Automating Fixture Setups Based on Point Cloud Data & CAD Model

by

Satchit Ramnath

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved July 2016 by the
Graduate Supervisory Committee:

Jami Shah, Chair
Joseph Davidson
Dianne Hansford

ARIZONA STATE UNIVERSITY

August 2016

© 2016 Satchit Ramnath
All Rights Reserved

ABSTRACT

Metal castings are selectively machined-based on dimensional control requirements. To ensure that all the finished surfaces are fully machined, each as-cast part needs to be measured and then adjusted optimally in its fixture. The topics of this thesis address two parts of this process: data translations and feature-fitting clouds of points measured on each cast part. For the first, a CAD model of the finished part is required to be communicated to the machine shop for performing various machining operations on the metal casting. The data flow must include GD&T specifications along with other special notes that may be required to communicate to the machinist. Current data exchange, among various digital applications, is limited to translation of only CAD geometry via STEP AP203. Therefore, an algorithm is developed in order to read, store and translate the data from a CAD file (for example SolidWorks, CREO) to a standard and machine readable format (ACIS format - *.sat). Second, the geometry of cast parts varies from piece to piece and hence fixture set-up parameters for each part must be adjusted individually. To predictively determine these adjustments, the datum surfaces, and to-be-machined surfaces are scanned individually and the point clouds reduced to feature fits. The scanned data are stored as separate point cloud files. The labels associated with the datum and to-be-machined (TBM) features are extracted from the *.sat file. These labels are further matched with the file name of the point cloud data to identify data for the respective features. The point cloud data and the CAD model are then used to fit the appropriate features (features at maximum material condition (MMC) for datums and features at least material condition (LMC) for TBM features) using the existing normative feature fitting (nFF) algorithm. Once the feature fitting is complete, a global datum reference frame (GDRF) is constructed based on the locating method that will be used to machine the part. The locating method is extracted from a fixture library that specifies the type of fixturing used to machine the part. All entities are transformed from its local coordinate system into the GDRF. The nominal geometry, fitted features, and the GD&T information are then stored in a neutral file format called the Constraint Tolerance Feature (CTF) Graph. The final outputs are then used to identify the locations of the critical features on each part and these are used to establish the adjustments for its setup prior to machining, in another module, not part of this thesis.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Jami Shah for his guidance and support throughout this research. All the progress made, things learned, would not have been possible without his valuable guidance. I would like to thank Dr. Davidson for his supervision and support, in executing some of the most crucial parts of the research work. I would also like to thank Dr. Dianne Hansford for taking out time to serve on my committee.

I am deeply thankful to my colleagues at DAL lab for their immense support, friendship and guidance.

I must express my very profound gratitude to my parents and to my brother for providing me with unfailing support and continuous encouragement through the process of researching and writing this thesis. I also want to thank my dear Prerana for her heartwarming attention.

Finally, I would like to acknowledge the financial support from DMDII (Digital Manufacturing and Design Innovation Institute), Grant No. DMDII-14-07-03.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

CHAPTER 1

INTRODUCTION

Efficiency in manufacturing is not always improved by replacing existing tools (machinery or machine tools) with newer or expensive ones. Reducing the setup time on the existing machinery or tools, without compromising on the quality, can also help accomplish dramatic results. Sand castings have poor dimensional control, requiring some features to be machined to attain the required finish and achieve dimensional control between surfaces where other components will be attached.

1.1 Background

Sand casting uses disposable molds; a close replica of the desired part (pattern) is used to create a cavity in the sand. After the liquid metal is poured, it is allowed to cool down and solidify, and the cast part is then removed/broken out of the mold to complete the process. Metal castings are generally used for parts involving large parts with non-critical strength requirement which is expensive to manufacture through other methods.

1.1.1 Sand Casting

Sand casting is the most common type due its simplicity and low costs. It enables the production of smaller quantities of castings. The process is characterized by using sand as the mold material. Over 70% of the total metal castings produced are produced by this method. In this process, the sand is held in frames or mold boxes called Flasks. The gate system (for pouring the metal) and mold cavities are created by compacting the sand around models or patterns or are carved directly into the sand. The molten metal is then poured into the mold and allowed to solidify before the mold is separated to access the finished part.

Large castings produced using the sand casting method have a high probability of having part to part variations. Dimensional inaccuracies can be induced by sand used in the process. The sand is abrasive in nature, sometimes causing excessive wear on the machine tools, which in turn affects the precision and reproducibility. Also, when the hot metal is poured into the mold, the ferro-static pressure, and head height, shift the core and induce deformation causing dimensional instability. Uneven cooling and solidification of the metal may also account for dimensional variabilities. As the size of the casting grows, the impact of these variations on the tolerances become very significant.

Surfaces on metal castings are selectively machined to achieve dimensional control between surfaces where other components will be attached. For every casting, two CAD models, one to provide information on casting and one for the finished part obtained after machining, are required. The CAD model for casting is created based on the coring needs, distortion, shrinkage and machining allowance. This also has tolerances specific to the casting process. On the other hand, the CAD model of the finished part after machining contains the dimensions and tolerances that are required at the end of the manufacturing process. Mating surfaces are to be machined to achieve the required surface finish and specified tolerances.

Many cast parts require some features to be machined. These may include bearing surfaces, sealing surfaces, small features etc. Extra material needs to be left to allow machining. One concern is to maintain specific critical parameters, like minimum wall thickness, width and so on, within the specified dimensional limits for certain features. Another is to set up the part on the machining fixture in such a way that the features being machined are cleaned completely and lie within the tolerance specification. Figure 1 shows an example part, as cast and as machined.

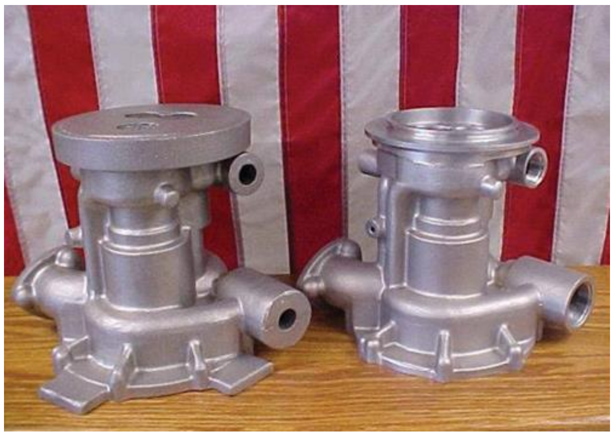
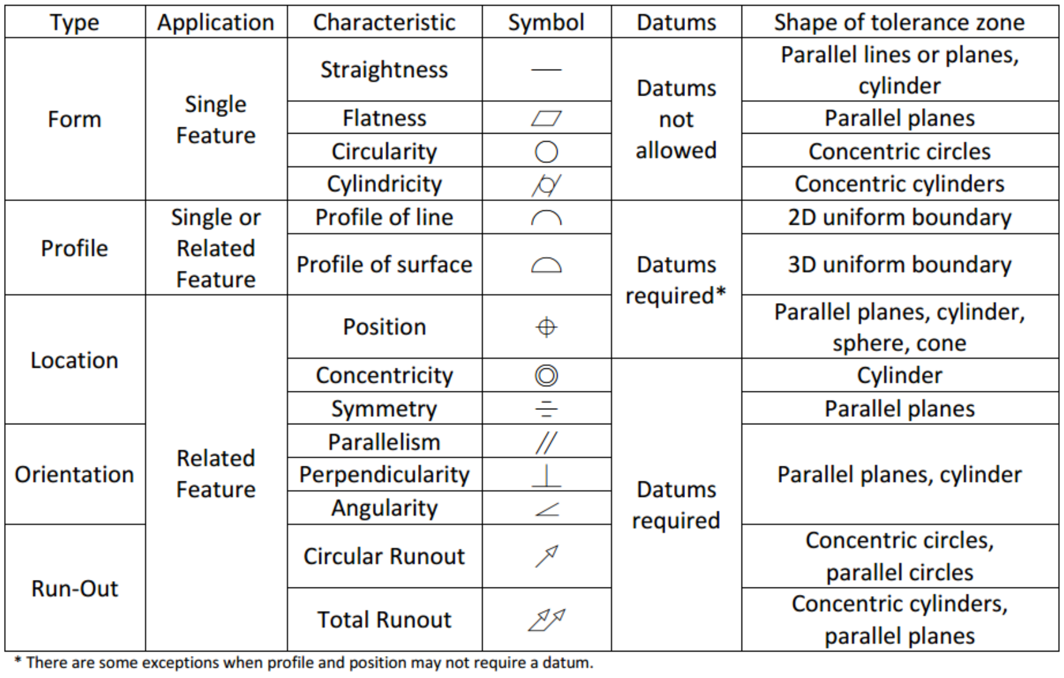


Figure 1: As Cast Part (L), Machined Part (R) [1]

1.1.2 Geometric Dimensioning and Tolerancing

Geometric Dimensioning and Tolerancing (GD&T) is a symbolic language used to specify the allowable limits within which a part may be manufactured. According to ASME Y14.5-2009 standard, the purpose of GD&T is to describe the engineering intent of parts and assemblies.[2] Dimensions specify the nominal or theoretically perfect geometry while tolerances specify the allowable variations. GD&T can be used either on drawings or in 3D models to describe the nominal geometry and the variations allowed. These values specify the amount of precision and accuracy that is required during the machining process for a part. The GD&T specifications on 3D models are called semantic PMI data. This Model Based Design (MBD) approach has helped eliminate the need for traditional drawings.

Tolerances play an important role in design and manufacture. It is practically impossible to machine parts to perfection, hence specifying the right tolerances ensures that the dimensional and geometric variations are within the allowable limits enabling the part to assemble and perform its intended function.[3] Figure 2 shows the different classes, for geometric tolerances, along with their tolerance symbols and a few additional details. Figure 3 shows an example of GD&T on a drawing.



Type	Application	Characteristic	Symbol	Datums	Shape of tolerance zone
Form	Single Feature	Straightness	—	Datums not allowed	Parallel lines or planes, cylinder
		Flatness	⏥		Parallel planes
		Circularity	○		Concentric circles
		Cylindricity	⌭		Concentric cylinders
Profile	Single or Related Feature	Profile of line	⌒	Datums required*	2D uniform boundary
		Profile of surface	⌓		3D uniform boundary
Location	Related Feature	Position	⌖		Parallel planes, cylinder, sphere, cone
		Concentricity	◎	Datums required	Cylinder
		Symmetry	⌯		Parallel planes
Orientation		Parallelism	//		Parallel planes, cylinder
		Perpendicularity	⊥		
		Angularity	∠		
Run-Out		Circular Runout	↗		Concentric circles, parallel circles
		Total Runout	⌰		Concentric cylinders, parallel planes

* There are some exceptions when profile and position may not require a datum.

[*] Position tolerance doesn't require a datum when it is used to control spacing within a pattern

Figure 2: Geometric Tolerance Classes [4]

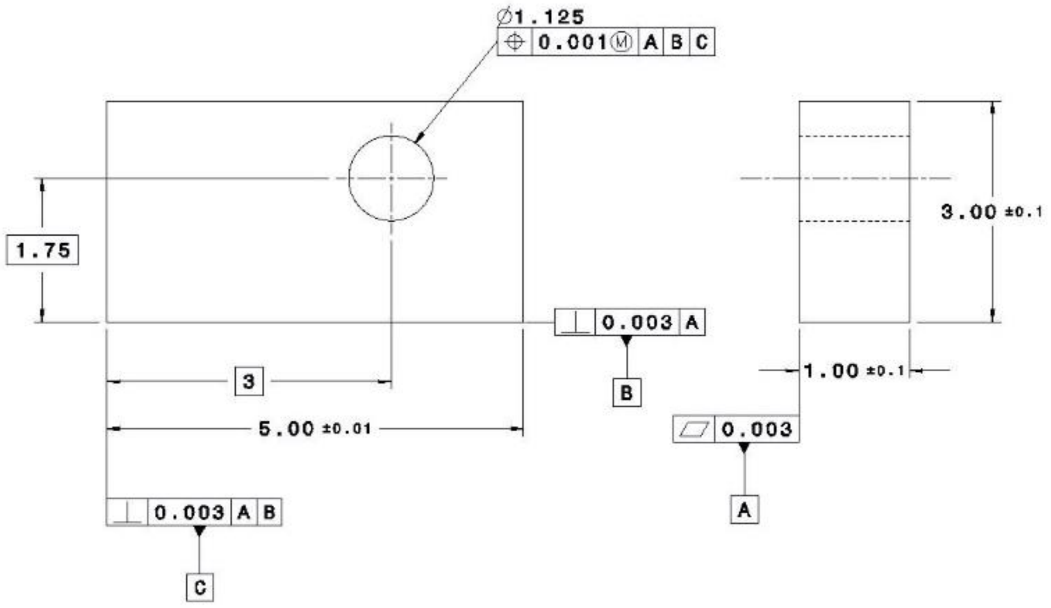


Figure 3: GD&T Sample

The use of GD&T in manufacturing provides multiple benefits, such as ensuring assemblability, increasing interchangeability, and easing integration between multiple systems like CAD, CAM, and CAPP. GD&T, being a symbolic language, reduces ambiguity and the need for special manufacturing notes/instructions. Table 1 shows a list of modifiers that can be used in a feature control frame.

