms a point cluster. Each point cluster can be individually analyzed and manipulated.

3. 2. 1 Feature Recognition

The next step for point cloud processing is feature recognition. Recognizing geometry features from the point cloud data of a physical part will allow us to isolate the design feature and further enable us to edit the feature. In the problem that is described in this thesis, what we are interested in is machining feature. There are different views of machining feature classification. From the machining point of view, X. Yan, et al classified all the machining features into five categories: Hole, Pocket, Open Pocket, Face and Boss[26]. Further decompose theses machining feature into geometry features, the basic geometry feature for machining are: Plane, hole, free form surfaces and the combination of them. Free form surfaces on a Near Net Shaping part are those surface feature we want to maintain, so the

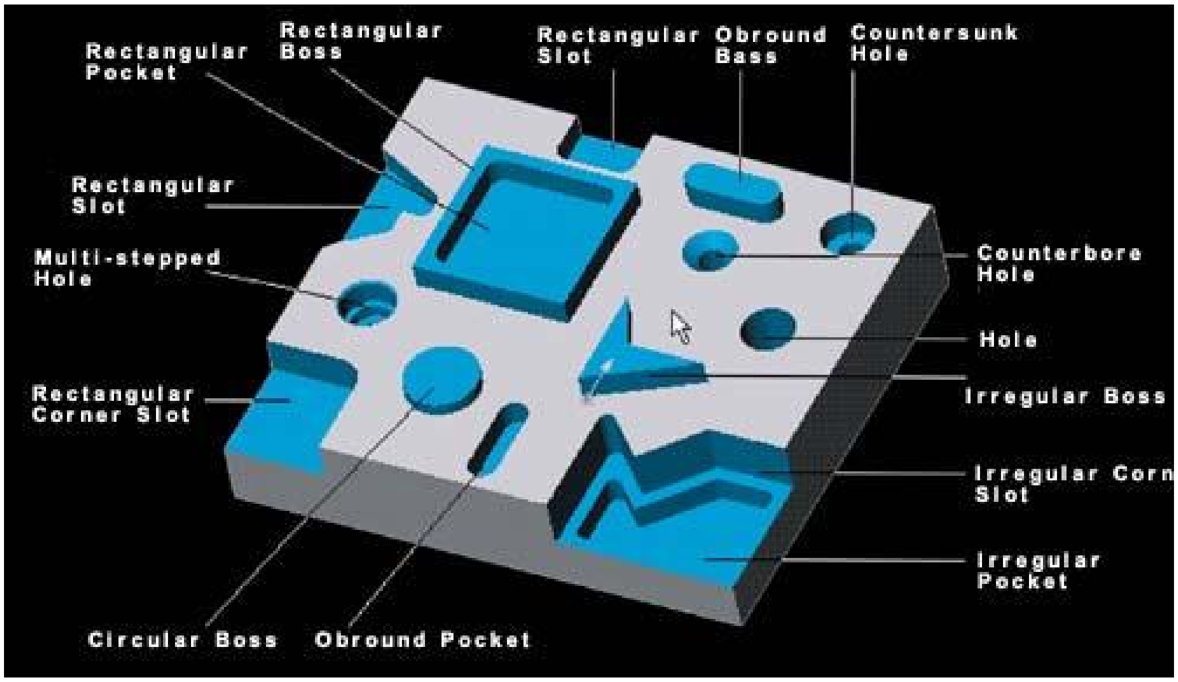


Figure 3.4. Machining Feature Classification[36]

basic geometry features that needed to be analyzed are planes and holes. In this thesis work, feature recognition will be limited to planes, cylinder holes.

A Random Sample Consensus (RANSAC) method is implemented in PCL origins from Fischler and Bolles's[18] algorithm. This algorithm works by selecting a random subset of the point clouds, and fitting the subset onto a hypothetical model (geometry primitives as plane, cylinder, sphere and so on). Take a plane model as an example. For every subset it tested, all specifications of the hypothetical plane (Four numbers of the Hessian normal form defines a plane, two points and a radius value will define a cylinder, one point and a radius value will define a sphere) will be calculated out from the subset. An estimated error of the hypothetical model and all the point data will be calculated to evaluate this hypothetical plane. After an assigned number of iterations of this random subset and error evaluation, the best fitted hypothetical plane will be output as the best fit plane.

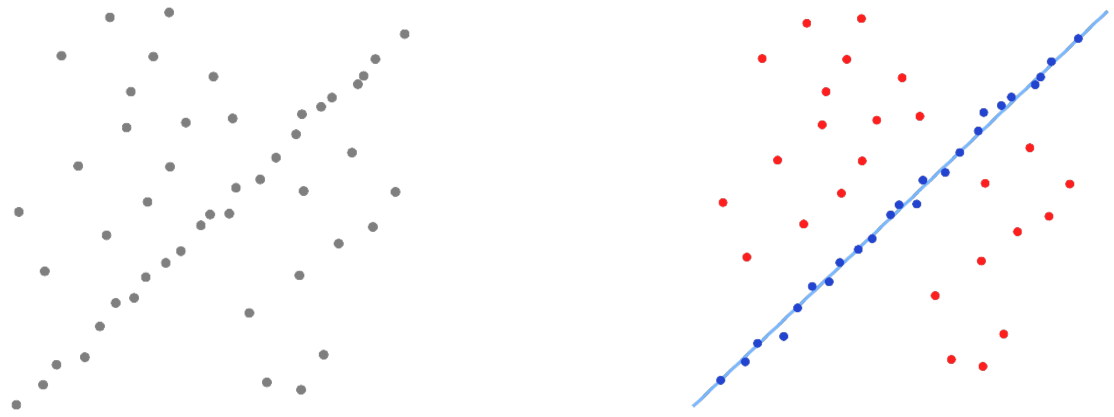


Figure 3.5. RANSAC shape recognition[37]

Based on this RANSAC algorithm, a method for automatically recognizing geometry feature is developed in this thesis work. A ransac model base of plane, cylinder and etc. has already been created in PCL. For a point cluster, a best fit plane and a best fit cylinder can be calculated through the RANSAC algorithm. Then the model with more number of inlier points will be chosen as the best fit shape. As the ransac model base is developing, more and more primitive shape will be included, and more geometry feature can be recognized through this method.

From the example part, machining features can be recognized and parametrized.

Cylinder (mm)	
X1	80.4731
Y1	16.1073
Z1	-19.5884
X2	44.3296
Y2	15.3691
Z2	-73.3384
R	45.8197

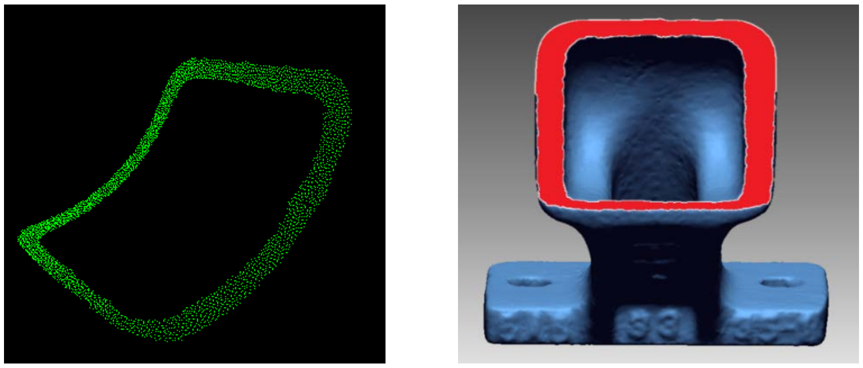


Figure 3.6(a). Machining Feature (Cylinder Arc)

Cylinder (mm)	
X1	-38.1775
Y1	-10.2774
Z1	-20.7793
X2	-47.0802
Y2	-6.91848
Z2	-15.2367
R	6.12927

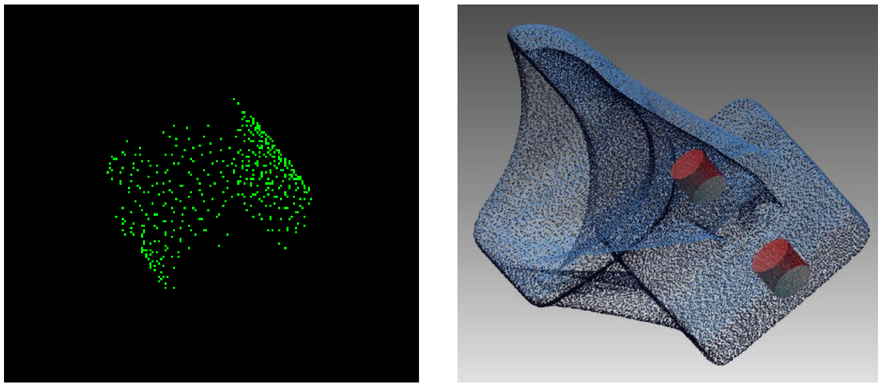


Figure 3.6(b). Machining Feature (Cylinder Hole)

Plane (mm)	
Nx	0.768179
Ny	-0.388573
Nz	-0.508834
D	26.3663

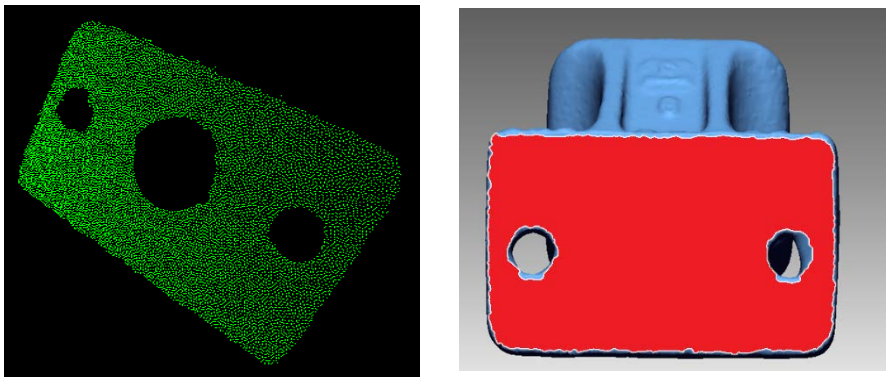


Figure 3.6(c). Machining Feature (Plane)

3. 2. 2 Machined Feature Geometry Modification

Having the parametrized geometry feature allows us to manipulate those features. For machined surfaces, an offset operation can be applied to add machining allowance. The offset operation is different for planes and cylinders. For a plane or a large arc cylinder surface, the offset is in two steps. First, offset points according to its normal. Then, offset the boundary point of the point cluster. The second step is for filling the gap that will be created from the offsetting of the plane point cluster.