Table 1: Modifiers used in feature control frame

Symbol	Modifier	Notes
Ⓕ	Free state	Applies only when part is otherwise restrained
Ⓛ	Least material condition (LMC)	Useful to maintain minimum wall thickness
Ⓜ	Maximum material condition (MMC)	Provides bonus tolerance only for a feature of size
Ⓟ	Projected tolerance zone	Useful on threaded holes for long studs

Symbol	Modifier	Notes
Ⓢ	Regardless of feature size (RFS)	Not part of the 1994 version. See para. A5, bullet 3. Also para. D3. Also, Figure 3-8.
Ⓣ	Tangent plane	Useful for interfaces where form is not required
Ⓤ	Unilateral	Refers to unequal profile distribution.

1.1.3 Machining Fixtures

Fixtures for machining play an important role in the manufacturing process. They are required to hold the part, position it correctly with respect to the tool and support it during the machining process. There are various fixture schemes that can be employed for parts to be machined. The selection of the right fixture scheme is vital since fixtures have a direct impact on the quality of the machined part and total cost of production. The costs associated with fixture design and manufacturing can account for 10-20% of the total cost of a manufacturing system. [5] Also, the setup time can be a significant portion of the machine time used on each part. Figure 4 shows an example of a part in a fixture.

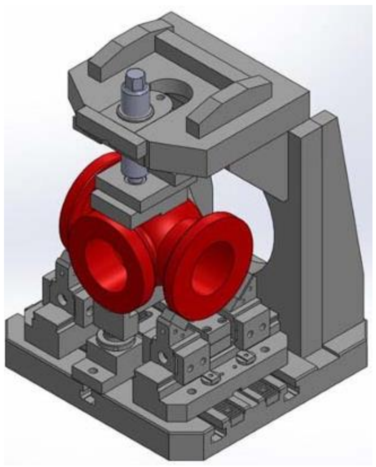


Figure 4: Part in a Fixture [6]

The selection of a right fixture scheme is an experience-based approach and requires a thorough understanding of the machining processes involved. A poor choice of fixture scheme may lead to a surge in the total manufacturing costs, increased machining effort/time or a complete rejection of the part. The selection might also sometimes include trial and error methods to arrive at a decision.

Therefore, the advancements in manufacturing technologies and increased competitiveness required manufacturers to improve the product quality and reduce the total costs. This has led to automating the process of fixture design and integrating various systems like CAD/CAM, computer-aided process planning (CAPP) and computer aided fixture design (CAFD) into the design and manufacturing process. [7]

1.1.4 CAD Data Exchange & Standards

The evolution of CAD systems has transformed the way parts are manufactured. Data exchange between various CAD systems involves multiple software technologies to exchange data from one file format to the other. Current STEP format (AP203) helps exchange only the geometry of the part and not product manufacturing information (PMI) like GD&T. This calls for a translator that can help exchange geometry or boundary representation (BRep) along with the PMI.

Since each CAD system has its own file format and structure, there is nearly always some loss of information during the translation of data. Even in the case of a neutral file format, there is a limitation on what data can be successfully translated from one system to the other since the interpretation of reading and writing a file may be different in each CAD system. Hence it becomes very important to identify specific data that must be translated.

For example, apart from the geometry, if the translator is required to translate other information like PMI, construction history and so on, it has to be specified and sometimes additional steps might be required to fetch these details. There are various translators available, but few are more successful than the others.

1.1.5 Definition of Feature

A feature is defined as a single face or a combination of two or more faces on a part. Features are classified into two types: real and basic. Basic features are further divided into two categories: a feature of size and planar feature.[2] A feature of size is associated with a measurement parameter such as a radius, width, and height while a planar feature is a plane with infinite boundaries.

Table 2 lists the different types of basic features along with the tolerances classes applicable to them.[3]

Table 2: Basic Feature Types & Applicable Tolerance Classes

Feature	Feature Type	Tolerance Classes
Linear Edge or Circular Edge	Line	Form, Orientation, Position & Size
Circular Face or Rectangular Face	Plane	Form, Orientation & Size
Cylindrical Surfaces	Cylinder (Pins and Holes) (Features of Size)	Form, Orientation, Position & Size
Tabular Surfaces	Tabs and Slots (Features of Size)	Form, Orientation, Position & Size

1.2 Research Objective & Scope of Thesis Work

The objective of this research is to reduce the setup time for machining while ensuring a well-finished part from each casting. This done by devising a method using ASU's T-Maps® and S-Maps® to determine fixture adjustments, thereby ensuring that all the required features lie within the specified tolerances. Inputs to construct the T-Maps® and S-Maps® are the nominal geometry, PMI like GD&T, and the location of the datums (something used as a basis for calculating or measuring) and to-be-machined (TBM) features on the actual metal casting. The nominal geometry along with the PMI is obtained from the CAD model, while the location of features is obtained from a set of data points in a particular coordinate system, called a point cloud, of the sand casting.

The scope the thesis is to develop a module to translate CAD data, utilize the point cloud data to represent the features using surface reconstruction, called feature fitting and then communicate the results, in an effective manner that can be used to develop a set of methods to reduce the setup time for machining of large castings.

The scope of this research work extends to planar and cylindrical features.

Major tasks involved in this research are:

- Determine the data content required for performing the required tasks
- Translate CAD data (Brep and PMI) from a commercial CAD system (like CREO) and communicate the required information
- Superimpose the CAD with the scanned point cloud data from the cast part, to represent each face at its actual position using surface reconstruction

- Modify an existing fixture library developed by Payam et-al.[8] to develop algorithms for generalizing locating methods
- Construct a Global Datum Reference Frame (GDRF) based on the locating method used to perform machining operations, and transform all the entities to the GDRF
- Extract PMI information and parse the same into a specialized data structure which will further be used to write a modified version of Constraint Tolerance Feature (CTF) Graph [9] called Machining Constraint Tolerance Feature (MCTF) Graph
- Create each section of the module such that it can be used as independent APIs (Application Programming Interface)

1.2.1 Assumptions

The GD&T information is already specified on the CAD model for the final machined part. The intent is to look for GD&T information specified on the datum features and the TBM features. The scope of this research is defined by assuming the GD&T information specified is complete, accurate and as per the ASME Y14.5 standard. [2]

While doing this research, certain assumptions were made about the raw casting, the point cloud data and the CAD model for the final machined part:

- The only defects on the raw casting are surface defects (like abrasions/ridges) while otherwise, the casting is structurally perfect
- The casting is made within the tolerance limits specified in the CAD model for casting

1.3 Approach Overview

Once the casting process is completed, the datums and TBM features are scanned, for each casting, using a laser scanner in order to generate the point cloud data. The CAD model along with the point cloud data serve as the initial input. The point cloud specifies details about the features on the casting. The type of fixturing used is obtained from the CAD model which, along with the point cloud data helps construct a new coordinate system. All the entities/features are then transformed into the new coordinate system. The transformed entities and their corresponding PMI, are then written out to the output file which is used to develop a set of methods to reduce the fixture setup time for machining of large castings. The digital flow of the research work is shown in Figure 5.

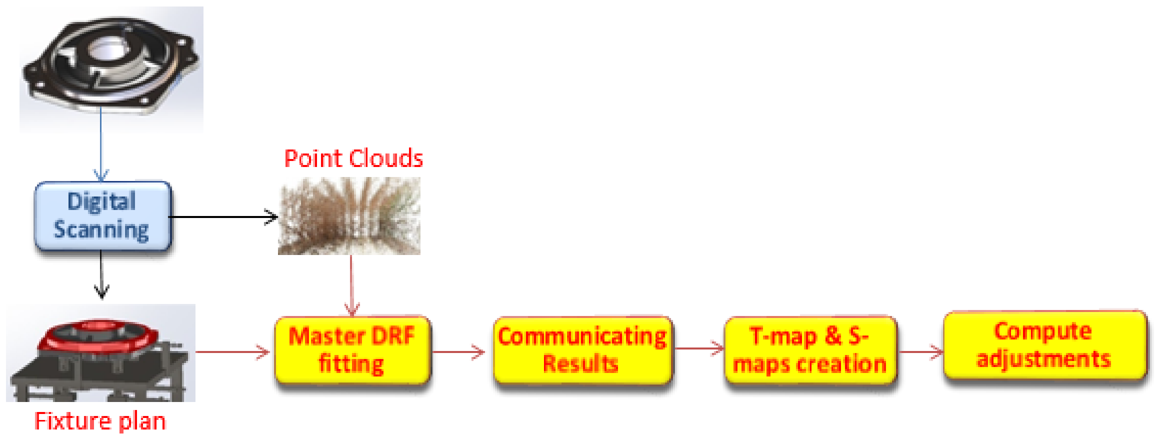


Figure 5: Approach Overview of the Research Project

1.4 Thesis Organization

Chapter 2 is the literature review and detailed description of topics like CAD data exchange, feature fitting and fixturing methods. Chapter 3 talks about the conceptual design. The approach to solving the research problem is discussed in Chapter 4. It also describes the improvements of any existing algorithms and the construction of a coordinate frame based on fixture schemes, and their implementation. Chapter 5 describes validations and verifies the algorithms with test cases. Afterward, conclusions, limitations, and future work in this field of research are discussed in Chapter 6.

CHAPTER 2

LITERATURE REVIEW

The primary areas of interest within the scope of this thesis are (i) CAD data exchange, (ii) PMI interoperability, (iii) Feature fitting to point clouds, and (iv) Fixturing method used to fix a part prior to machining.

2.1 CAD File Formats and Data Exchange

CAD systems are built on modeling kernels which give rise to various CAD data formats based on the kernel. A kernel is defined as a collection of classes and components comprised of mathematical functions performing specified modeling task.[10]

The translation and exchange of CAD data among different CAD systems are categorized as follows [11]:

- Direct CAD translator
- Translating to an intermediate neutral format

Direct CAD system import/export involves a built-in translator in the CAD system that can read/write files that are from a different CAD file format.[12] This is the most convenient method to exchange data between various systems. A major drawback is that most of the file formats are locked to a particular organization and CAD systems not belonging to the same organization (or child organization) may not be able to open/access these files.

Intermediate exchange formats are the most widely used method for data exchange between multiple CAD systems. The CAD system exports data into a neutral file format that the other CAD system can read. It is becoming an increasingly common practice to use such neutral file formats to exchange CAD data. A few example file formats are STEP (Standard for Exchange of Product Data Model) AP203/214, IGES (Initial Graphic Exchange Standard), XML (Extensible Markup Language), 3DXML (3D Extensible Markup Language) to name a few.

There are also a few unique CAD systems that have dual built-in kernel systems. For example IronCAD®, formerly an ACIS®-only system, has been upgraded to incorporate the Parasolid® kernel too, making it the first dual kernel system. The development of a dual kernel system is efficient only when ACIS® and Parasolid® kernels are paired. [10]

The complexity of data translation in a direct translator scales to O(n^2) while the use of an intermediate neutral file format is of the order O(n), where ‘n’ is the number of CAD systems.[11] The Figures 6 and 7 show the complexities of each translation.

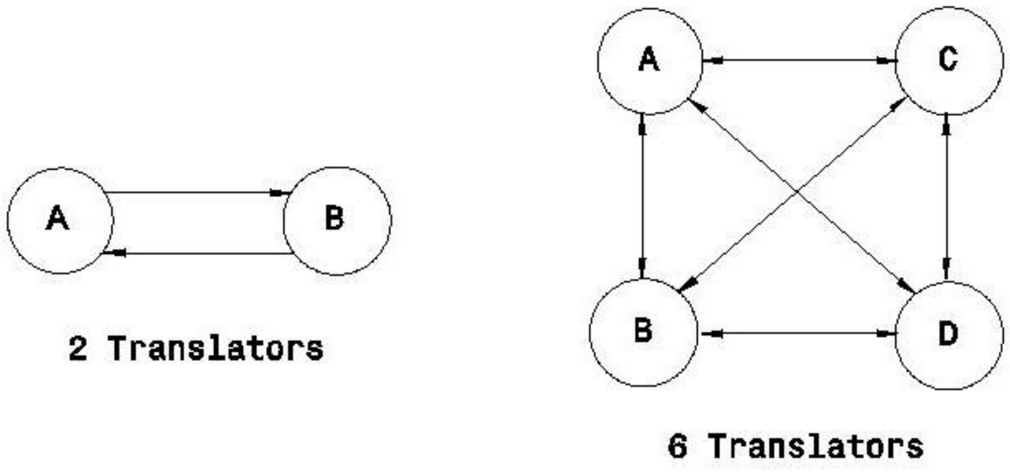
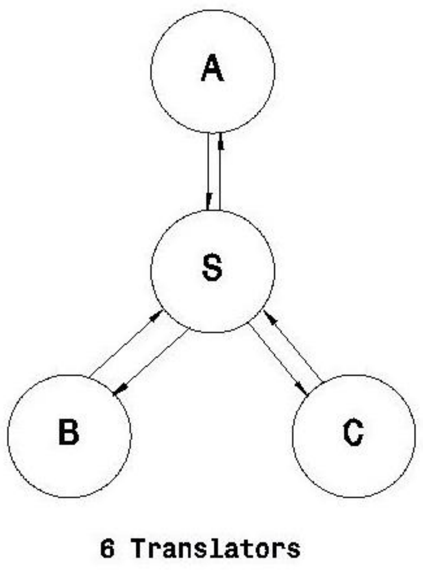


Figure 6: Direct translation between different file formats (n$\times$[n-1])



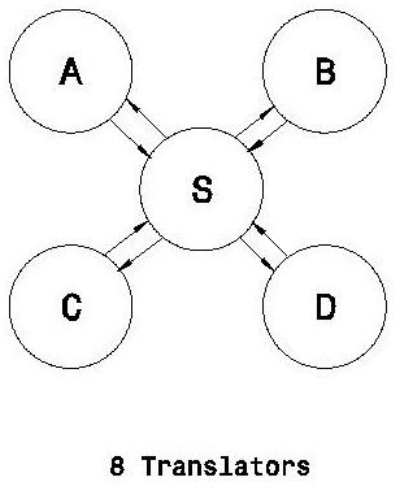


Figure 7: Translation to intermediate file formats (2×n)

The product data quality is dependent on a lot of factors like the kernel, file format and source and destination CAD system. Table 3 below explains how various factors affect the data translation quality[10] and shows that the use of intermediate file formats is the best available option in CAD data translation.