The normal of a point in the point cloud is determined by estimating the normal of a prediction plane that tangent to the surface at this point[27]. The offset step is by moving the point along its normal for the input offset value.

Through a boundary point finding function that is detailed in Radu Rusu's thesis[27], the boundary point of a point cluster can be found. Then offset the boundary point as the figure 3. 8 for gap filling.

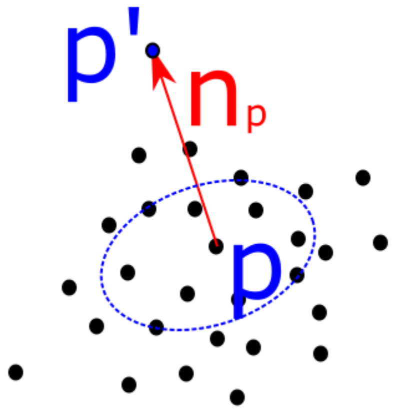


Figure 3.7. Point offset

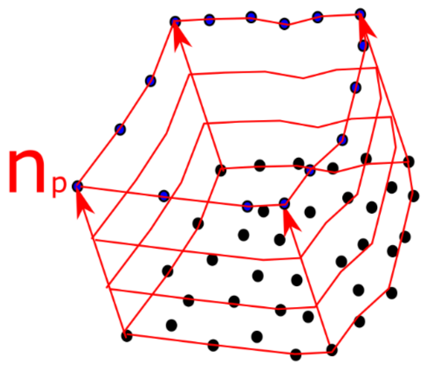


Figure 3.8. Boundary point offset

For a drilled, cylindrical hole, there are two possibilities of how this hole can be created. The first possibility is, this hole is first cast with a smaller diameter and then drilled to the "as-is" machined hole. The second possibility is this that hole is not created via casting, it is directly drilled to the "as-is" shape. For the first situation, an offset operation should be applied since this hole needs to be cast with a smaller diameter. The offset operation is implemented by recreating this hole geometry from the specification of this cylindrical hole. For the second situation, this hole is directly machined, so this geometry should be deleted from the casting geometry. A circle cap will be generated to cover this hole.

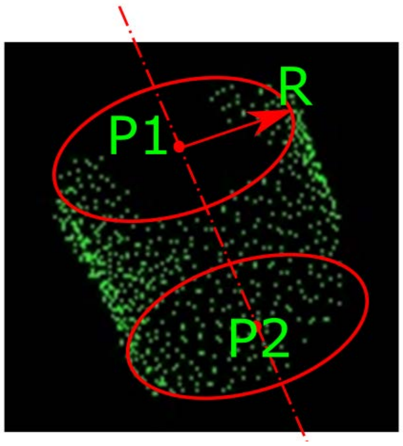


Figure 3.9(a). Point Cloud of Cylinder Hole

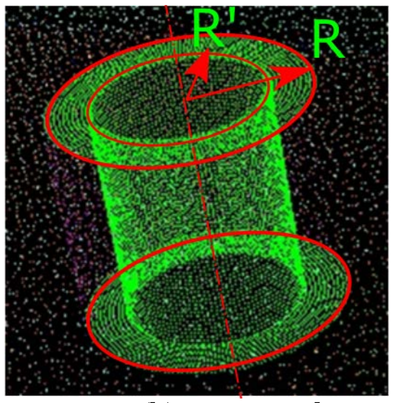


Figure 3.9(b). Recreated Point Cloud of Casted then Machined Cylinder Hole

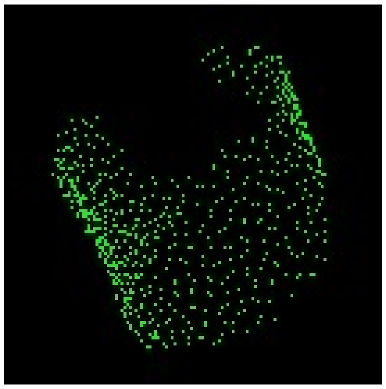


Figure 3.10(a) Point Cloud of Cylinder Hole

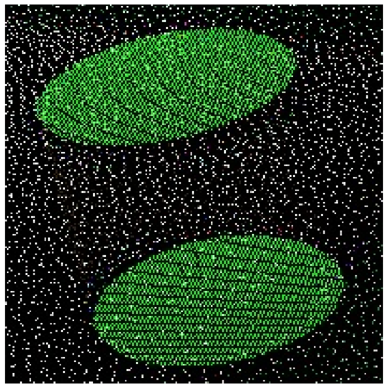


Figure 3.10(b). Recreated Cap Point Cloud for Machined Only Cylinder Hole

The two points P1, P2 and the radius value R that defines this cylinder can be calculated as has been introduced before in the feature recognition part. To recreate the cast then machined cylindrical hole as figure 3.9(b) shows, it needs to be detailed in two steps:

First, create the cylindrical surface. This newly created cylindrical surface is the offset surface from the original surface, so the radius of the new cylinder surface is: $R' = (R - offset\ value)$. Set the angle step to be θd and height step to be d to control the point's density, then the created points can be calculated as:

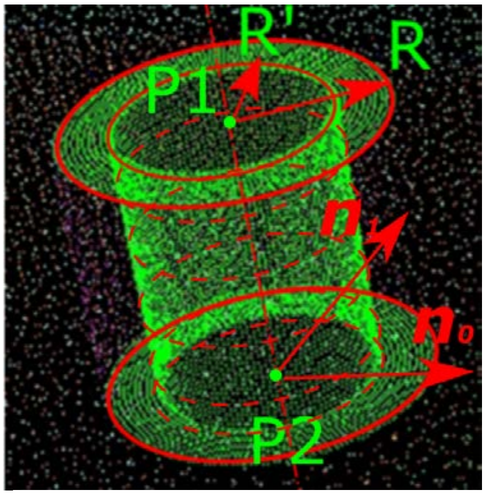


Figure 3.11. Create point on cylinder side surface

$$\boldsymbol{p_{i,j}} = \boldsymbol{p_2} + R' \cdot (\boldsymbol{n_0} \cdot \sin(i \cdot \theta_d) + \boldsymbol{n_1} \cdot \cos(i \cdot \theta_d)) + \frac{\boldsymbol{P1-P2}}{|\boldsymbol{P1-P2}|} \cdot j \cdot d$$

Here, $\boldsymbol{n_0} = \frac{\boldsymbol{P1\times P2}}{|\boldsymbol{P1\times P2}|} \times \frac{\boldsymbol{P1-P2}}{|\boldsymbol{P1-P2}|}, \boldsymbol{n_1} = \frac{\boldsymbol{P1\times P2}}{|\boldsymbol{P1\times P2}|}, i, j \in N\ and\ i\epsilon\left[0, \frac{2\pi}{\theta_d}\right), j\epsilon[0, \frac{|\boldsymbol{P1-P2}|}{d}]$

Second, create the circular ring as figure 3.11 shows.

$$\boldsymbol{p_{i,j}} = \boldsymbol{p_1} + \boldsymbol{n_0} \cdot i \cdot d + \boldsymbol{n_1} \cdot j \cdot d$$

or $\boldsymbol{p_2} + \boldsymbol{n_0} \cdot i \cdot d + \boldsymbol{n_1} \cdot j \cdot d$

Here, $i\epsilon Z\ and\ i\epsilon\left[-\frac{R}{d}, \frac{R}{d}\right], j \in Z\ and\ j \in$

$[\frac{\sqrt{R'^2-(i\cdot d)^2}}{d}, \frac{\sqrt{R^2-(i\cdot d)^2}}{d}] \cup [-\frac{\sqrt{R^2-(i\cdot d)^2}}{d}, -\frac{\sqrt{R'^2-(i\cdot d)^2}}{d}]$

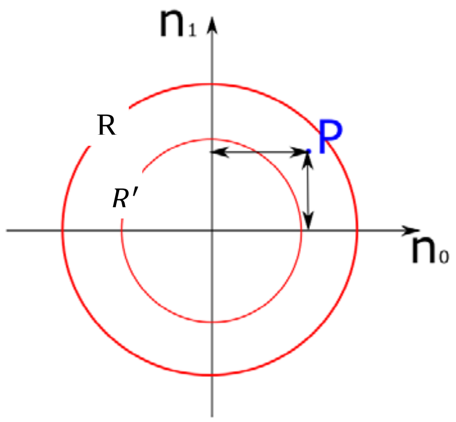


Figure 3.12. Create point on the circle ring

The way to create the cylindrical cap to fill the machined only holes as figure 3.10(b) shows is the same as creating the circle ring, just need to replace R' with 0, and limit j to this range:

$$j \in [-\frac{\sqrt{R^2 - (i \cdot d)^2}}{d}, \frac{\sqrt{R^2 - (i \cdot d)^2}}{d}]$$

3. 3 Adding draft on non-feature based triangle mesh.

The objective is to add draft on those facets that are parallel or nearly parallel to the parting direction. Adjusting the normal of each facet of a triangle mesh is conducted in three steps: the first step, check the normal of each facet and decide which facets that need to be adjusted normal according to the criteria of casting pattern draft. Then, the second step, setting the normal vector by assigning a new normal vector that will meet the pattern draft criteria to this facet. The new normal vector is not the geometry normal of this facet for now, and that's why the vertex update step is needed. The third step, updates the vertex position according to the new normal vector of the facet and the vertex position of its neighboring facets and make sure the triangle mesh is complete and correct.