Table 3: Data Translation Matrix

Data Translation Method	**User Friendly**	**Practical**	**Efficiency**
Use of software with the same kernels	****	*	****
Applications that operates on dual kernels	**	**	****
Use of intermediate file formats	****	***	***
Use of direct data translator	***	**	**

* Low; **** High

2.2 PMI Interoperability

PMI conveys non-geometric attributes in CAD systems. The scope of PMI includes (i) GD&T information, (ii) 3D text annotations, (iii) 3D symbols, and (iv) user defined attributes. Figure 8 shows examples of various types of PMI.

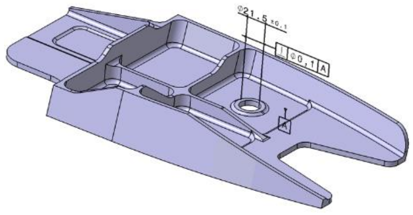
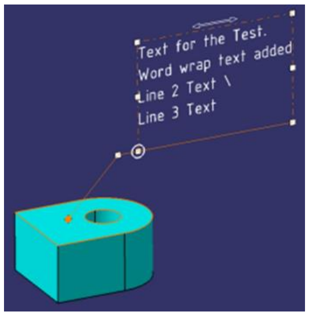
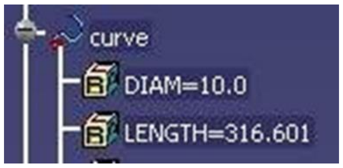
PMI Type	**Example**
3D GD&T	
3D annotations	 Text for the Test. Word wrap text added Line 2 Text \ Line 3 Text
3D symbols	
User defined attributes	 curve DIAM=10.0 LENGTH=316.601

Figure 8: Types of PMI [13]

Interoperability of PMI is based on the way it is specified. It includes capabilities of exchanging information among multiple CAD systems. The PMI can be specified in two ways, based on the intended functionalities, i) PMI semantic representation and (ii) PMI graphic presentation.[13]

PMI graphic representation is used to display the information in its original form in the 3D model by breaking down the annotations and symbols into basic geometry. This method is human-readable only and not machine interpretable. On the other hand, PMI semantic representation describes a way to associate the required PMI to the CAD geometry and further pass it on to other CAD/CAM applications. Most popular CAD systems read the semantic representation and generate a graphical representation for the user to view, thereby making it a human readable and machine interpretable method.

2.3 Feature Fitting of Point Clouds

Several different algorithms have been developed in order to fit features to a point cloud. Some of these algorithms are robust and achieve the results which better approximate the measured part.[3] The choice of algorithm depends on various factors like computation time, type of feature fit and so on.

2.3.1 Least Squares Fits

Least square fits are the most commonly used feature fitting algorithms in the industry. Most CAD systems use least squares fit to fit surfaces to a set of points. However, the results produced by this method are only an approximation of the actual feature. The least square fits begin with an initial guess on where the surface is located in the point cloud. After the initial guess, the algorithm computes the sum of squares of distances between the guessed feature and measured points and then minimizes this value. As the algorithm proceeds, the feature is translated and/or rotated until a minimum value is obtained for the sum of squares of distances. Methods like single value decomposition (SVD) can be used for solving linear least square fit for planar features while Gauss-

Newton algorithm is used to fit cylindrical features. Figure 9 shows a few examples of least square fits. A minimum of two points are required for a line fit; three non-collinear points for planar fits and five points for a cylinder.

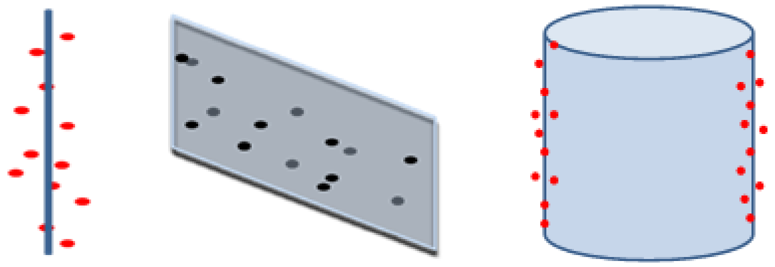


Figure 9: Least Square Fits for Line, Plane, and Cylinder [3]

2.3.2 Unconstrained One Sided Fits

Unconstrained one-sided fits are used to assess if the feature being fit conforms to the specified tolerances for the determination of size, orientation and position as well as establishing a datum. The one-sided fitting algorithm fits a feature to one of the sides of the point cloud i.e. a feature can either be fit at its least material condition or the maximum material condition. To perform a linear fit, two inputs are required; namely, the point cloud data and the initial direction of fit. For a one-sided planar fit, at least three non-collinear points are required along with the initial direction of the fit (dictated by the geometric entity).

The one-sided cylinder fits are of two types: Minimum Circumscribed Cylinder and Maximum Inscribed Cylinder. The type of fit is determined by the type of feature, either an external feature (concave) or an internal feature (convex), of which the point cloud data is generated. For fitting a cylinder at least five points are required that are not coplanar.

The inputs to this algorithm are the point cloud, a point on the axis, radius and the direction of the axis. Figure 10 shows the one-sided fits for planar and cylindrical features.

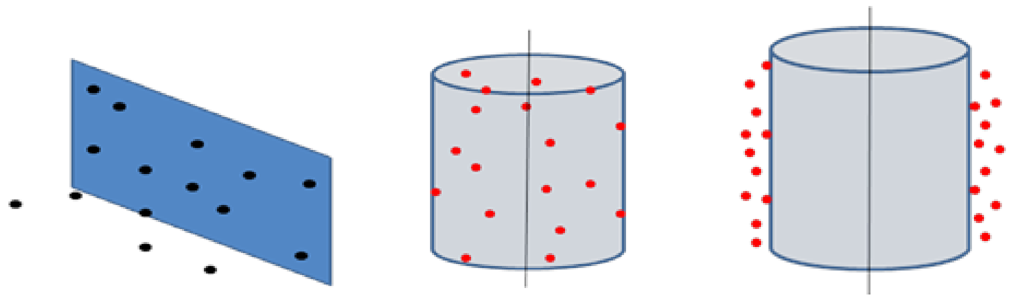


Figure 10: One Sided Fits (L-R → Planar, Circumscribed Cylinder, Inscribed Cylinder)[3]

Table 4 shows the objective functions that have to be satisfied for a one-sided cylindrical fit.

Table 4: Objective Functions for a One Sided Cylindrical Fit[3]

min(R)	For minimum circumscribed cylinder
max(R)	For maximum inscribed cylinder

R = Radius of the cylinder to be fit

2.4 Fixturing Methods

Machining fixtures provide a means to reference and align the cutting tool with the part being machined.[14] For a part to be machined accurately, it is necessary to maintain the position and orientation of the part in space, with respect to its datum points and/or surfaces. This is accomplished by arresting the degrees of freedom of the part in a specific coordinate system. The six degrees of freedom to be arrested are the three (3) translations and three (3) rotations about x, y and z-axes.

Fixturing methods can be classified into various types based on multiple factors.[14] One such classification is based on the construction of the fixtures. They are (i) Plate Fixtures, (ii) Angle-Plate Fixtures, (iii) Vise-Jaw Fixture, (iv) Indexing Fixtures, and (v) Multi-Part Fixtures. Figure 11 shows a few examples of such fixtures.

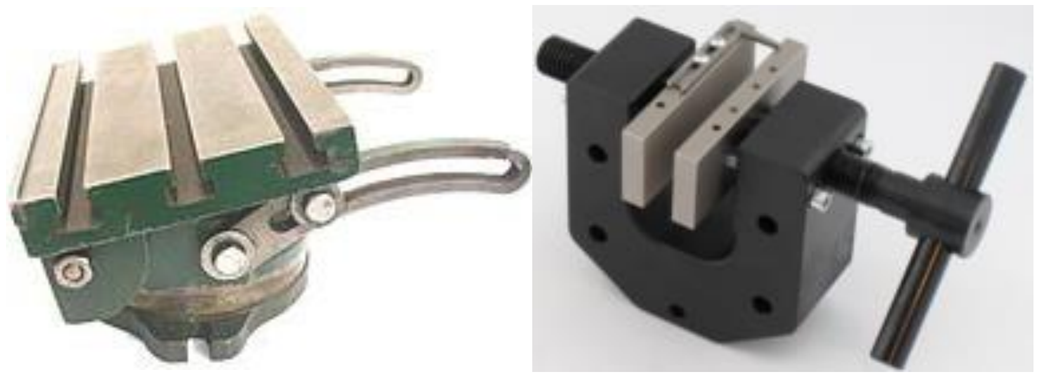


Figure 11: (L→ R) Adjustable Plate Fixture[15], Vise-Jaw Fixture[16]

Another prominent classification is based on the machining process or the machine tool to be used in the process. The primary types include, (i) Milling Fixtures, (ii) Lathe fixtures, (iii) Grinding Fixtures, and (iv) Broaching Fixtures.

Milling fixtures are the most common type of fixtures used that include standard vises and clamps. Fixtures are mounted on the table using elements such as clamps, t-slot bolts, nuts, and jacks. The fixture type considered in this research is limited to milling fixtures only.

2.4.1 Locating Model Classification

Locating method selection is based on the workpiece fixture requirement. It is an important part of the machining fixture which controls the position and orientation of the part. Although various fixture configurations can be found in the industry, the most widely used fixture surfaces and locating methods are quite limited. [17] The locating surfaces are divided into three categories: plane, pin-hole, and external profile. For each of the locating methods, different variations exist. The most common locating methods, based on milling fixtures, are shown in Figure 12.

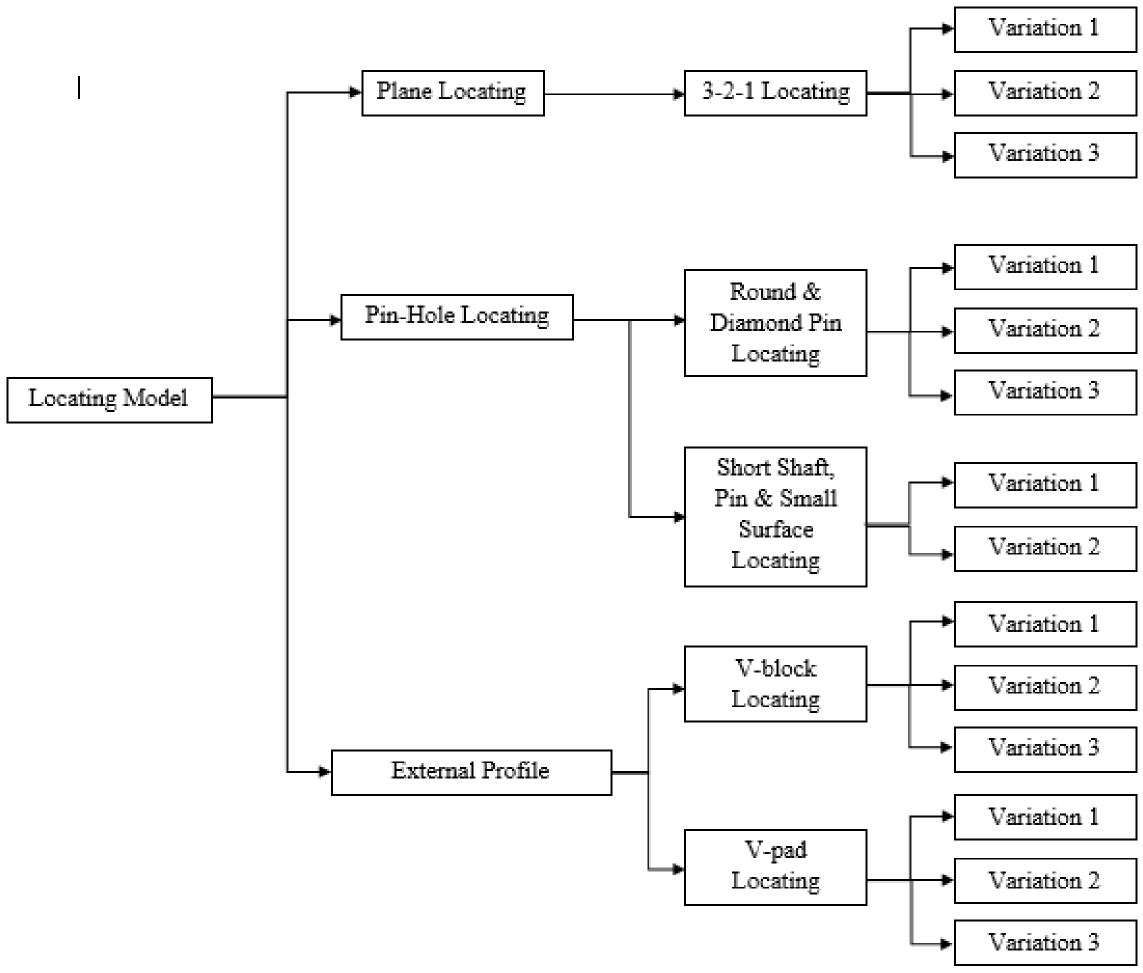


Figure 12: Locating Method Classifications

CHAPTER 3

CONCEPTUAL DESIGN

The final results of the fixture adjustments are obtained using T-Maps® and S-Maps®. Inputs required to develop these methods are:

(i) The nominal geometry of the machined part, along with PMI like GD&T

(ii) The position & orientation of the datums and TBM features on the actual casting

There are various CAD file formats that can be used to communicate the nominal geometry along with the PMI. For representing the position and orientation of the datums and TBM features, the features are scanned on the actual casting to generate a point cloud. Each feature has a separate point cloud file. These point clouds are then simulated to fit the appropriate feature based on the input CAD model. The output obtained after fitting features to the point cloud, along with the CAD geometry and PMI, serve as inputs to devise methods that compute the fixture adjustments.