Here is the detail of how all these three steps are conducted. First step, checking the normal of each facet. Suppose the parting direction is along Z axis, represented by vector $n_{\text{axis-z}} = (0,0,1)$. Suppose the criteria angle of pattern draft is θ_d.The normal vector of target facet is $n = (n_x, n_y, n_z)$. The angle between facet normal vector and x-y plane can be calculated as:

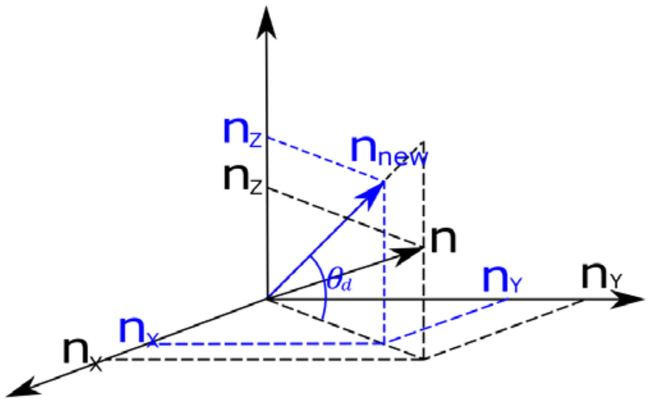


Figure 3.13(a). 3D view of new normal calculation

$$\theta = 90°-\cos^{-1}\frac{n_{\text{axis-z}} \cdot n}{|n_{\text{axis-z}} \cdot n|}.$$

If $\theta < \theta_d$, it means the draft on this facet is not enough.

Then the second step, assign the normal vector of this facet to a new vector n_{new}. n_{new} is calculated by changing the n_z portion of this vector. $n_{new-x} = 1 \cdot \cos\theta_d \times \frac{n_x}{\sqrt[2]{n_x{}^2+n_y{}^2}}$, $n_{new-y} = 1 \cdot \cos\theta_d \times \frac{n_y}{\sqrt[2]{n_x{}^2+n_y{}^2}}$, $n_{new-z} = 1 \cdot \sin\theta_d$.

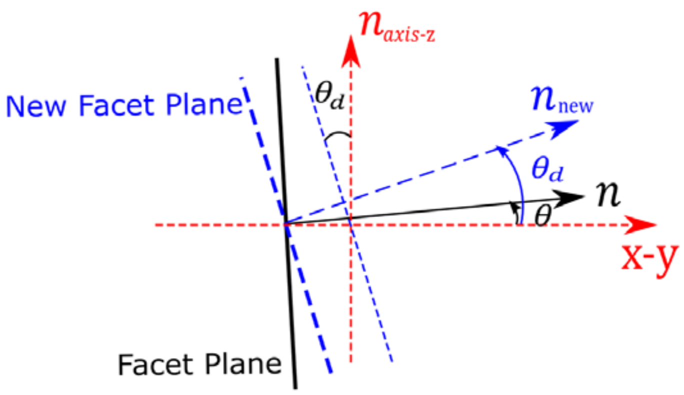


Figure 3.13(b).2D view of new normal calculation

The third step, vertex updating. In the approach that described in S. Xianfang, P. L. Rosin, R. R. Martin, and F. C. Langbein's work[28], this function below is applied for vertex updating.

$$x'_i = x_i + \frac{1}{|F_v(i)|} \sum_{k \in F_v(i)} \boldsymbol{n'_k}(\boldsymbol{n'_k} \cdot (c_k - x_i))$$

In this function, $x_i$represents the original vertex, x'_irepresents the vertex after vertex updating, $F_v(i)$represents the number of facets that are around the target vertex, $\boldsymbol{n'_k}$represents the new normal that is assigned to the facet, and $c_k$represent the center point of the facet.

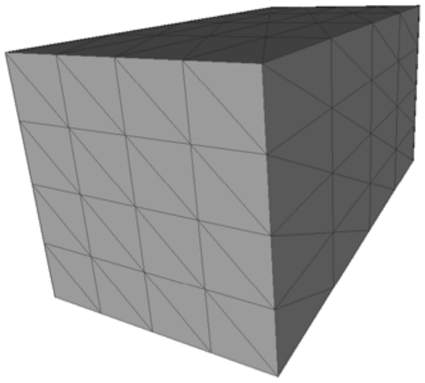


Figure 3.14a). Cube STL model

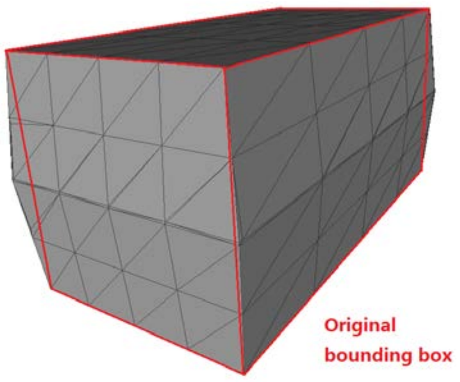


Figure 3.14 (b). Cube STL model adding draft

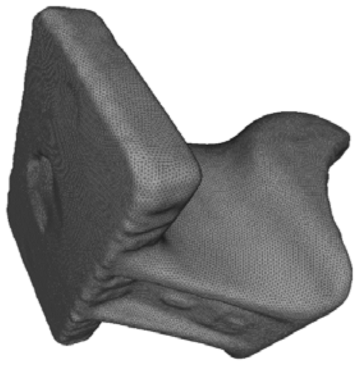


Figure 3.15(a). Demo STL model

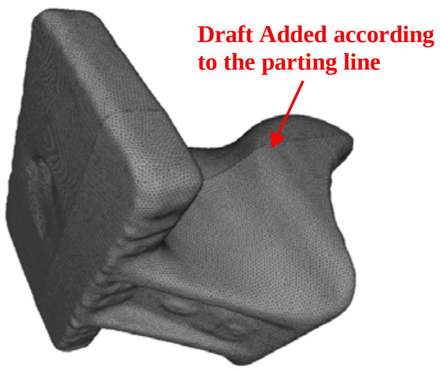


Figure 3.15(b). Demo STL model adding draft

Although this approach is not the perfect way to do vertex updating, it is very efficient. After the chosen number of iterations, the result is satisfies.

3. 4 Pattern Model Division trough Triangle Mesh Divide Algorithm

Triangle mesh, represented in STL format is the most popular format for feature free geometry representing and rapid prototyping. The whole triangle mesh is loaded into the OpenMesh half-edge data structure[29]. All the method and algorithm described is based on the half-edge data structure.

The objective of this polygon mesh divide algorithm is to divide one polygon mesh into two separate polygon meshes according the parting line. Parting line here is a user defined polyline. In Cartesian coordinate system, assuming the dividing direction is along Z axis, then the polyline parting line is defined in X-Z plane. In order to divide the triangle mesh, and maintain the triangle mesh structure after one mesh is divided into two part, the whole parting process is in three steps. First, parting line alignment; Second, data transfer and store in two separate part; Third, Open boundary triangulation.

3. 4. 1 Preparation work for mesh divide

The purpose of this process is to partly re-triangulate the facets that intersected with the parting line by creating new faces, edges and vertices. So the new triangle mesh will have edges that exactly coincide with the parting line.

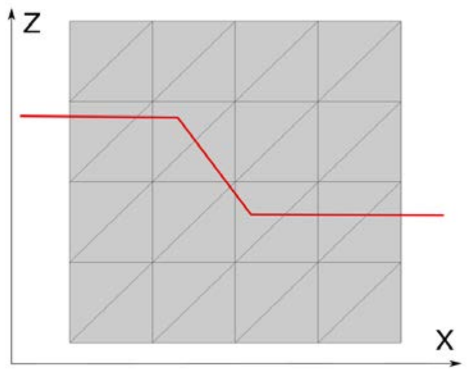


Figure 3.16(a). 2D view of parting line

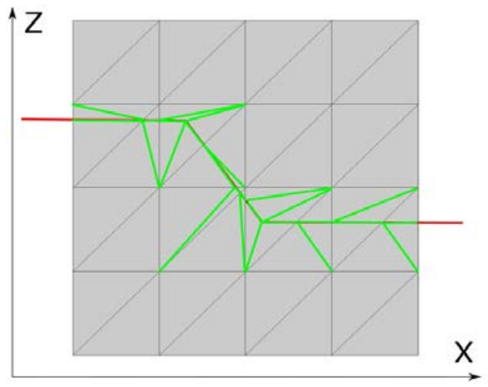


Figure 3.16(b). 2D view of parting line partially re-triangulated

3. 4. 1. 1 Intersection categorize

The polyline parting line is made up of two components: lines and turning points.

For lines of the parting line that intersect with edges of the triangle mesh, it's the problem of two lines intersection.

Two lines can intersect in three different ways: intersect in the middle, intersect on one line's end point and intersect on both lines' end points.

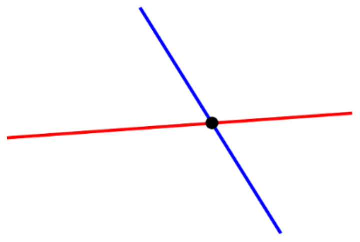


Figure 3.17(a). Type 1 Line to Line intersection

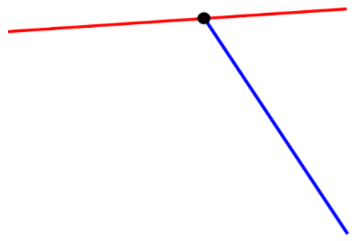


Figure 3.17(b). Type 2 Line to Line intersection

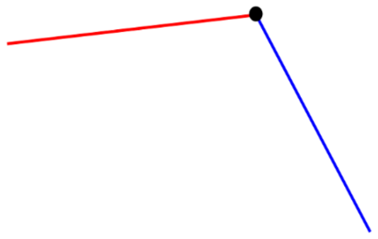


Figure 3.17(c). Type 3 Line to Line intersection

Also they can have two “intersect” points in the flowing three different ways:

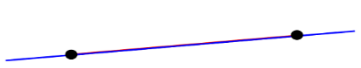


Figure 3.17(d). Type 4
Line to Line intersection

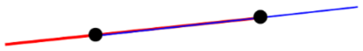


Figure 3.17(e). Type 5
Line to Line intersection

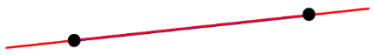


Figure 3.17(f). Type 6
Line to Line intersection

For the turning points of the parting line, its’ relationship with the triangle mesh can be classified into three types: lies on the vertex, lies on the edge, and lies within a triangle.

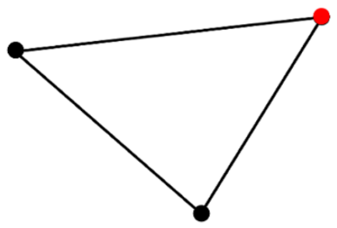


Figure 3.18(a) Type 1
Point to Triangle intersection

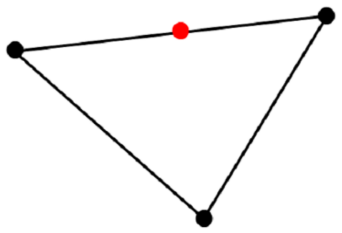


Figure 3.18(b) Type 2
Point to Triangle intersection

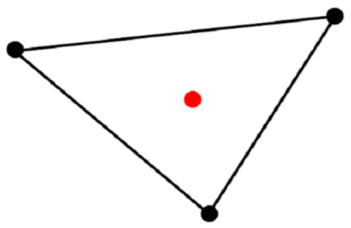


Figure 3.18(c) Type 3
Point to Triangle intersection

3. 4. 1. 2 Discussion of all types of intersection

The intersection of parting line and triangle mesh creates new points. Starting from those new points and linking with the neighbor points to create new edges and facet, that’s the strategy of partially re-triangulation.

For the 6 types of intersection between parting line and edges of triangle mesh, the 6 types will be discussed separately on how to create new points, edges and facets.

Type 1, intersect in the middle. This type creates new vertex through intersection, and it changes the structure of the triangle mesh. This type will be discussed in detail.

Type 2 and Type 3 interested at one end of the parting line then the scenario becomes the same as the first two types of intersection of turning points and triangles: lies on the vertex and lies on the edge. More discuss of Type 2, if one of the triangle vertex lies on the middle of a parting line, then this type of intersection does not create new vertex, so this situation can be ignored.

For two intersect points situation. Type 1, triangle edge lies within the parting line. This type doesn’t create new vertex, so it can be ignored. Type 2, triangle edge and parting line overlays partly, this situation is a combination of intersect in the middle and intersect at the end of a triangle edge. As have been discussed before, intersect at the end of a triangle edge can be ignored, so this situation will be treated as same as Type 1 line and edge intersection (intersect in the middle). Type 3, parting line lies within the triangle edge. This situation should not happen, because comparing to the parting line the length of a triangle edge is too small, this situation should be ignored.

For the 3 types of intersection between turning point and triangle mesh. Type 1 lies on the vertex, this type can be ignored because it doesn’t create new vertex. Type 2 and Type 3 creates new point. This two type will be discussed in detail.

3. 4. 1. 3 Intersection Re-Triangulation

Based on the discussion of all types of intersection, three types out of nine should be studied: Line to edge intersection in the middle, point to triangle intersection on the edge and point to triangle intersection within the triangle.

The next question would be how to re-triangulate the intersected part. This question will be separately discussed for all the three types.

Re-triangulate: Line to edge intersection in the middle. This type can be represented as the figure below: black line represent the edge of the original mesh, red line represent the parting line, red dot represent the intersected point, blue line represent the new edge created for re-triangulation.

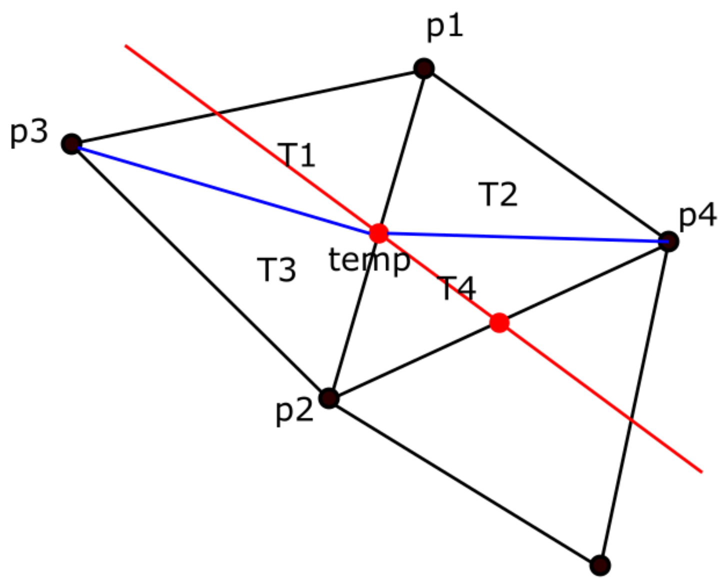


Figure 3.19. Re-Triangulate: Line to edge intersection in the middle

This process can be done in five steps:

First, calculate the intersect point; the intersection is happening in the X-Z projection plane. So only the X and Z coordinate should be counted. This problem actually is line to polyline intersection problem. The polyline can be considered as several connected lines. First thing to be determined is the two end point of an edge falls into which lines X coordinate range. This is done through X coordinate comparison. If end point's X coordinate is bigger or equal than line one's minimum X coordinate and smaller than line one's maximum X coordinate then this end point falls onto line one's region. Then decide whether this point is above or below the line through an ABTest function. This ABTest function can be described as the figure below.

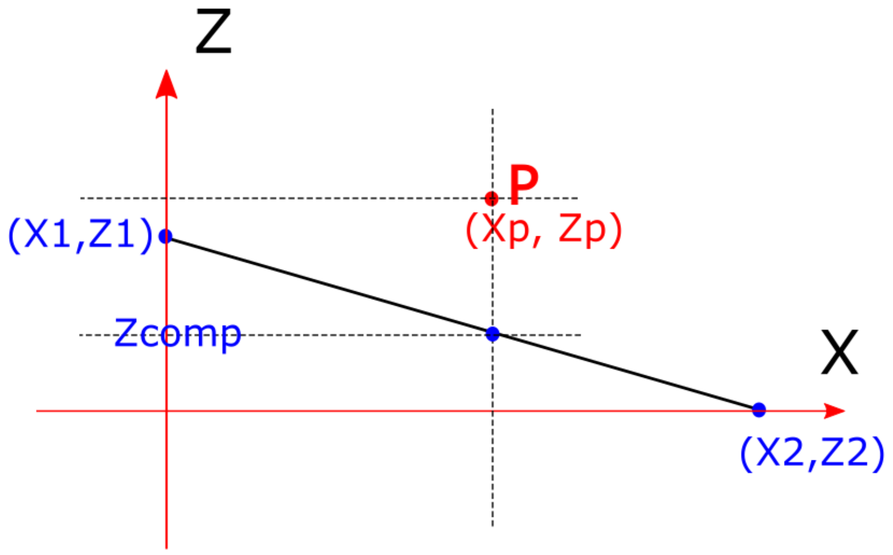


Figure 3.20. Above or below test (ABTest function)

Edge end point P has X coordinate Xp, and Z coordinate Zp. The line has minimum X coordinate X1 and maximum coordinate X2. According to the X2-Xp and Xp-X1 ratio, projection point of P on the line can be found, and it's Z coordinate would be $Zcomp =$

$Z1 + \frac{Xp-X1}{X2-X1} \times (Z2 - Z1)$. Comparing Zp to Zcomp. If Zp is bigger than Z comp then P is above the line. If Zp is smaller than Z comp then P is below the line.

If the one end point of an edge is above the parting line and the other is below the parting line, then there will be an intersection point. The coordinate of the intersecting point can be calculated.

Second, add the point to the mesh data; Third, find the two opposite vertices of the intersect edge; Fourth, delete the two facets corresponding to the intersect edge; Fifth, add four new triangles according to the related four vertices and the new point.

Re -triangulate: Point to triangle intersection on the edge. This situation is slightly different from the situation of intersection in the middle. Since the turning point is the joint point of two part of a polyline parting line. When calculating the intersection point, the edge would intersect with both lines. To avoid redundant calculation, each line is treated as a one

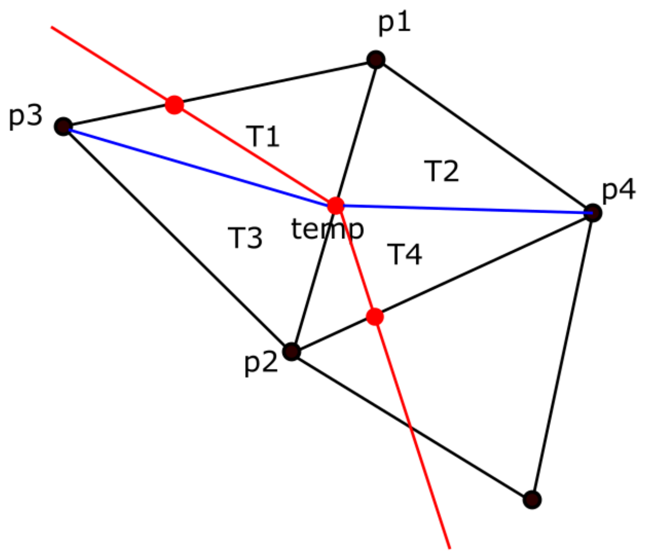


Figure 3.21.Re-Triangulate: Point to triangle intersection on the edge

end line. Only the start point will be counted as part of the line. The rest of this situation should be treated the same as Line to edge intersection in the middle.

Re-triangulate: Point to triangle intersection in the middle. This type can be represented as the figure shown below. The re-triangulation progress can be done in five steps:

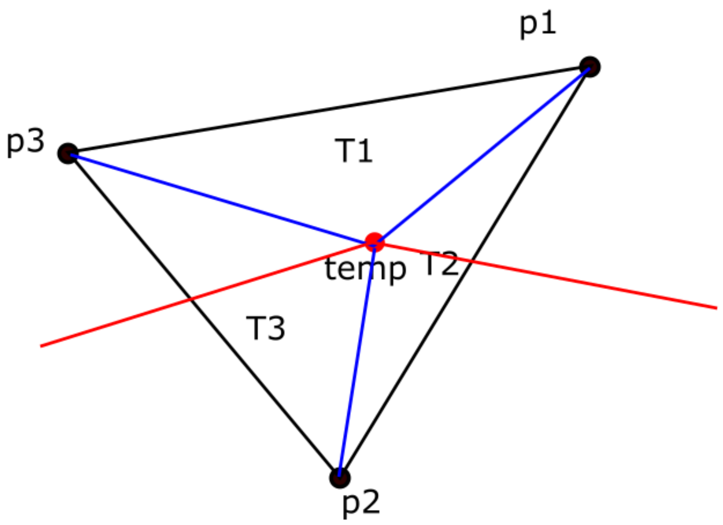


Figure 3.22. Re-Triangulate: Point to triangle intersection in the middle

First, calculate the intersect point[30]; This problem is a point projection problem, W. Heidrich uses Barycentric Coordinates to represent the projection point. Here this method is used to determine if a point can be projected within a triangle or not, and if it can, calculate the position in Cartesian Coordinates. In Heidrich's method the triangle is represented as the following figure: assuming the triangle is given by a point q and two vector *v* and *u*, P' is the projection point of P onto this triangle. Then the Barycentric Coordinates of P' can be represented by this formula:

$$Let\ \vec{w} = \boldsymbol{P} - \boldsymbol{q}\ , \vec{n} = \vec{u} \times \vec{v}$$

$$b_2 = [(\vec{u} \times \vec{w}) \cdot \vec{n}]/\vec{n}^2$$

$$b_1 = [(\vec{w} \times \vec{v}) \cdot \vec{n}]/\vec{n}^2$$

$$b_0 = 1 - b_2 - b_1$$

In this function if: $0 < b_0 < 1, 0 < b_0 < 1$ *and* $0 < b_0 < 1$, then the projection point P' lies within the triangle.