3.1 Data Formats and Conversion

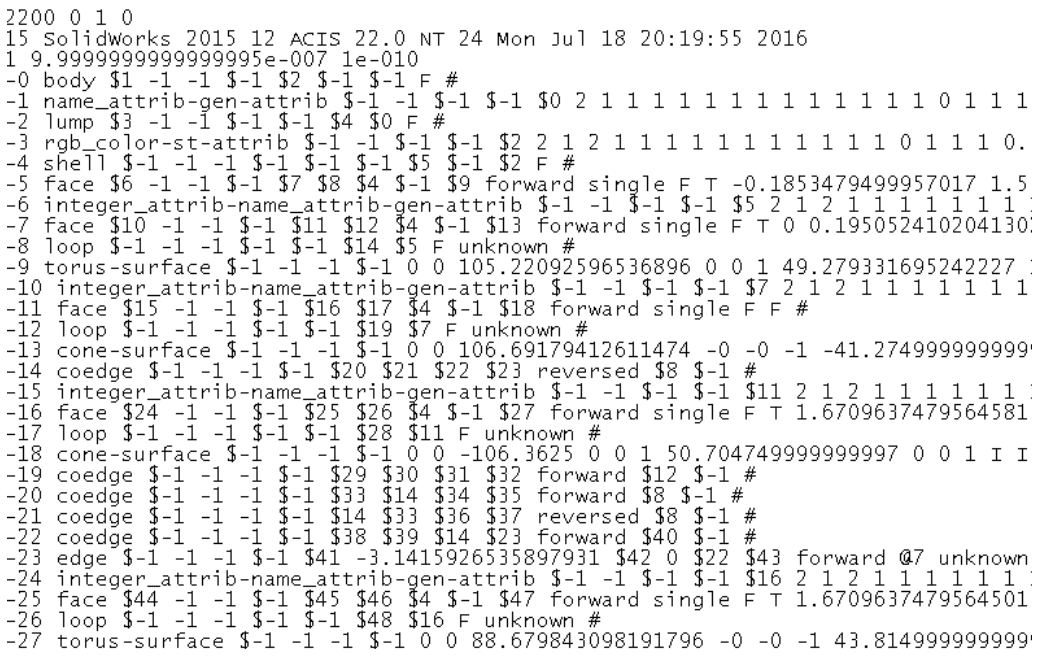
The CAD data required can be communicated using various file formats like SolidWorks part file, CREO part file, or in neutral file formats such as IGES, STEP. The data from these files are extracted using Spatial's geometric translator called InterOp® and written into an ACIS SAT file. An exercise was performed at the beginning of the research to identify suitable file formats to communicate the data. The summary is shown in Table 5.

Table 5: Summary for File Formats

File Type	Data Content	Formats Tested	Results
CAD File	CAD geometry + PMI	ACIS® SAT (*.sat)	CAD Geometry only
		STEP AP203 (*.step)	CAD Geometry only
		STEP AP242 (*.stp)	CAD Geometry + PMI
		SolidWorks Part (*.sldprt)	CAD Geometry only
		CREO® (*.prt)	CAD Geometry + PMI

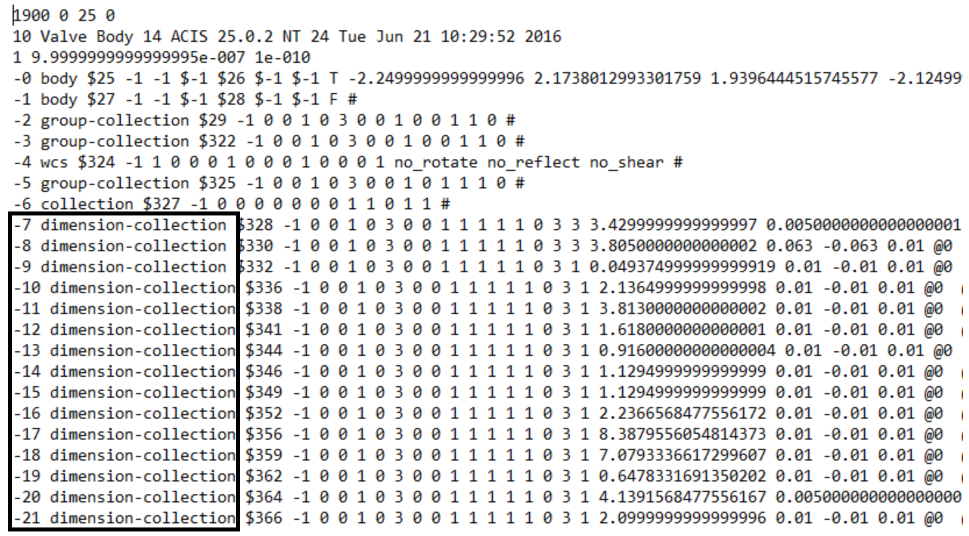
The file formats listed above provides means to transfer of CAD geometry and in some cases PMI. The ACIS SAT, STEP AP203, and SolidWorks Part files, when written out directly by commercial CAD systems, were only able to communicate the CAD geometry and not PMI, during translation. Whereas, the STEP AP242 and CREO part file could effectively communicate the geometry along with the PMI, when translated using InterOp.

Figure 13(a) shows an example of an ACIS SAT file written out directly using SolidWorks and Figure 13(b) shows an example of an ACIS SAT file written after translating data from a CREO® part file, using InterOp. The SAT file written out directly by the CAD system doesn't contain the PMI.



```
2200 0 1 0
15 SolidWorks 2015 12 ACIS 22.0 NT 24 Mon Jul 18 20:19:55 2016
1 9.9999999999999995e-007 1e-010
-0 body $1 -1 -1 $-1 $2 $-1 $-1 F #
-1 name_attrib-gen-attrib $-1 -1 $-1 $-1 $0 2 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1
-2 lump $3 -1 -1 $-1 $-1 $4 $0 F #
-3 rgb_color-st-attrib $-1 -1 $-1 $-1 $2 2 1 2 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0.
-4 shell $-1 -1 -1 $-1 $-1 $-1 $5 $-1 $2 F #
-5 face $6 -1 -1 $-1 $7 $8 $4 $-1 $9 forward single F T -0.1853479499957017 1.5
-6 integer_attrib-name_attrib-gen-attrib $-1 -1 $-1 $-1 $5 2 1 2 1 1 1 1 1 1
-7 face $10 -1 -1 $-1 $11 $12 $4 $-1 $13 forward single F T 0 0.19505241020413
-8 loop $-1 -1 -1 $-1 $-1 $14 $5 F unknown #
-9 torus-surface $-1 -1 -1 $-1 0 0 105.22092596536896 0 0 1 49.279331695242227
-10 integer_attrib-name_attrib-gen-attrib $-1 -1 $-1 $-1 $7 2 1 2 1 1 1 1 1 1
-11 face $15 -1 -1 $-1 $16 $17 $4 $-1 $18 forward single F F #
-12 loop $-1 -1 -1 $-1 $-1 $19 $7 F unknown #
-13 cone-surface $-1 -1 -1 $-1 0 0 106.69179412611474 -0 -0 -1 -41.27499999999
-14 coedge $-1 -1 -1 $-1 $20 $21 $22 $23 reversed $8 $-1 #
-15 integer_attrib-name_attrib-gen-attrib $-1 -1 $-1 $-1 $11 2 1 2 1 1 1 1 1 1
-16 face $24 -1 -1 $-1 $25 $26 $4 $-1 $27 forward single F T 1.6709637479564581
-17 loop $-1 -1 -1 $-1 $-1 $28 $11 F unknown #
-18 cone-surface $-1 -1 -1 $-1 0 0 -106.3625 0 0 1 50.704749999999997 0 0 1 I I
-19 coedge $-1 -1 -1 $-1 $29 $30 $31 $32 forward $12 $-1 #
-20 coedge $-1 -1 -1 $-1 $33 $14 $34 $35 forward $8 $-1 #
-21 coedge $-1 -1 -1 $-1 $14 $33 $36 $37 reversed $8 $-1 #
-22 coedge $-1 -1 -1 $-1 $38 $39 $14 $23 forward $40 $-1 #
-23 edge $-1 -1 -1 $-1 $41 -3.1415926535897931 $42 0 $22 $43 forward @7 unknown
-24 integer_attrib-name_attrib-gen-attrib $-1 -1 $-1 $-1 $16 2 1 2 1 1 1 1 1 1
-25 face $44 -1 -1 $-1 $45 $46 $4 $-1 $47 forward single F T 1.6709637479564501
-26 loop $-1 -1 -1 $-1 $-1 $48 $16 F unknown #
-27 torus-surface $-1 -1 -1 $-1 0 0 88.679843098191796 -0 -0 -1 43.81499999999
```

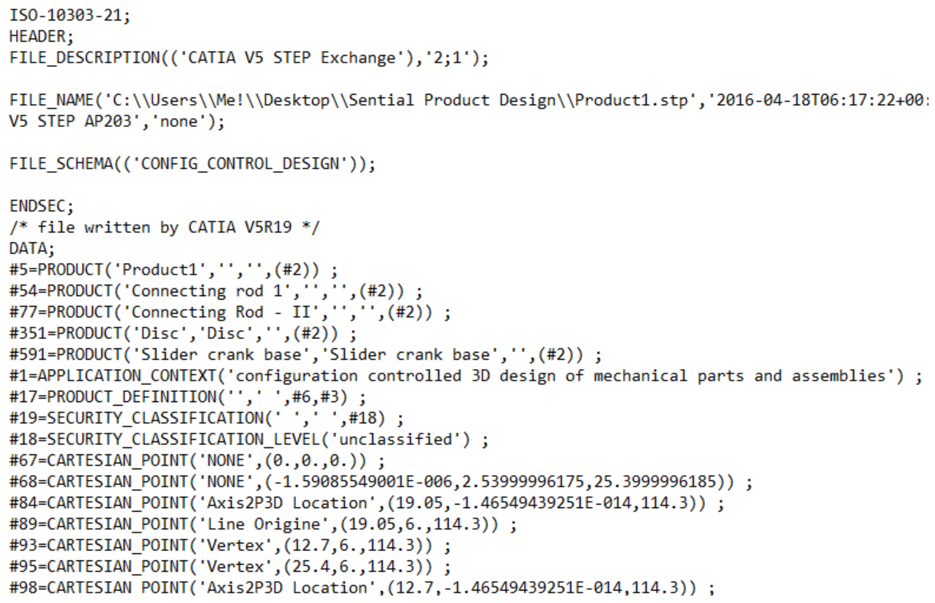
Figure 13(a): Sample ACIS SAT file written out directly using a CAD system



```
1900 0 25 0
10 Valve Body 14 ACIS 25.0.2 NT 24 Tue Jun 21 10:29:52 2016
1 9.9999999999999995e-007 1e-010
-0 body $25 -1 -1 $-1 $26 $-1 $-1 T -2.2499999999999996 2.1738012993301759 1.9396444515745577 -2.12499
-1 body $27 -1 -1 $-1 $28 $-1 $-1 F #
-2 group-collection $29 -1 0 0 1 0 3 0 0 1 0 0 1 1 0 #
-3 group-collection $322 -1 0 0 1 0 3 0 0 1 0 0 1 1 0 #
-4 wcs $324 -1 1 0 0 0 1 0 0 0 1 0 0 0 1 no_rotate no_reflect no_shear #
-5 group-collection $325 -1 0 0 1 0 3 0 0 1 0 1 1 1 0 #
-6 collection $327 -1 0 0 0 0 0 0 0 1 1 0 1 1 #
-7 dimension-collection $328 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 3 3.4299999999999997 0.0050000000000000001
-8 dimension-collection $330 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 3 3.8050000000000002 0.063 -0.063 0.01 @0
-9 dimension-collection $332 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 0.049374999999999919 0.01 -0.01 0.01 @0
-10 dimension-collection $336 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 2.1364999999999998 0.01 -0.01 0.01 @0
-11 dimension-collection $338 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 3.8130000000000002 0.01 -0.01 0.01 @0
-12 dimension-collection $341 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 1.6180000000000001 0.01 -0.01 0.01 @0
-13 dimension-collection $344 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 0.91600000000000004 0.01 -0.01 0.01 @0
-14 dimension-collection $346 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 1.1294999999999999 0.01 -0.01 0.01 @0
-15 dimension-collection $349 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 1.1294999999999999 0.01 -0.01 0.01 @0
-16 dimension-collection $352 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 2.2366568477556172 0.01 -0.01 0.01 @0
-17 dimension-collection $356 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 8.3879556054814373 0.01 -0.01 0.01 @0
-18 dimension-collection $359 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 7.0793336617299607 0.01 -0.01 0.01 @0
-19 dimension-collection $362 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 0.6478331691350202 0.01 -0.01 0.01 @0
-20 dimension-collection $364 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 4.1391568477556167 0.0050000000000000000
-21 dimension-collection $366 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 2.0999999999999996 0.01 -0.01 0.01 @0
```