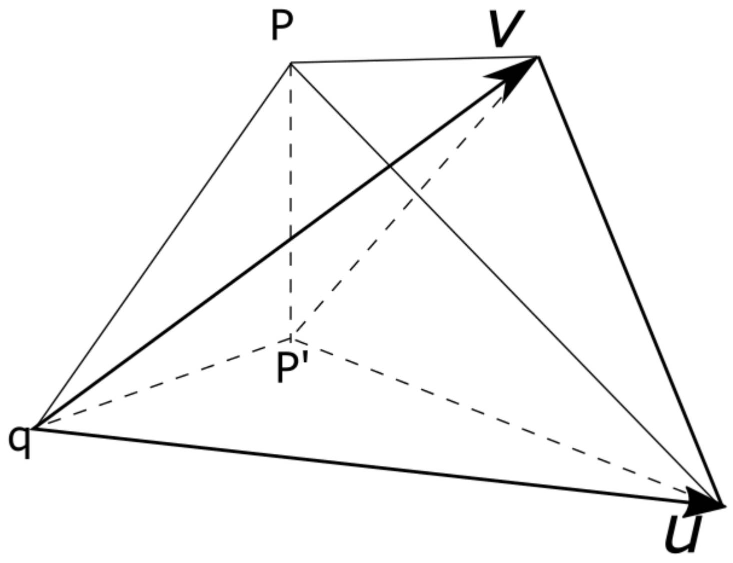


Figure 3.23. Hendrich's method for point to triangle projection

In Cartesian Coordinate the position of P' is:

$$P' = b_0 q + b_1(q + \vec{u}) + b_2(q + \vec{v})$$

Second, add the new point to the mesh data structure; third, find the three vertex of the intersected triangle; fourth, delete the current facet; fifth, create the three new triangle according to the new point and three vertices.

3. 4. 2 Divide and boundary re-mesh

In order to separate the "as was cast" model to the two half of polygon mesh model according to the parting line, the first thing need to be done is determining which triangle mesh belongs to the cope pattern and which belongs to the drag pattern. Using the ABTest that mentioned before in this chapter to determine whether the centroid of a triangle mesh is above or below the parting line. If the centroid of the triangle mesh is above the parting line, this triangle is stored to the data structure of the cope pattern mesh, otherwise it is stored to the data structure of the drag pattern mesh.

When this divide process is done, for both of the polygon mesh of the cope and drag pattern, there is an open boundary as this figure showed below. To get a complete water tight polygon mesh, a boundary re-mesh process is needed to be done. An ear-clipping algorithm[31] is applied. Ear-clipping algorithm is a triangulation method for simple or non-simple polygon in 2D space. In this situation, the whole set of boundary points make up a 3D polygon, and this polygon is completely visible in Z direction. The projection of this boundary polygon onto X-Y plane is simply just consider the X and Y coordinate of each point. That makes this 3D polygon a 2D polygon and ear-clipping algorithm can be applied to triangulate this 2D polygon. When this 2D polygon is triangulated, map it back to the 3D polygon to have the triangulated boundary.

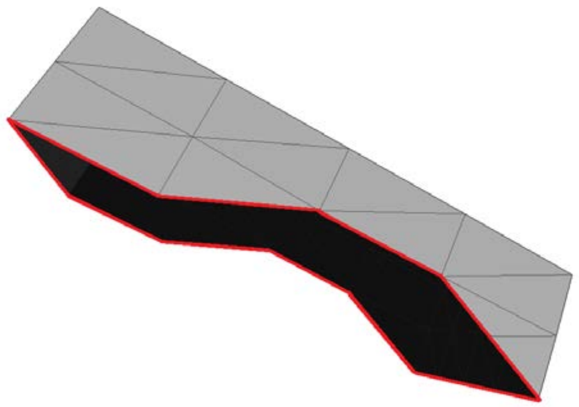


Figure 3.24(a). STL model Divided along parting line

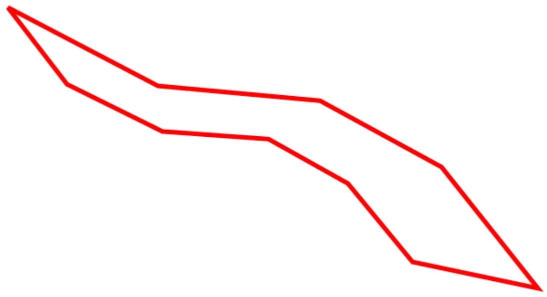


Figure 3.24(b). Boundary Poly lines

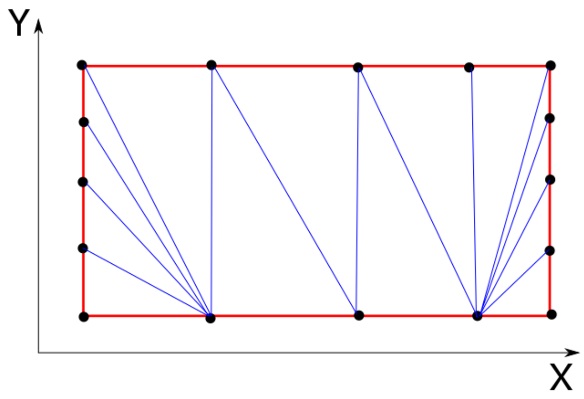


Figure 3.24(c). Boundary Polyline projection onto X-Y plane and ear clipping triangulation

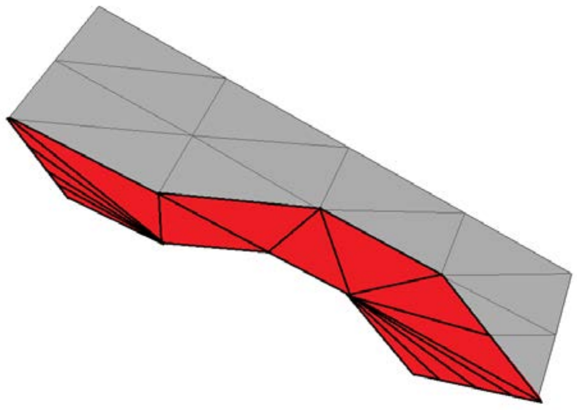


Figure 3.24(d). Map boundary triangulation back to the STL model

Now, the basic tooling geometry for cope and drag pattern is generated.

CHAPTER 4. METHOD IMPLEMENTATION AND DEMO ANALYSIS

The whole method for assisting reverse engineering legacy casting part is implemented through C++ program and be delivered in two pieces of software. The first piece of software implements the point cloud processing part named RECast that accomplish the machining feature editing function and output as point cloud model. A commercialized software *Geomagic Design X*[32] is used to tessellate the point cloud model into triangulated mesh model as the input for the second piece of software. The second piece of software named NNSEditor implements the draft adding and pattern dividing part.

4. 1 RECast (Reverse-Engineered Casting)

The user interface is shown as below.

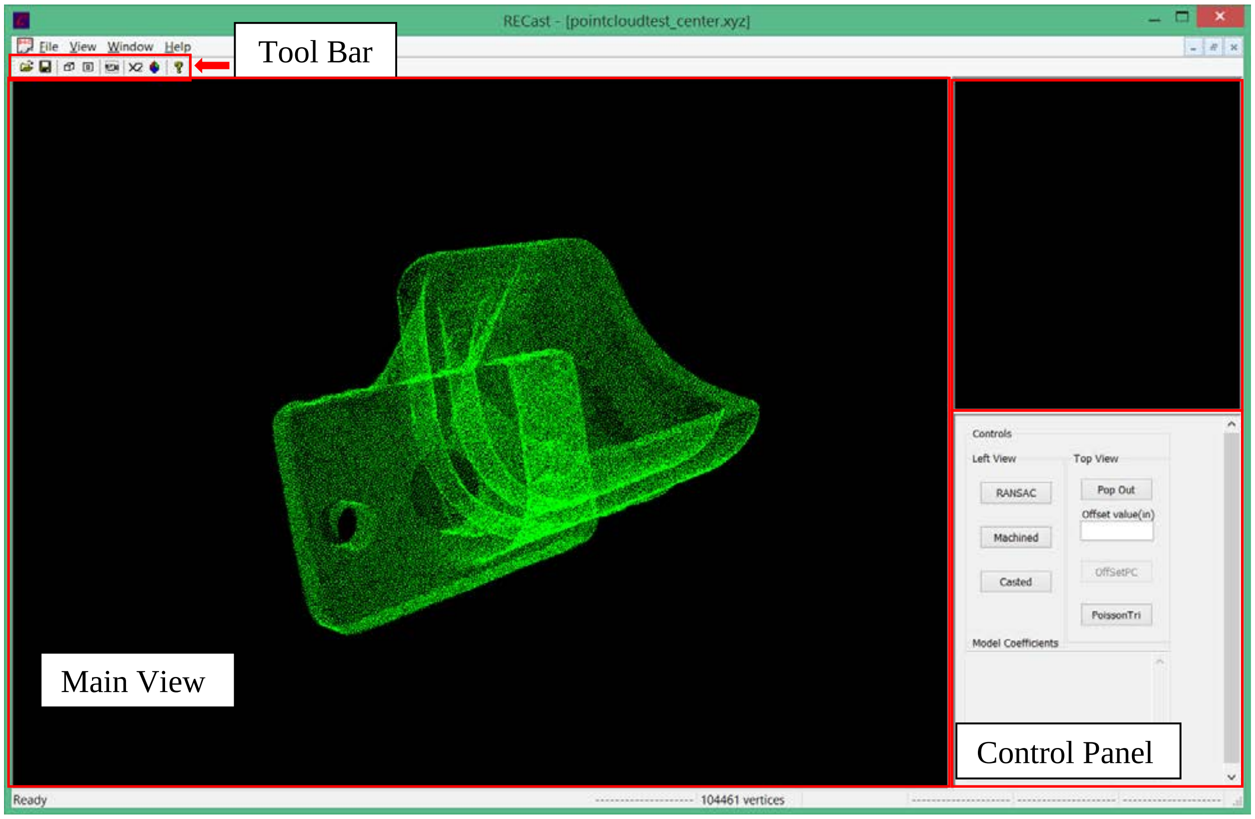


Figure 4.1 RECast software User Interface

The RECast UI consists of four components; 1) a tool bar, 2) the main view, 3) second-view and 4) a control panel. The tool bar integrates the functions of view control and the segmentation function while the main view shows the point cloud for processing.   The second-view shows the point cloud cluster that is selected by the user and the control panel integrates the RANSAC and offset functions.

The entire process of data flow is shown graphically in Figure 4.1.

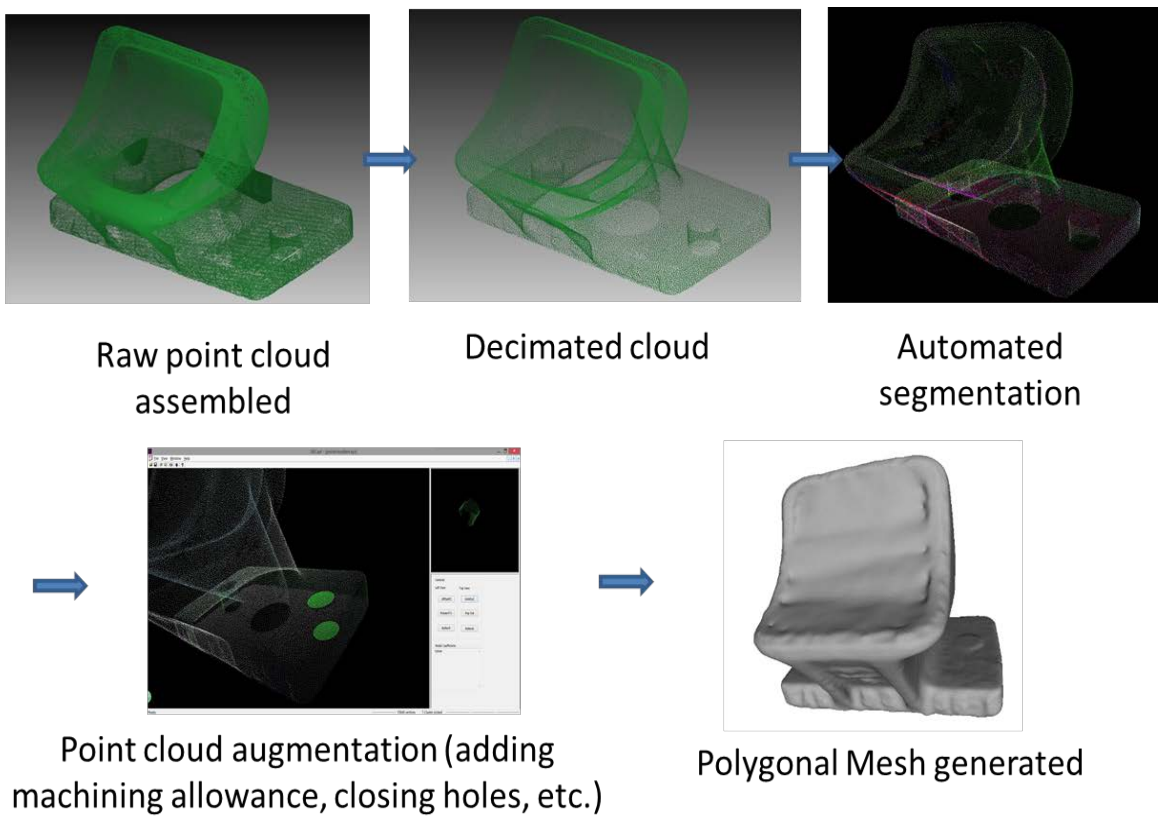
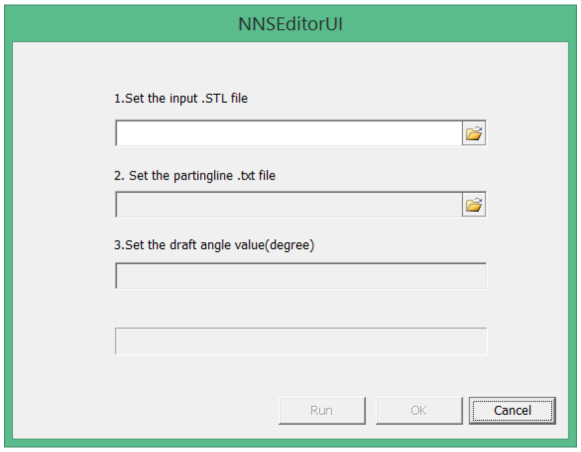

Raw point cloud assembled
Decimated cloud
Automated segmentation
Point cloud augmentation (adding machining allowance, closing holes, etc.)
Polygonal Mesh generated

Figure 4.2. Process flow of RECast

As the process flow shows, when the software takes in the point cloud, the user will click the segmentation button on the tool bar. The whole point cloud will be segmented and allow the user to pick each point cluster to analyze or edit it. For cylinder machined feature and plane machined feature the user can input the offset value on the control panel Offset value(in) to offset the machined surface. For a cylinder hole, if the user decide this hole to be machined only Machined button shall be clicked, and as have described before a cap will be created to cover this hole. Or if the user decide this hole to be casted and then machined Casted shall be clicked, and this hole will be offset by a defined value.

4. 2 NNSEditor (Near Net Shaping Editor)

The user interface for NNSEditor is shown as below.


NNSEditorUI

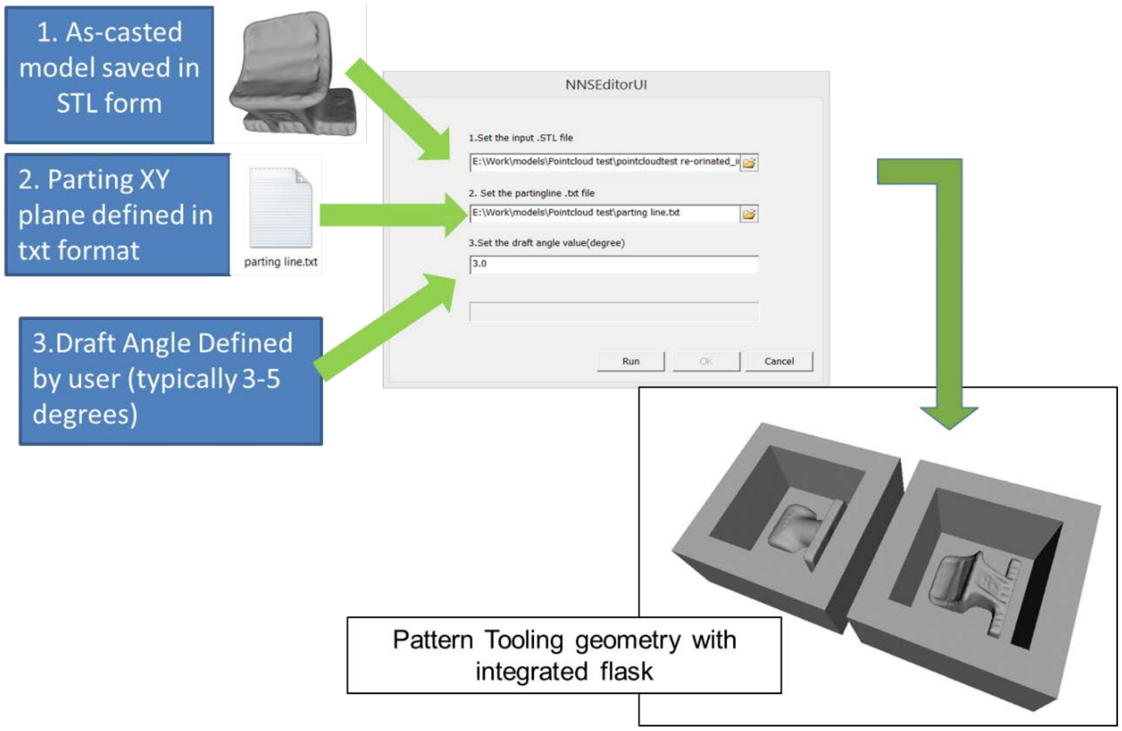
1.Set the input .STL file

2. Set the partingline .txt file

3.Set the draft angle value(degree)

Figure 4.3. User Interface of NNSEditor

This software program takes in the triangulated polygon mesh that generates from the point cloud result in .STL format. Then takes a user defined parting line points coordinates in .txt format. After defining the parting line, set up the draft angle and click “Run” button, it will automatically generates the pattern model.


NNSEditorUI

1.Set the input .STL file: E:\Work\models\Pointcloud test\pointcloudtest re-orinated_i

2. Set the partingline .txt file: E:\Work\models\Pointcloud test\parting line.txt

3.Set the draft angle value(degree): 3.0

1. As-casted model saved in STL form
2. Parting XY plane defined in txt format
parting line.txt
3.Draft Angle Defined by user (typically 3-5 degrees)
Run
OK
Cancel
Pattern Tooling geometry with integrated flask

Figure 4.4. Inputs and Outputs of NNSEditor

4. 3 Output model deviation study

The output model deviation analysis uses the scanned polygon mesh of the “as-is” part as the reference geometry. The surface deviation study is between the output polygon mesh with the reference geometry.