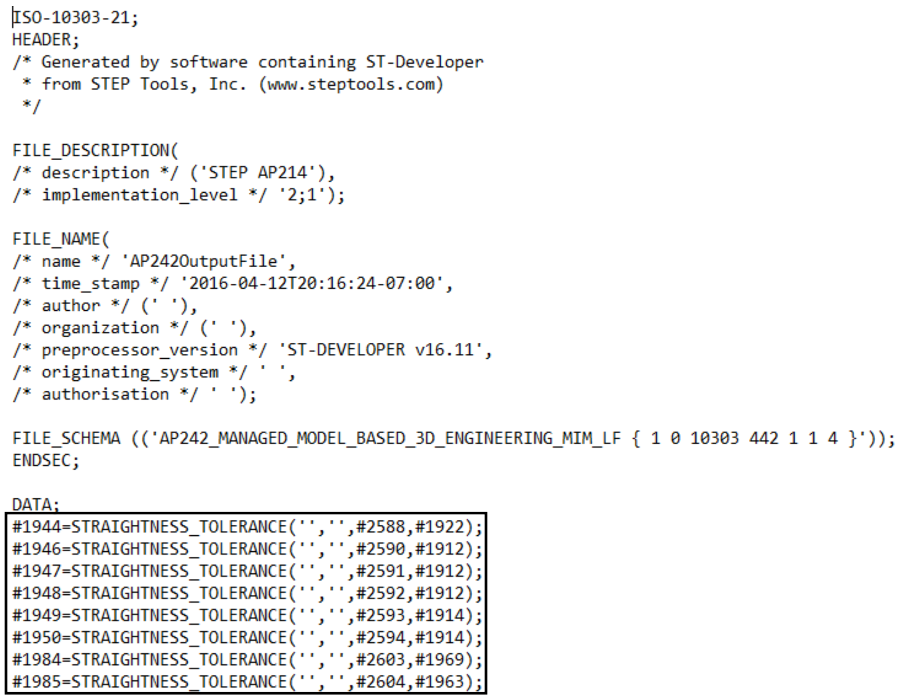
Figure 13(b): Sample ACIS SAT file written using InterOp, containing PMI (eg. dimensions)

Figure 14 is an example of a STEP AP203 file. The contents of this file are limited to CAD geometry only and Figure 15 is an example of STEP AP242 which contains the CAD geometry and the PMI.



```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('CATIA V5 STEP Exchange'),'2;1');

FILE_NAME('C:\\Users\\Me!\\Desktop\\Sential Product Design\\Product1.stp','2016-04-18T06:17:22+00:
V5 STEP AP203','none');

FILE_SCHEMA(('CONFIG_CONTROL_DESIGN'));

ENDSEC;
/* file written by CATIA V5R19 */
DATA;
#5=PRODUCT('Product1','','',(#2)) ;
#54=PRODUCT('Connecting rod 1','','',(#2)) ;
#77=PRODUCT('Connecting Rod - II','','',(#2)) ;
#351=PRODUCT('Disc','Disc','',(#2)) ;
#591=PRODUCT('Slider crank base','Slider crank base','',(#2)) ;
#1=APPLICATION_CONTEXT('configuration controlled 3D design of mechanical parts and assemblies') ;
#17=PRODUCT_DEFINITION('',' ',#6,#3) ;
#19=SECURITY_CLASSIFICATION(' ',' ',#18) ;
#18=SECURITY_CLASSIFICATION_LEVEL('unclassified') ;
#67=CARTESIAN_POINT('NONE',(0.,0.,0.)) ;
#68=CARTESIAN_POINT('NONE',(-1.59085549001E-006,2.53999996175,25.3999996185)) ;
#84=CARTESIAN_POINT('Axis2P3D Location',(19.05,-1.46549439251E-014,114.3)) ;
#89=CARTESIAN_POINT('Line Origine',(19.05,6.,114.3)) ;
#93=CARTESIAN_POINT('Vertex',(12.7,6.,114.3)) ;
#95=CARTESIAN_POINT('Vertex',(25.4,6.,114.3)) ;
#98=CARTESIAN_POINT('Axis2P3D Location',(12.7,-1.46549439251E-014,114.3)) ;
```

Figure 14: Sample STEP AP203 containing only CAD geometry



```
ISO-10303-21;
HEADER;
/* Generated by software containing ST-Developer
 * from STEP Tools, Inc. (www.steptools.com)
 */

FILE_DESCRIPTION(
/* description */ ('STEP AP214'),
/* implementation_level */ '2;1');

FILE_NAME(
/* name */ 'AP242OutputFile',
/* time_stamp */ '2016-04-12T20:16:24-07:00',
/* author */ (' '),
/* organization */ (' '),
/* preprocessor_version */ 'ST-DEVELOPER v16.11',
/* originating_system */ ' ',
/* authorisation */ ' ');

FILE_SCHEMA (('AP242_MANAGED_MODEL_BASED_3D_ENGINEERING_MIM_LF { 1 0 10303 442 1 1 4 }'));
ENDSEC;

DATA;
#1944=STRAIGHTNESS_TOLERANCE('','',#2588,#1922);
#1946=STRAIGHTNESS_TOLERANCE('','',#2590,#1912);
#1947=STRAIGHTNESS_TOLERANCE('','',#2591,#1912);
#1948=STRAIGHTNESS_TOLERANCE('','',#2592,#1912);
#1949=STRAIGHTNESS_TOLERANCE('','',#2593,#1914);
#1950=STRAIGHTNESS_TOLERANCE('','',#2594,#1914);
#1984=STRAIGHTNESS_TOLERANCE('','',#2603,#1969);
#1985=STRAIGHTNESS_TOLERANCE('','',#2604,#1963);
```

Figure 15: Sample STEP AP242 containing CAD geometry along with PMI

The file formats CREO® part file and STEP AP242 are the only file formats that have the capability to communicate the CAD geometry along with PMI. Current CAD systems lack the capability to write out a STEP AP242. Hence the end results from the exercise suggest that a CREO® part file is the most suitable format for data communication.

3.2 Point Cloud Data

The datum and TBM features are located on the actual casting by scanning it using a laser or touch probe scanner. Apart from the location, these point clouds also specify the position and orientation of the datum and TBM features, on the actual part. Initially, the approach adopted to locate the point cloud data on the CAD model was to use a set of surfaces that were fit using the least square method, to the point cloud data. The overlapping of the point cloud data and the set of surfaces would then help associate each point cloud data set to its corresponding nominal feature on the nominal CAD geometry. This set of surfaces were to be communicated as separate CAD files, for each feature. The various formats tested to communicate the point cloud and the set of surfaces are shown in Table 6. The IGES format was inefficient to read and extract data, given the huge number of points in a single CAD file. On the other hand, the ASCII file which had the x, y and z-coordinates of the points for one feature at a time, was a better way to communicate the point cloud data.

Table 6: File Formats for Set of Surfaces used to Identify Point Cloud Location

Data Type	**Content**	**File Formats**
Point Cloud	Datum + TBM features	IGES (*.igs)
		ASCII (*.asc)
NURBS	Least squares fit to point cloud	*.stp, *.igs, *.sat

Another approach to associate the point clouds to their respective features on the CAD model was to use a specific naming convention for each of the features. The label specified to a feature in the CAD model will then be used to name the point cloud file for that particular feature. This eases the process of associating the point cloud to its corresponding feature as well as saves a lot of computation time since individual point cloud files contain less number of points thereby not requiring the software to filter and/or perform the additional computation. In Figure 16, the figure on the left is an example of a laser scanner used to scan parts while the one on the right shows how a point cloud looks like after scanning a part.

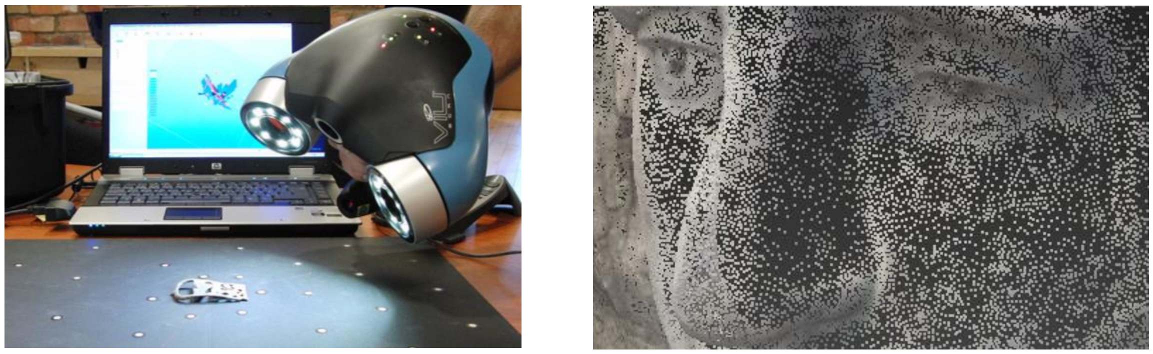


Figure 16: L→R: Laser Scanner[18], Sample Point Cloud[19]

3.3 Normative Feature Fitting (nFF) Library

Fitting the appropriate feature to each of the point cloud data set is required to compare the position and orientation of the feature on the actual part to its intended position and orientation on the nominal CAD geometry. In this research, the features are fit using the nFF library developed by Mohan et-al[3]. The nFF library is an in-house feature fitting library developed at ASU. It consists of 26 algorithms for various fits on point cloud data for a line, circle, plane, and cylinder. The types of fits that can be performed using the library are least squares fit, nominal one sided and two sided (zone) fits. Table 7 represents all the algorithms in the nFF library.[3]

Table 7: nFF Library

Algorithm Type				
	Unconstrained		**Constrained**	
Least Square	**One Sided**	**Minimum Zone**	**One Sided**	**Minimum Zone**
line	line	line zone	1D: line	line zone
circle	- circumscribed circle - inscribed circle	annular zone	- circumscribed circle - inscribed circle	annular zone
plane	plane	- external plane zone - internal plane zone	plane	- external plane zone - internal plane zone
cylinder	- circumscribed cylinder - inscribed cylinder	cylinder zone	- circumscribed cylinder - inscribed cylinder	cylinder zone

This research is limited to least squares and unconstrained one-sided fits to planar and cylindrical features only.

The feature fitting algorithms are developed in C++ in Visual Studio. The feature fitting algorithms are designed as separate function application programming interfaces (APIs). Matrices are used to store the point cloud data and parameters of the fitted features. Vectors and matrices are also used to store the intermediate data from various computations.

3.3.1 Least Square Fit Algorithm for Plane

The least squares fit algorithm for a planar fit requires the point cloud data as input parameters, and the output of the algorithm are a point on the fitted plane and a direction vector normal to the plane. The following steps are followed, in order, to perform a least square planar fit to a point cloud:

I. Number of input points with coordinates *xi*, *yi*, *zi* (i = ith point) is equal to *n*

II. From the point cloud data, the centroid ($\bar{x}$, $\bar{y}$, $\bar{z}$) is calculated which is a point on the fitted plane

III. Matrix *A* of size $n \times 3$ is constructed such that *i*th row is ($xi - \bar{x}$, $yi - \bar{y}$, $zi - \bar{z}$)

IV. Single value decomposition (SVD) of matrix *A* is calculated

V. The vector obtained from the SVD, which corresponds to the smallest single value, gives the direction vector of the fitted plane

3.3.2 Least Square Fit Algorithm for Cylinder

The set of parameters required as input (from the nominal geometry), to fit a cylinder using the least squares algorithm are, the point cloud (x, y & z coordinates for every point), a point (*xo*, *yo*, *zo*) on the cylinder axis, a unit vector (*a*, *b*, *c*) along the cylinder axis and the radius of the cylinder. The input point cloud is rotated such that the initial direction of fit (same as the direction of the cylinder axis) is aligned with the z-axis of the coordinate system. An initial estimate of the cylinder parameters is taken as input followed by Gauss-Newton iterations.

The details and steps for fitting a cylinder by least square method are as follows:

I. The point cloud data is translated such that initial estimate of the point on the axis lies at the origin

II. Point cloud data is rotated by a rotation matrix U such that the axis of cylinder is initially along the z-axis

III. A Jacobian matrix *J* is constructed as (-xi/ri, -yi/ri, -xi zi /ri, - yi zi /ri, -1) where ri is the distance of the *ith* point from the cylinder axis

IV. Right-hand side vector *d* is constructed such that *di = ri – r*, where *di* is the *ith* element of the vector

V. Solve the system of linear equations Jx = – d to obtain the values of x & y coordinate of a point on the axis *(Pxo, Pyo),*components of the axis vector (*Pa, Pb),* and the radius *(Pr)*

VI. The initial set of parameters is updated with these results

VII. These values are rotated (by multiplying with the matrix U^T) and the radius and estimated point on the axis are updated by replacing the old values with the new ones. The direction vector is also updated by taking the new vector as the new estimate

VIII. Process is repeated till the algorithm converges

IX. Every iteration begins with a fresh set of point cloud data which is neither translated nor rotated

3.3.3 Unconstrained One-Sided Fit Algorithm for Plane

In a one-sided planar fit, the plane is defined by three non-collinear points. The 3D convex hull along with the initial direction of fit are the input parameters to the algorithm. The steps to fit a plane are as follows:

I. Point cloud data is read from a text file and stored in a $n \times 3$ matrix (where n corresponds to the number of points)

II. The convex hull of the point cloud data is constructed using quick hull

III. The normal (in the direction away from the convex hull) for all the planes in the convex hull are computed

IV. The angle between each plane normal and the initial direction is calculated

V. The plane normal that is closest to being parallel to the initial direction is selected and the corresponding plane gives the final fit.