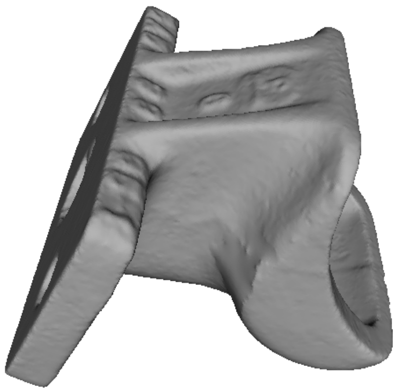


Figure 4.5(a). Reference Geometry: Triangle mesh of original scanned model

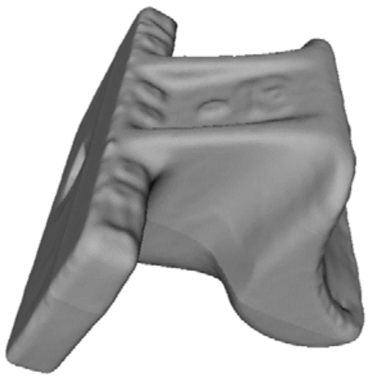


Figure 4.5(b). Compared Geometry: Triangle mesh of output “as-was” model

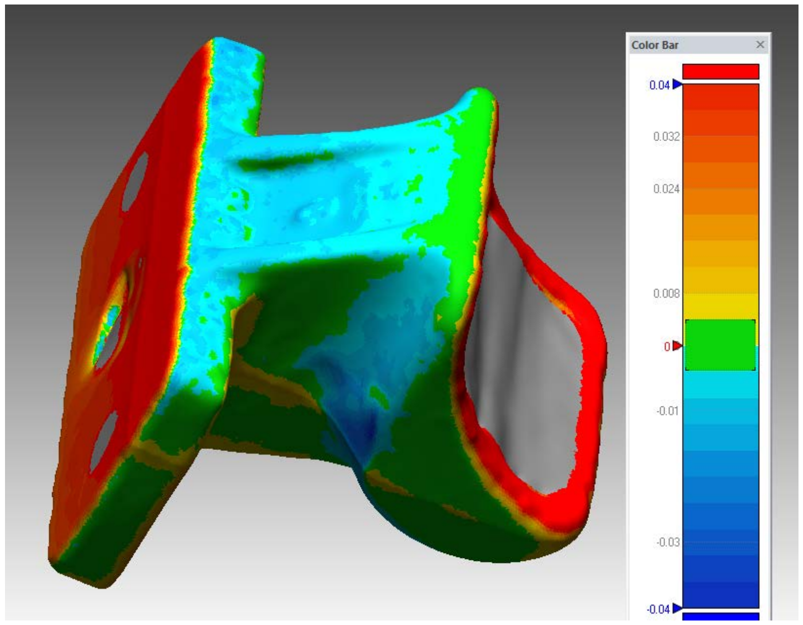
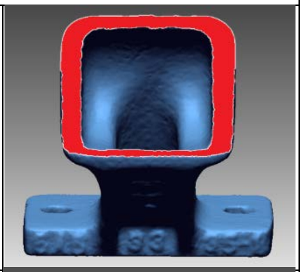
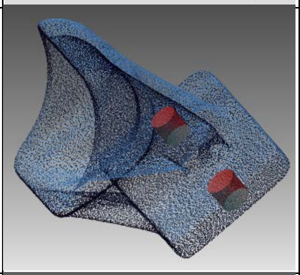
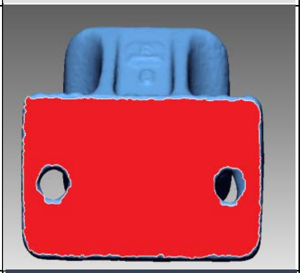
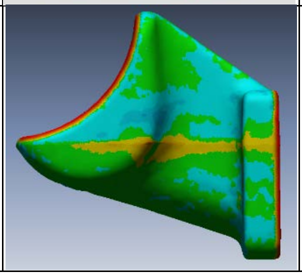


Figure 4.5(c). Surface Deviation Analysis result created from Geomagic Design X64 (Unit:inch)

As the colored map of the surface deviation shows in Figure(c), the red colored area are those surfaces offseted and added machining allowance, green colored area are casted surfaces that no changes has been made to. Along the parting line area, the color is gradually shifted from green to yellow, which exactly fits what is anticipated for the draft adding process. To show the deviation result in more detail, the table below shows the changes feature by feature. All the value is roughly measured from the deviation map legend.

Table 4.1. Surface Deviation Analysis by feature (Unit: inch)

Features		Prescribed	Measured
Cylinder Machining feature		Offset 0.05	Above 0.04+
Cylinder Hole feature		Machined hole to be removed	Machined Hole Removed
Plane Machining feature		Offset 0.05	Above 0.04+
Draft		Draft angle 3 degree	Roughly 0.4~ 3.4degree $\tan^{-1}\frac{0.01}{1.5}$ ~$\tan^{-1}\frac{0.03}{0.5}$

From table 4.1, we can tell that for cylinder machining feature and planar machining feature it gives a result of above 0.04, which is close to 0.05 the value for offset. Considering the accuracy of this measuring method and the offset process, this result is reasonable. As has been explained in chapter 3 method part, the offset of these machining feature is based on the normal calculated from each points’ one ring neighbor, so the normal is not perfectly accurate. The point cloud of a planar surface cannot be perfectly in a plane due to the accuracy limit of the laser scan. Same for the cylinder machining feature. So this result is satisfying and meet the expectation. For cylinder hole feature, it is removed as expected. For draft feature, the measured draft angle is ranges from 0.4 to 3.4 degree. This two number is calculated roughly from the distance that measured from the deviation map legend. Apart from the measuring error, the method of adding draft can introduce error too. The draft adding process is done from a set number of iteration of vertex updating. Considering the running time, the number of iteration is set to be 100. With the number of iteration increases, the final shape will be closer to what it is expected to be. This is beyond the scope of this research, so it won’t be discussed to more detail. Overall, the draft adding process do have added draft on the model, and this result is acceptable.

4. 4 Experiment demo part

A demo part is created to test the method. From the pattern tooling geometry that outputs from the NNSEditor, a wood pattern can be automatically created in the Rapid Pattern Machining that developed in RMPL. Then the sand mold can be pulled out from the pattern tooling, and finally a RE casted part is manufactured. The whole process is show as the figure below.

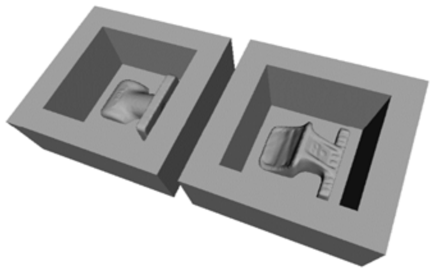


Figure 4.6(a). Output Model of pattern

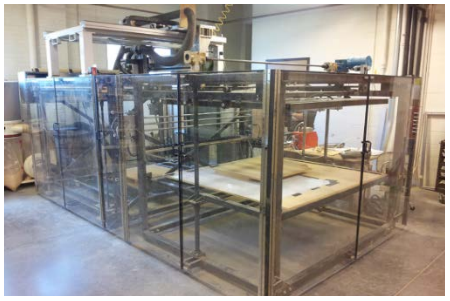


Figure 4.6(b). Rapid Pattern Maker

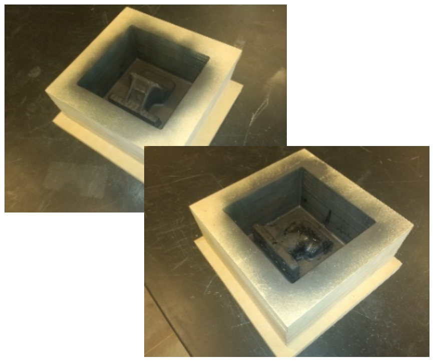


Figure 4.6(c). Wood Pattern created

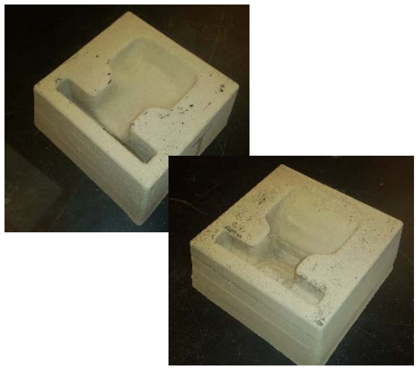


Figure 4.6(d). Sand mold pulled

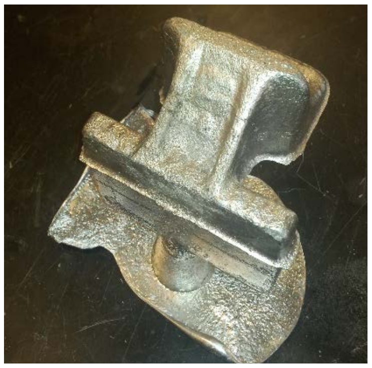


Figure 4.6(e). RECasted Aluminum Part

The overall pattern creating time is showed in table 4.2.

Table 4.2. Time sheet for pattern making (Unit: hour)

Process	Setup	Process Planning	Pattern building	Finishing	Overall
Time	2	0.25	4.3	0.5	7.05

A deviation study is also conducted for the re-cast demo part with reference to the original part. The re-casted aluminum part is created by 5% scaling up. As can be seen from

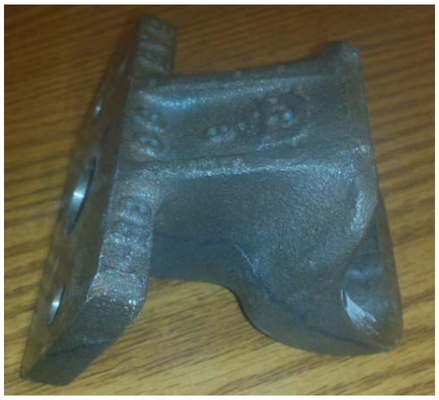


Figure 4.7(a). Reference Geometry: Original demo part

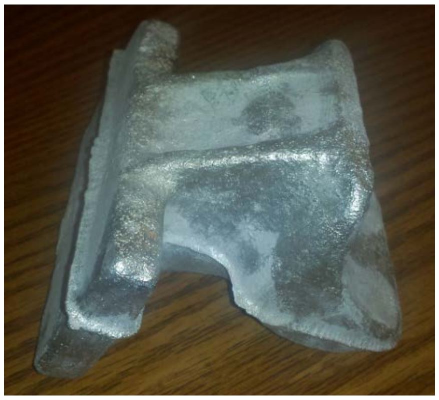


Figure 4.7(b). Compared Geometry: RECasted Aluminum part

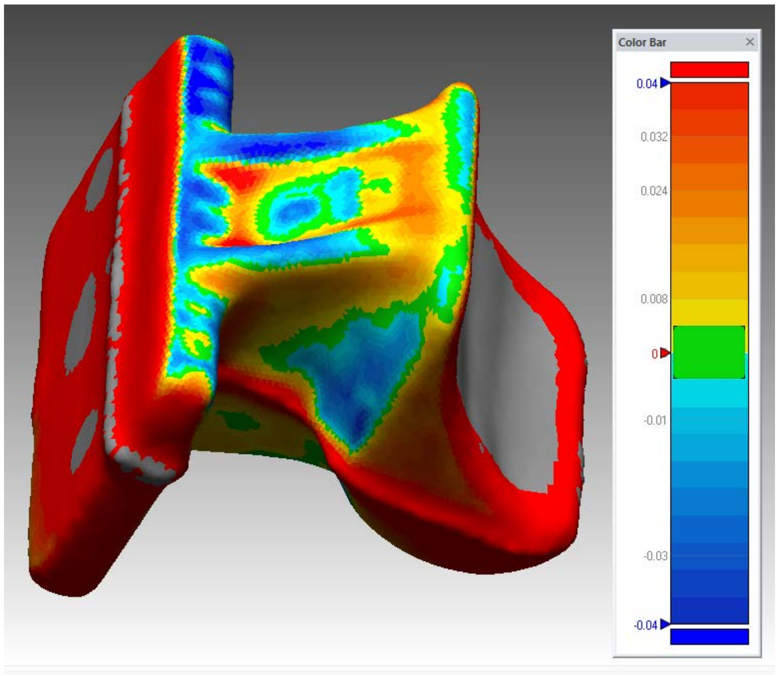


Figure 4.7(c). Surface Deviation Analysis result created from Geomagic Design X64 (Unit:inch)

the colored map, the overall casted geometry is recreated, machining allowance and drafts are also added.