The procedure to fit a plane to a point cloud is explained in Figure 17 which also shows the face normal (90^o to the face).

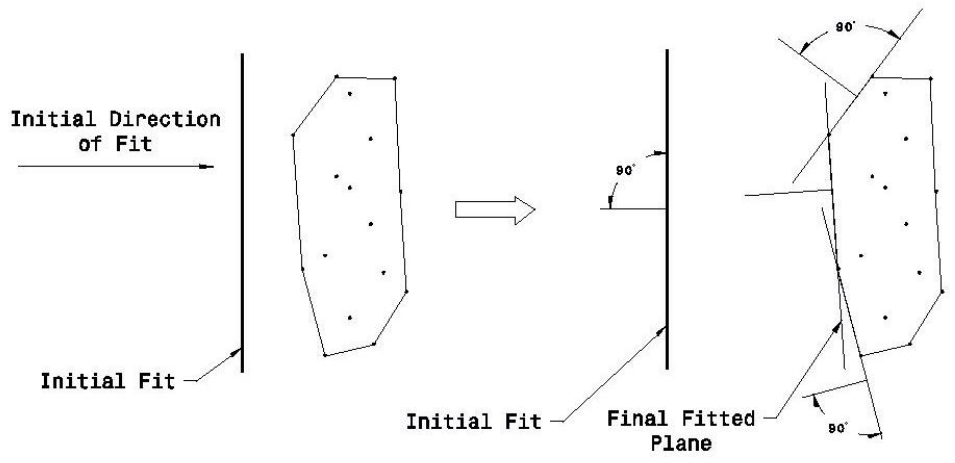


Figure 17: Unconstrained One Sided Planar Fit

3.3.4 Unconstrained One-Sided Fit Algorithm for Cylinder

In order to fit a cylinder, the set of input parameters required are the point cloud (x, y & z coordinates for every point), a point on the axis of the cylinder (*xo, yo, zo*), direction vector (unit vector) of the axis (*a, b, c*) and the radius of the cylinder. The algorithm starts by using the least squares method to fit an initial cylinder to the point cloud. The steps to fit a cylinder to the point cloud are listed below:

I. Point cloud data from a text file are read and stored in a $n \times 3$ matrix (where *n* is the number of points)

II. A least square cylinder is fit (F1) to the point cloud using the given point on the axis, unit vector representing the axis and radius

III. The cylinder axis of F1 is rotated such that it aligns with the Z axis and the same transformation is applied to the point cloud

IV. Extreme ends (e1, e2) of the cylinder are found

V. Two hexagons, h1 & h2 of size equal to 10% of the cylinder radius, are constructed at both the ends (e1 & e2), such that they are parallel to the XY plane

VI. All the points on one hexagon are connected to all the points on the other, creating a total of 36 lines

VII. Each line is considered as the axis of the cylinder, and its corresponding cylindrical fit (F_i) is compared to the previous fit (F_{i-1}) to see if a better solution is obtained

VIII. Upon finding a better solution, the input parameters for F1 are replaced by the ones of Fi, the point cloud is transformed to original location and the steps II, III, IV, V & VI are repeated

IX. The size of the hexagon (circumradius) is reduced by half in case the solution obtained for Fi is not better than F1 and steps II through VI are repeated

X. The same process is repeated till the size of the hexagon (circumradius) falls below 0.01% of the cylinder radius

The convergence criteria for the algorithm (0.01% of the F1 cylinder radius) can be modified as per the desired accuracy. In the case of the minimum circumscribed cylinder, the better solution would be a cylinder with a smaller radius. For a maximum inscribed cylinder, the better solution would be a cylinder with a bigger radius.

3.4 Representation of Fixture Method

The scope of the research is limited to machining parts using milling operation. Based on the common types of work holding devices on a milling machine, a fixture library has been developed that can be used to hold, locate and support the workpiece. The fixturing methods are broadly classified as:

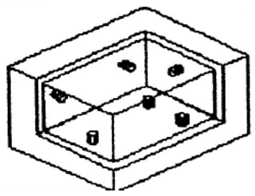
- 3-2-1 point locating (A1): Only planar surfaces are used for locating
- One plane & two pins locating (A2): One main locating plane surface is used for primary locating, and one inner cylindrical surface for inner locating
- Long-pin locating (A3): One inner cylindrical surface is used as the primary locating surface

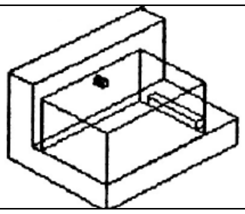
- V-block locating (A4): An external cylindrical surface is used in the primary location direction
- V-pad locating (A5): An external cylindrical surface is used as the secondary locating surface

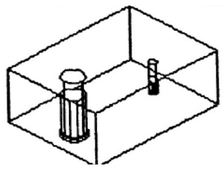
The fixture library used to develop algorithms is shown in Table 8. All variations in each locating method can be integrated into the algorithm with minor updates to the mathematical calculations.

Table 8: Fixture Library used in the algorithm

Method	Locating method name:	Deg. of freedom	Type of fitting : (MMC)	Picture

Method 1 (A1)	3-2-1	**Total:** 6
		Datum A: 3
		Datum B: 2
		Datum C: 1
			6 MMC fit datum
			A1:(one-sided planar fit)
			A2:(one-sided planar fit)
			A3:(one-sided planar fit)
			B1:(one-sided planar fit)
			B2:(one-sided planar fit)
			C:(one-sided planar fit)
				

	Plane-line-point (or 3 planes)	**Total:** 3
		Datum A: 1
		Datum B: 1
		Datum C: 1
			3 MMC fit datum
			A1:(one-sided planar fit)
			B1:(one-sided planar fit)
			C1:(one-sided planar fit)
				

Method 2 (A2)	Plane-pin-pin	**Total:**
		Datum A: 3
		Datum B: 2
		Datum C: 1
			3 MMC fit datum
			A:(one-sided planar fit)
			B:(inscribed cyl fit)
			C1:(inscribed cyl fit)
				

The fixturing method is specified in the CAD model using a text annotation. It is prefixed by *"fixture_type"* followed by a number that specifies the type of method, as shown in Table 7.

3.5 Datum Reference Frames (DRF)

Datums and DRFs are used as references for measurements; they specify the direction of measurement and set up coordinate systems. Datums are theoretically exact points, axes, lines, and planes. They are neither on the measured part nor on the gage blocks or inspection tooling; they are simulated by contact between the two. A DRF is a set of two or three mutually perpendicular features (planes, mid-planes, axes) that are derived from sufficient datum features or portions of them. Figure 18 illustrates the 3-2-1 principle for simulating a DRF with three mutually perpendicular datums. [8]

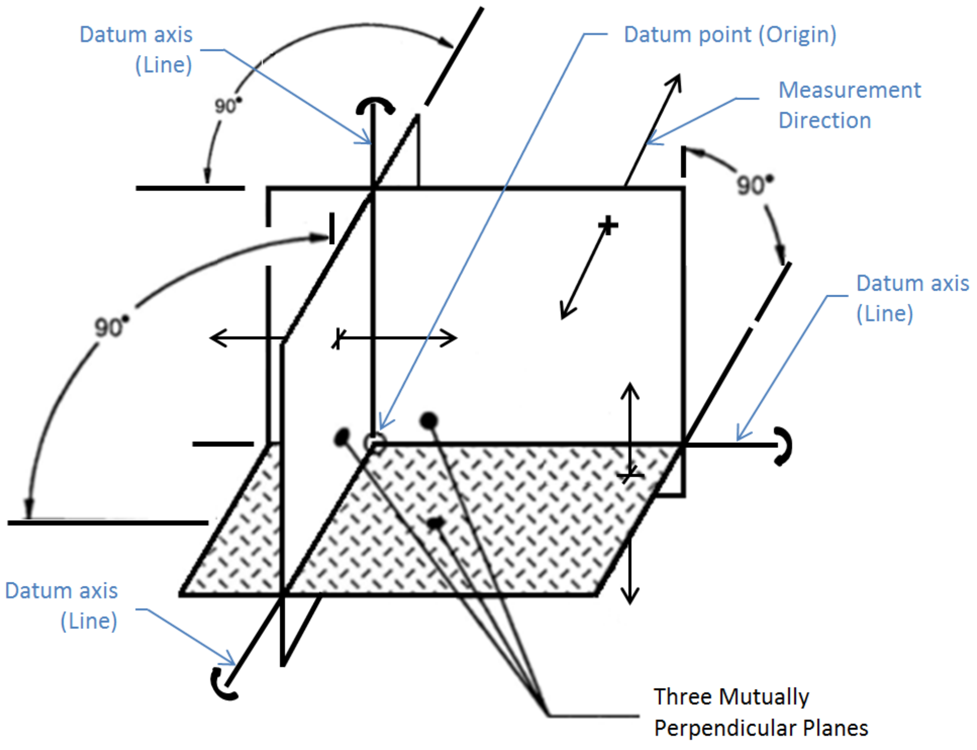


Figure 18: Illustration of 3-2-1 principle

Since some types of errors (e.g., orientation, position) need a datum as a reference, it is necessary to identify the datums first. And since, datums correspond to the way the part is located and fixed in each machine tool, it can be derived from the fixturing features and fixturing method in each setup.

3.6 Overview of Constraint Tolerance Feature Graph (CTF graph)

The Constraint Tolerance Feature graph[20] is an in-house, attributed model, developed at the Design Automation Lab at Arizona State University (ASU). A CTF Graph is a neutral representation of the CAD geometry in the form of features along with constraints and adds the tolerance values as attributes. The final output is a CTF graph file with a doubly linked list of related entities. The CTF graph provides a seamless interface to transfer PMI along with its parent geometry. The CTF graph includes nominal geometry (the features), related constraints (including dimensions & mating conditions if any), tolerances and degrees of freedom.

3.6.1 Real and Trimmed Features

A feature is a stereotypical shape defined by specific topology, geometry, and constraints.[8] A real feature may be of any shape and size. There are no abstracted primitives like a point feature, an infinite line or plane feature. The features represented in designs and CAD systems are an approximated or tolerance version of the real features called trimmed features.

To view the CTF graph, the user would like to use either the real feature or the trimmed ones, but not the primitives or their combination. Primitive geometries are defined as the simplest geometric objects that the system can handle (draw or store). In 2D, the primitive objects are points, lines or line segments, and polygons. A few examples of 3D primitive objects are a cube, cylinder, sphere, and torus.

3.6.2 Constraints and Metric Relationships

A geometric constraint in GD&T corresponds to a basic metric relationship between the primitives. Each metric relationship may be expressed in one or more analytical equations from the analytical geometry.

Different metric relationships exist between the primitives and constrain the DoFs of geometric entities with respect to each other. [21] We use the same representation i.e. (X_i, Y_i, Z_i) for representing primitive points, $A_iX + B_iY + C_iZ + D_i = 0$ for primitive planes and $(X-X_i)/p_i = (Y-Y_i)/q_i = (Z-Z_i)/r_i$ for primitive lines.

Geometric constraints can be dimensions, mating conditions or geometric relations, like perpendicularity or parallelism. Size dimensions may be directly attributed to the feature. A geometric constraint may have a direction but it is not always necessary. For example, if a plane is attributed with a constraint, the measurement direction is in the direction of the plane normal. Hence a geometric constraint, especially a dimension, requires a direction and that direction will be the same direction in which the constraint will be measured/inspected upon manufacturing. The designer has the ability to specify the direction of measurement by specifying the appropriate datum for this measurement.

3.6.3 General CTF Graph Structure and CTF Graph Sample

Figure 19 represents the structure of a CTF graph and Figure 20 represents a sample CTF graph.

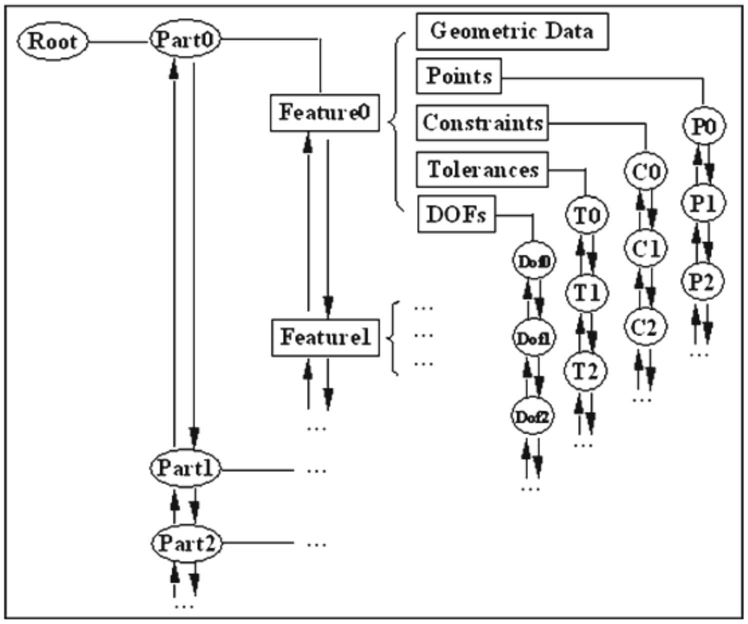
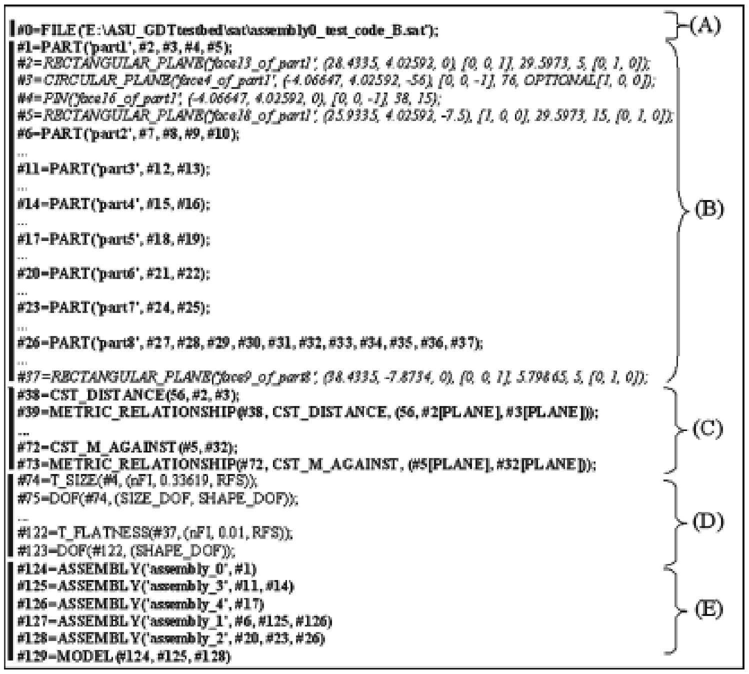


Figure 19: Structure of CTF Graph[20]



```
#0=FILE('E:\ASU_GDTtestbed\sat\assembly0_test_code_B.sat');
#1=PART('part1', #2, #3, #4, #5);
#2=RECTANGULAR_PLANE('face13_of_part1', (28.4335, 4.02592, 0), [0, 0, 1], 29.5973, 5, [0, 1, 0]);
#3=CIRCULAR_PLANE('face4_of_part1', (-4.06647, 4.02592, -56), [0, 0, -1], 76, OPTIONAL[1, 0, 0]);
#4=PIN('face16_of_part1', (-4.06647, 4.02592, 0), [0, 0, -1], 58, 15);
#5=RECTANGULAR_PLANE('face18_of_part1', (25.9335, 4.02592, -7.5), [1, 0, 0], 29.5973, 15, [0, 1, 0]);
#6=PART('part2', #7, #8, #9, #10);
...
#11=PART('part3', #12, #13);
...
#14=PART('part4', #15, #16);
...
#17=PART('part5', #18, #19);
...
#20=PART('part6', #21, #22);
...
#23=PART('part7', #24, #25);
...
#26=PART('part8', #27, #28, #29, #30, #31, #32, #33, #34, #35, #36, #37);
...
#37=RECTANGULAR_PLANE('face9_of_part8', (38.4335, -7.8734, 0), [0, 0, 1], 5.79865, 5, [0, 1, 0]);
#38=CST_DISTANCE(56, #2, #3);
#39=METRIC_RELATIONSHIP(#38, CST_DISTANCE, (56, #2[PLANE], #3[PLANE]));
...
#72=CST_M_AGAINST(#5, #32);
#73=METRIC_RELATIONSHIP(#72, CST_M_AGAINST, (#5[PLANE], #32[PLANE]));
#74=T_SIZE(#4, (nF1, 0.33619, RFS));
#75=DOF(#74, (SIZE_DOF, SHAPE_DOF));
...
#122=T_FLATNESS(#37, (nF1, 0.01, RFS));
#123=DOF(#122, (SHAPE_DOF));
#124=ASSEMBLY('assembly_0', #1)
#125=ASSEMBLY('assembly_3', #11, #14)
#126=ASSEMBLY('assembly_4', #17)
#127=ASSEMBLY('assembly_1', #6, #125, #126)
#128=ASSEMBLY('assembly_2', #20, #23, #26)
#129=MODEL(#124, #125, #128)
```

Figure 20: CTF Graph Sample[20]

The CTF graph file is divided into multiple sections (shown in Figure 20), each section representing a particular set of data. The description of the sections are as follows:

Section A: Name of the BRep file

Section B: Geometric information about all the parts

Section C: Constraints and metric relationship information

Section D: Tolerance information

Section E: Assembly hierarchy information

This is a robust model that can be used to transfer GD&T information to other modules with ease. And, since it contains limited information (details pertaining to the tolerances that are required to be transferred), the CTF graph file is compact too. A CTF graph file parser is a standard parser developed at ASU which can be used to read the data stored in the same.

CHAPTER 4

ALGORITHMS & IMPLEMENTATION

Since AP203 does not yet support PMI data, such as GD&T, the information is extracted from the CREO (*.prt) file using Spatial's CAD kernel, ACIS, and its CAD translator, InterOp. The CAD geometry along with the PMI information is translated to an ACIS (*.sat) file. The PMI information is extracted from the ACIS file into a separate text file which is later parsed in order to populate a custom data structure that stores the required information.

Point cloud data is generated by scanning the casting using a laser scanner, for each datum and TBM feature and stored in separate ASCII (*.asc) files. The location, position, and orientation of the datum and TBM features on the casting are determined by fitting appropriate features to the point cloud data. Once the details pertaining to these features are obtained, a DRF has to be constructed called GDRF, based on the fixturing type which in turn is specified in the CAD model. Once the GDRF is constructed, all the entities are transformed accordingly into the GDRF. These details are further written out in a modified CTF graph which can further be used to compute the fixture adjustments. Figure 21 shows the flow chart describing the entire process.

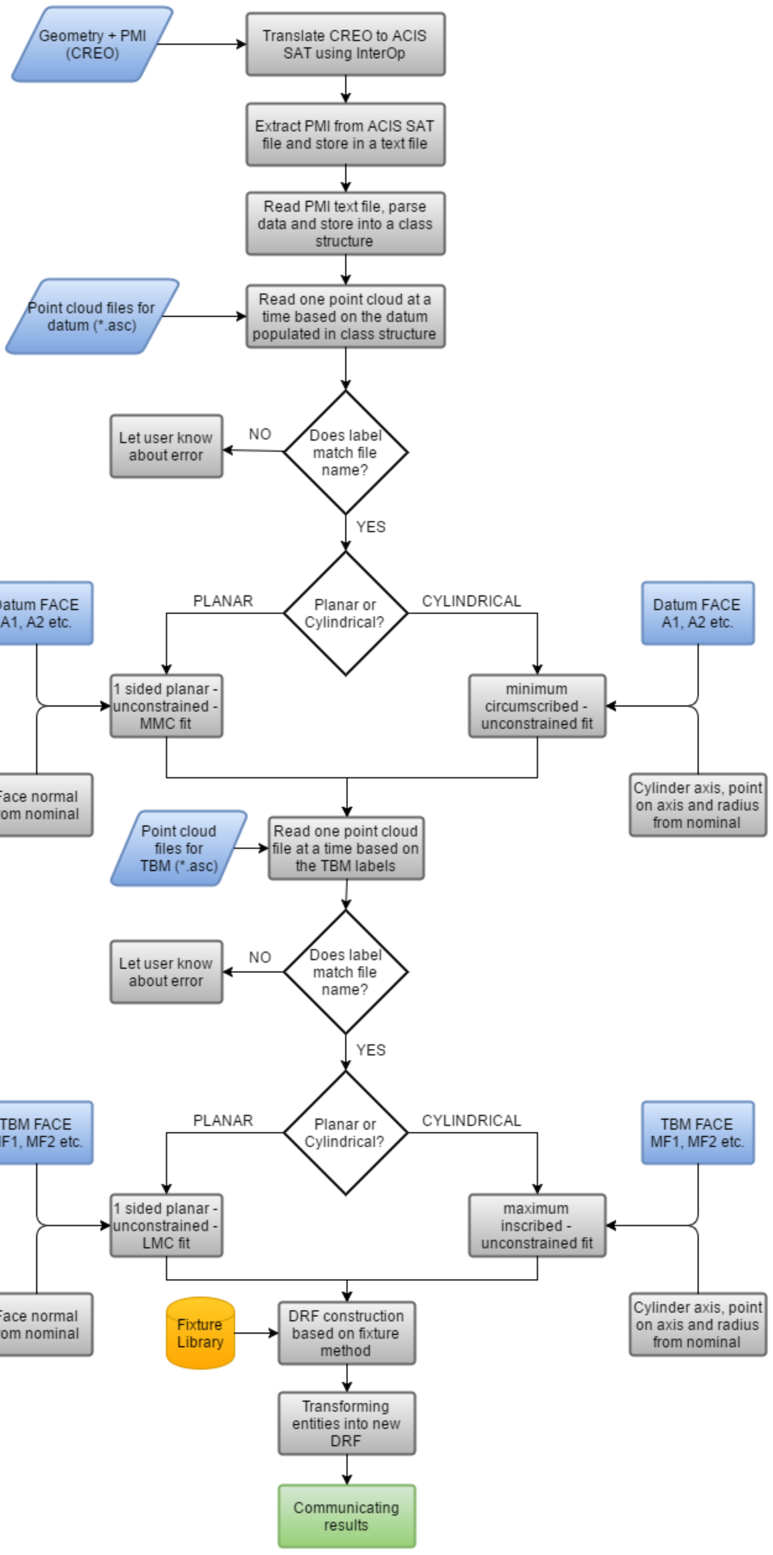

Geometry + PMI (CREO)
Translate CREO to ACIS SAT using InterOp
Extract PMI from ACIS SAT file and store in a text file
Read PMI text file, parse data and store into a class structure
Point cloud files for datum (*.asc)
Read one point cloud at a time based on the datum populated in class structure
Let user know about error
NO
Does label match file name?
YES
Datum FACE A1, A2 etc.
PLANAR
Planar or Cylindrical?
CYLINDRICAL
Datum FACE A1, A2 etc.
1 sided planar - unconstrained - MMC fit
minimum circumscribed - unconstrained fit
Face normal from nominal
Cylinder axis, point on axis and radius from nominal
Point cloud files for TBM (*.asc)
Read one point cloud file at a time based on the TBM labels
Let user know about error
NO
Does label match file name?
YES
TBM FACE MF1, MF2 etc.
PLANAR
Planar or Cylindrical?
CYLINDRICAL
TBM FACE MF1, MF2 etc.
1 sided planar - unconstrained - LMC fit
maximum inscribed - unconstrained fit
Face normal from nominal
Cylinder axis, point on axis and radius from nominal
Fixture Library
DRF construction based on fixture method
Transforming entities into new DRF
Communicating results

Figure 21: Flow Chart of the Process

4.1 CAD Data Exchange

The 3D ACIS® Modeler (ACIS) is a 3D geometric modeling kernel developed by Spatial Corporation. With the use of ACIS, the underlying 3D modeling functionality can be put to great use. It features an open, object-oriented C++ architecture that enables robust, 3D modeling capabilities. [22]

ACIS is designed using a software component technology which makes it a collection of software items like functions, classes and so on, grouped to perform a particular task. It enables the use of various individual components instead of using the whole software package. ACIS, on a Windows operating system, is compiled and executed using Microsoft Visual Studio. There are three core functionalities of ACIS: 3D modeling, 3D model management, and 3D model visualization.

ACIS saves the information pertaining to a model in external files that have a neutral format hence enabling various CAD systems to access the information. For a CAD system to understand and read an ACIS file, the CAD system must recognize the structure of the file, data encapsulation, types of data and references. There are two kinds of ACIS save files, a Standard ACIS Text (*.sat) and a Standard ACIS Binary (*.sab). An ACIS (*.sat) file is a human readable text file that can be opened using a regular text editor while a *.sab is a binary file that is only machine readable.

The structure of an ACIS save file is as follows:

- A three line header
- Entity records
- Begin history data marker*
- Old entity records required for history and roll back*
- End history data marker*
- An end marker

* Optional: User may choose to include this information while writing out a SAT file

It is also required for a SAT file to include the product ID and the units in which the CAD model is saved.

4.1.1 Extracting GD&T information using ACIS/InterOp

The CAD model along with the semantic PMI information serves as the first input. The PMI information attributed to the CAD model contains information like datum/datum targets, dimensions, size tolerances and geometric tolerances. The pseudo code for translating the CAD file from CREO to ACIS entities is:

I. A source object with the input file name as parameter is created

II. The destination object (ACIS entity_list type) is created to store the data after conversion

III. An object for unit value is created to store the default units of the CAD model

IV. A representation object with "BRep+PMI" as parameter is created

V. The options are required to be set before the conversion can take place

VI. Create a log file to note the status of the conversion process

VII. An object converter is created to set the required options and convert the source file into the destination format

VIII. The conversion is logged by calling the StartLog function with the log file as parameter

IX. The convert function is called with source and destination variables as parameters and store the return value in object result which is of type SPAIResult.

X. The logging is stopped by calling the StopLog function with the log file as parameter

XI. GetEntities function is used to access files from object destination and store them in a list

4.1.2 Parsing PMI data into Data Structures

The PMI information extracted along with the BRep data from the input CREO file is stored into a *.sat file. The geometric modeler kernel, ACIS, extracts the PMI information from the *.sat file, and stores it in a human readable text file, which is further processed through a parser that can extract GD&T information specific to the datum and TBM features. The owner associated with each attribute is represented and recognized with an owner id. Single digit numbers are associated with the tolerance type, representing the type of tolerance and its material modifier. Table 9 represents various geometric tolerance types and Table 10 shows the material modifiers or tolerance sub-type.