However, there are some errors resulted from casting process and uniform scaling. Also, the casting parting line flash can be noticed from the re-casted part and colored map. This error is resulted from the machining process during pattern making, and that flash is from the ball milling process. Further improvement on the toolpath strategy of RPM can be made to eliminate this error.

Overall, the method of reverse engineering a casted part developed and implemented in this research can be approved by this demo part.

CHAPTER 5. FUTURE WORK AND CONCLUSIONS

5. 1 Future work

This research work successfully integrates advanced reverse engineering through laser scanning with a hybrid method for the additive/subtractive manufacturing of wood pattern tooling for sand casting. All sub-objectives are met: A new method for reverse engineering method is created, a software tool that is designed for feature free model editing is developed and the whole method and process is validated through metal casting experiment. This method is proved to be correct and applicable.

Although we have achieved all objectives of this research, there are still things needed to be done to improve it. Core areas are not considered in this research, and a complete casting system with gating, sprue and riser cannot be generated yet. The future work could possibly be adding the core area analysis by using other scanning technology to obtain the core area geometry, and making this software an add-on function to a casting system design software to enable the creation of a complete casting system. Also, this method be implemented on all other near net shaping process. More importantly, the ultimate goal of reverse engineering is to remanufacture a functional part from the existed part. The reverse engineering method that brought out in this research work is limited to stage one: producing the “as was” casted part, and this is still far from creating a functional part. From the point cloud analysis process of this method, all the machining feature can be identified and recognized. So this information can later be collected and passed on to the machining process. Future work could be integrating this method with the CNC-RP[33] technology.

Then a complete reverse engineering process from near net shaping blank to machining functional part can be established.

5. 2 Conclusions

This research work provides a validated new method along with software tools that overcomes the drawback of the current reverse engineering technologies and takes the benefit from both laser scanning and rapid prototyping technology to reverse engineer legacy casting parts rapidly and economically. The method will improve the situation of maintaining legacy machine by reducing the cost and shorten the time. Also a new concept for editing feature free model is brought out and implemented. It provides a possible way of allowing people to edit the model when reverse engineering a design rather than only take the current shape.

REFERENCES

[1] I. Zeid and C. A. D. Mastering, “Cam.” McGraw Hill international Edition, 2005.

[2] S. Motavalli, “Review of reverse engineering approaches,” *Comput. Ind. Eng.*, vol. 35, no. 1–2, pp. 25–28, 1998.

[3] Vinesh Raja; Kiran J Fernandes, “Introduction to reverse engineering,” in *Reverse Engineering: an Industrial perspective*, London: Springer, 2008, pp. 1–10.

[4] “FARO ScanArm,” 2015. [Online]. Available: http://www.faro.com/en-us/products/metrology/measuring-arm-faro-scanarm/overview.

[5] W. Böhler and A. Marbs, “3D scanning instruments,” *Proc. CIPA WG 6 Int. Work. Scanning Cult. Herit. Rec.*, pp. 9–18, 2002.

[6] M. Blair and T. L. Stevens, *Steel castings handbook.* ASM International, 1995.

[7] X. Luo, “Process planning for an Additive/Subtractive Rapid Pattern Manufacturing System,” *Grad. Theses Diss.*, vol. Ph.D., no. Paper 11027, 2009.

[8] P. J. Besl and R. C. Jain, “Three-dimensional object recognition,” *ACM Comput. Surv.*, vol. 17, no. 1, pp. 75–145, 1985.

[9] B. Guo, “Surface reconstruction: from points to splines,” *Comput. Des.*, vol. 29, no. 4, pp. 269–277, 1997.

[10] R. M. Bolle and B. C. Vemuri, “On three-dimensional surface reconstruction methods,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 1, pp. 1–13, 1991.

[11] T. Várady, R. R. Martin, and J. Cox, “Reverse engineering of geometric models—an introduction,” *Comput. Des.*, vol. 29, no. 4, pp. 255–268, 1997.

[12] W. B. Thompson, J. C. Owen, H. J. De St. Germain, S. R. Stark, and T. C. Henderson, “Feature-based reverse engineering of mechanical parts,” *IEEE Trans. Robot. Autom.*, vol. 15, no. 1, pp. 57–66, 1999.

[13] a. P. Rockwood, “Geometric primitives,” *ACM Comput. Surv.*, vol. 28, no. 1, pp. 149–151, 1996.

[14] a. Durupt, S. Remy, and G. Ducellier, “Knowledge Based Reverse Engineering—An Approach for Reverse Engineering of a Mechanical Part,” *J. Comput. Inf. Sci. Eng.*, vol. 10, no. December 2010, p. 044501, 2010.

[15] J. Huang and C. H. Menq, “Automatic data segmentation for geometric feature extraction from unorganized 3-D coordinate points,” *IEEE Trans. Robot. Autom.*, vol. 17, no. 3, pp. 268–279, 2001.

[16] C. Weber, S. Hahmann, and H. Hagen, “Sharp Feature Detection in Point Clouds,” *Shape Model. Int. Conf. (SMI), 2010*, 2010.

[17] T. Rabbani, F. a van den Heuvel, and G. Vosselman, “Segmentation of point clouds using smoothness constraint,” *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. - Comm. V Symp. ’Image Eng. Vis. Metrol.*, vol. 36, pp. 248–253, 2006.

[18] M. a. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[19] R. Schnabel, R. Wahl, and R. Klein, “Efficient RANSAC for point-cloud shape detection,” *Comput. Graph. Forum*, vol. 26, no. 0, pp. 214–226, 2007.

[20] “Point Cloud Library.” [Online]. Available: http://www.pointclouds.org/. [Accessed: 19-May-2015].

[21] R. B. Rusu and S. Cousins, “3D is here: point cloud library,” *IEEE Int. Conf. Robot. Autom.*, 2011.

[22] “Point Cloud Library (PCL) 1.8.0 Module segmentation.” [Online]. Available: http://docs.pointclouds.org/trunk/group__segmentation.html. [Accessed: 19-May-2015].

[23] “Point Cloud Library (PCL) 1.8.0 Module sample_consensus.” [Online]. Available: http://docs.pointclouds.org/trunk/group__sample__consensus.html. [Accessed: 19-May-2015].

[24] “Region growing segmentation.” [Online]. Available: http://pointclouds.org/documentation/tutorials/region_growing_segmentation.php. [Accessed: 27-May-2015].

[25] H. Seibert, D. Hildenbrand, M. Becker, and A. Kuijper, “Estimation of Curvatures in PointSets Based on Geometric Algebra,” 2009.

[26] X. Yan, K. Yamazaki, and J. Liu, “Recognition of machining features and feature topologies from NC programs,” *CAD Comput. Aided Des.*, vol. 32, no. 10, pp. 605–616, 2000.

[27] R. B. Rusu, “Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments,” *KI - K??nstliche Intelligenz*, pp. 1–4, 2010.

[28] S. Xianfang, P. L. Rosin, R. R. Martin, and F. C. Langbein, “Fast and effective feature-preserving mesh denoising,” *IEEE Trans. Vis. Comput. Graph.*, vol. 13, pp. 925–938, 2007.

[29] M. Botsch, S. Steinberg, S. Bischoff, and L. Kobbelt, “OpenMesh – a generic and efficient polygon mesh data structure,” *OpenSG Symp.*, 2002.

[30] W. Heidrich, “Computing the Barycentric Coordinates of a Projected Point,” *Journal of Graphics, GPU, and Game Tools*, vol. 10, no. 3. pp. 9–12, 2005.

[31] D. Eberly, “Triangulation by ear clipping,” *Magic Software, Inc*, pp. 1–13, 2002.

[32] “Geomagic Design X.” [Online]. Available: http://www.rapidform.com/products/xor/overview/. [Accessed: 25-Jun-2015].

[33] M. C. Frank, R. a. Wysk, and S. B. Joshi, “Rapid planning for CNC milling—A new approach for rapid prototyping,” *J. Manuf. Syst.*, vol. 23, no. 3, pp. 242–255, 2004.

[34] “Sand Casting.” [Online]. Available: http://www.the-warren.org/GCSERevision/engineering/casting.htm. [Accessed: 30-Jun-2015].

[35] “MANUFACTURING, DESIGN, MECHANICAL ENGINEERING.” [Online]. Available: http://www.learneasy.info/MDME/MEMmods/MEM30007A/processing/processing.html. [Accessed: 29-Jun-2015].

[36] E.-C. Modern Machine Shop, Mark Albert, “Feature Recognition--The Missing Link To Automated CAM,” 2001. [Online]. Available: http://www.mmsonline.com/articles/feature-recognition--the-missing-link-to-automated-cam. [Accessed: 30-Jun-2015].

[37] “PCL Documentation.” [Online]. Available: http://www.pointclouds.org/documentation/tutorials/random_sample_consensus.php. [Accessed: 30-Jun-2015].