Table 9: List of Geometric Tolerance Types and Tolerance Numbers

Tolerance Number	**Tolerance Type**
0	UNKNOWN
1	STRAIGHTNESS
2	FLATNESS
3	CIRCULARITY
4	CYLINDRICITY
5	PROFILE_LINE
6	PROFILE_SURF
7	COMP[*]_PROFILE_SURF
8	PARALLELISM
9	PERPENDICULARITY
10	ANGULARITY
11	POSITION
12	COMP[*]_POSITION
13	CONCENTRICITY
14	SYMMETRY
15	RUNOUT_CIRCULAR
16	RUNOUT_TOTAL

[*] Composite

Table 10: Material Modifiers and corresponding numbers

Modifier Number	**Material Modifier**
0	NONE
1	MMC
2	RFS
3	LMC
4	FS
5	TP
6	ST

The extraction of PMI information from the text file is done as follows:

I. The geometry is queried to obtain all the faces along with its face id

II. An *std::pair* data structure is created using the *FACE* entity and its corresponding face id

III. The PMI text file is parsed, line by line, using the parser to identify specific keywords like *"Datum attribute found", "Text attribute found", "Geomtol attribute found", "Dimension found"* and so on.

IV. Based on the type of attribute found, the data searched and stored are different

V. For every attribute, the common information searched for and extracted are the "*Owner"* type (example face or edge) and its corresponding id. The attribute is matched to its owner using the pair data structure created earlier

VI. In case of a datum attribute, the “*Label*” of the datum is also stored that denotes the datum name and will be used further to identify the point cloud data file

VII. The text attribute contains a field called *“Text”* that contains the value of the text specified in the CAD model (the text attribute MUST have a leader pointing to at least of the entities on the CAD model)

VIII. For a geometric tolerance attribute, the tolerance type, tolerance magnitude and zone modified (material modifier) and associated datum (if any) are extracted. The tolerance attribute is specified as an integer, which is later decoded to its corresponding tolerance text based on table 8

IX. For a size dimension, the dimension type, dimension sub-type (material modifier), dimension value and upper/lower tolerances are extracted

X. A position tolerance is shown as a geometric tolerance with tolerance type code “2”. In this case, the primary, secondary and tertiary (if any) are also extracted

XI. The data extracted are stored in a class structure and linked accordingly to the corresponding tolerance types and its values

XII. Any other data like *“Roughness attribute found”* and so on can also be added to the parser and data can be extracted with ease

The information stored in this data structure is a custom data structure and may be modified or updated based on the requirements. This data structure is then fed into the CTF writer, which writes out a modified CTF graph.

4.2 Feature Fitting

The features are fit to the point cloud based on the tag (Datum/Machined/Critical) associated with it. The features are fit as per these rules:

- For a planar datum feature, an unconstrained one-sided planar feature is fit to the point cloud data, at its MMC
- For a cylindrical datum feature, an unconstrained maximum inscribed cylinder is fit to the point cloud data, at the MMC of the feature
- For a planar machined feature, an unconstrained one-sided planar feature is fit to the point cloud data, at its LMC
- For a cylindrical machined feature, an unconstrained minimum circumscribed cylinder is fit to the point cloud data, at the LMC of the feature

For a datum feature, the feature is fitted at the MMC condition with respect to the geometric feature, since contact with a fixture would occur at the outermost points on the datum surface. However, a machined feature, the feature is fitted at the LMC since the produced feature should have all pits (low regions) machined away. Figure 22 explains the need for fitting features at MMC and LMC locations. The figure on the left shows the location where the fixture touches the datum i.e. on the highest points on the surface while the figure on the right explains how fitting a feature at LMC would get rid of the surface inaccuracies and help obtain a finished surface. Also, the reason for fitting an unconstrained feature is to obtain the true position of the feature on the part since a constrained fit might alter the actual position.

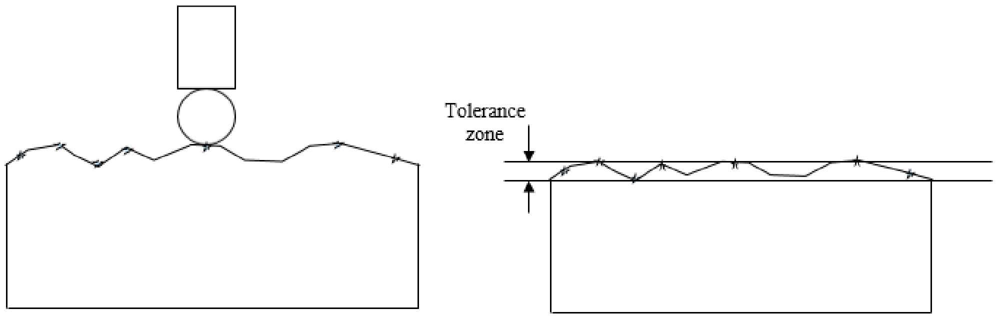


Figure 22: L→R: Fixture coming in contact with part, Tolerance zone on a TBM feature

4.2.1 Improvements to Normative Feature Fitting Algorithms

The feature fitting algorithms from the nFF library, for fitting planar and cylindrical features, have been modified to generalize, improve efficiency and make it more robust. The improvements done to the planar and cylindrical feature fitting algorithms are discussed separately below.

In the case of a planar fit, as discussed in the previous chapter, the initial direction of fit is derived from the CAD geometry. The initial direction of fit derived from the nominal CAD geometry may sometimes not accurately represent the orientation of the actual face on the part, and may constrain it in the nominal direction. Figure 23 explains the possibility of this error being induced in the feature being fit. This error arises even when the face normal of only one plane in the convex hull is in the same direction of the initial direction. This leads to the selection of the wrong plane as the final fit.

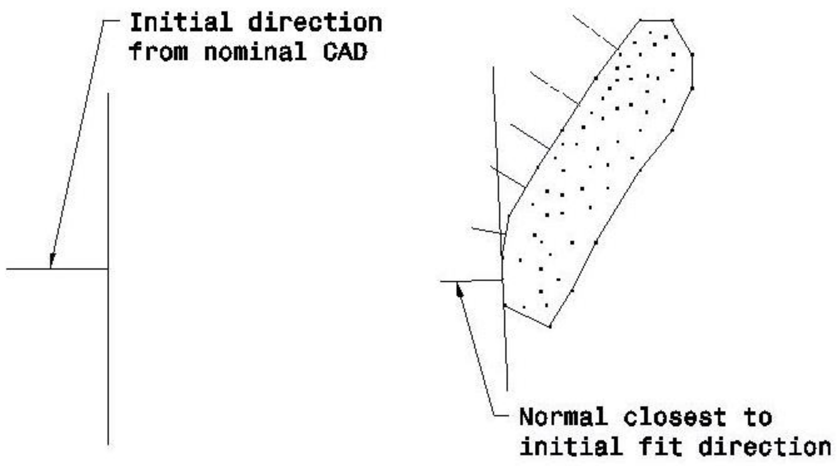

Initial direction from nominal CAD
Normal closest to initial fit direction

Figure 23: Error Induced due to Direction from Nominal CAD

Instead, a least square fit is done to the point cloud and the face normal of the least square fit is treated as the initial direction of fit (Figure 24). This method better represents the actual surface obtained from the point cloud data.

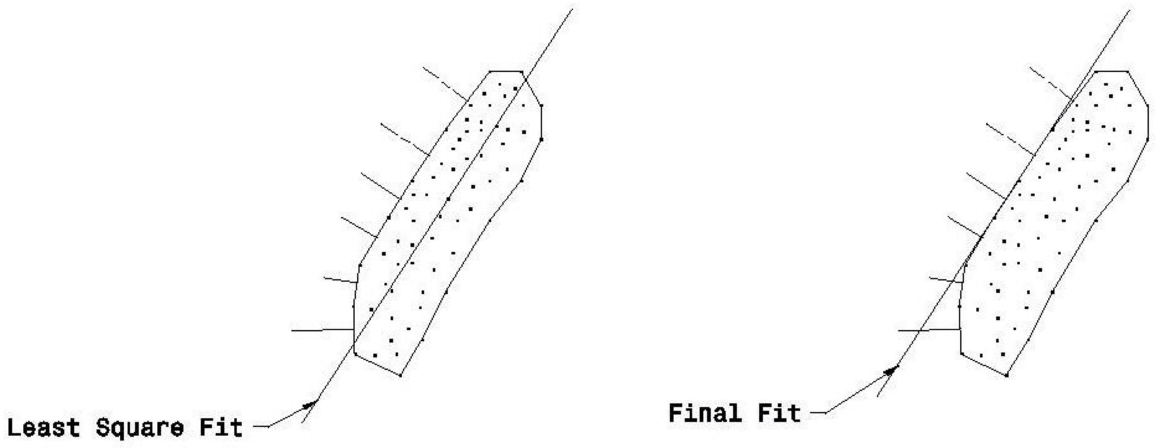


Figure 24: L→R: Least Square Fit, Final One-sided Fit

When fitting a feature to a point cloud data of a cylindrical surface, the algorithm begins with fitting a cylinder using the least square method. The axis of this cylinder is rotated such that it is along the z-axis (as described in step III of unconstrained cylindrical feature fitting algorithm). The axis vector is multiplied by the corresponding rotation matrix. But the existing algorithm performed best when the axis was already positioned along the z-axis, thereby limiting the use of the algorithm to cylinders oriented in other directions (the cylinder axis not aligned or parallel to x, y or z-axes).

Instead of rotating all the points in the point cloud to orient them about a fixed 'z-axis', a new axis (vector) is created using the following simple linear algebraic method:

I. The initial cylinder axis (R1) obtained from the least square fit is stored in a vector and a random vector is generated

II. The R1 is projected onto the random vector. This projected vector is then subtracted from the random vector to obtain R_rand

III. Vectors R1 & R_rand are normalized and then cross multiplied in order to obtain vector R2

IV. The cross product of R1 and R2 produces R3

V. The normalized form of all three vectors R1, R2 & R3 form the three rows of the rotation matrix (U)

This method does not require the point cloud to be in a fixed direction like x-axis or y-axis. Irrespective of the orientation of the point cloud, this vector algebra method is robust at transforming the point cloud data into one particular axis.

4.2.2 Feature Fitting Results

The results from the feature fitting algorithms are written out into individual files (named same as the datum/face label). The result from every planar fit is a point on the plane and its face normal while the cylindrical fit gives the cylinder axis, two points on the axis of the cylinder and the radius of the fitted cylinder. These results are further transformed to a different datum reference frame, if required, as per the transformation matrix obtained from the locating method (discussed in the next section).

4.3 Global Datum Reference Frame

The existence of multiple coordinate measurement frames leads to a need for excessive back-and-forth transformation of entities. The various coordinate frames possible are the coordinate frame of the CAD model, the coordinate frame of the point cloud and so on. Hence it is important for all the entities to be represented in the same coordinate frame. This unique coordinate reference frame is called as global datum reference frame (GDRF). The GDRF is unique to the locating method used as well as unique to each part to be machined.

The GDRF is constructed based on the locating method selected for fixing the part prior to machining. The steps involved in the creation of the GDRF and performing the required transformations to entities are discussed in the coming sections.

4.3.1 Construction of Global Datum Reference Frame based on Locating Methods

The construction of a Global Datum Reference Frame (GDRF) is based on the type of location method used to fix the part prior to machining. The steps involved in creating a DRF from the features being fit at the fixture locations are described below:

For 3-2-1 method:

I. The feature fitting algorithm is applied to the point cloud data at six locating methods, say A1, A2, A3, B1, B2 and C

II. The points for A1, A2 and A3 are used to construct a single plane called plane A. The normal to plane A (A_vector) is parallel to one of the axes of the coordinate system

III. The cross product of the face normal of plane A and vector obtained from points B1 and B2 gives a vector (B_vector) which is parallel to the second axis of the coordinate system

IV. The B_vector and point obtained on B1 gives the second plane, while the point and face normal of C give plane C (whose face normal is parallel to the third axis)

V. The intersection of planes A, B, and C generates the vector representing the origin of the coordinate system

All these values of vectors and points are in the existing DRF. These values are then used to construct a transformation matrix, which is applied to CAD geometry and all the feature fits to transform them into the GDRF. Figure 25 shows the 6 datum targets (A1, A2, A3, B1, B2, and C) and the planes created from them (A, B, C), with their face normal. The letter ‘O’ in the figure denotes the origin.

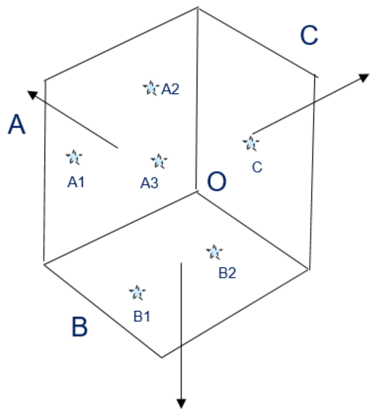


Figure 25: GDRF for 3-2-1 method

For 3-planes method (variation of 3-2-1):

I. For a 3-plane method, the feature fitting algorithms are applied to the point cloud data provided at three locations for example A, B, and C

II. The face normal of each of the planes serves as the three axes of the coordinate system

III. The intersection of the three planes gives the origin

A transformation matrix is constructed from these values which are used later to transform the entities into GDRF. Figure 26 shows the 3 datum targets (A, B, and C) and their face normal. The letter 'O' in the figure denotes the origin.

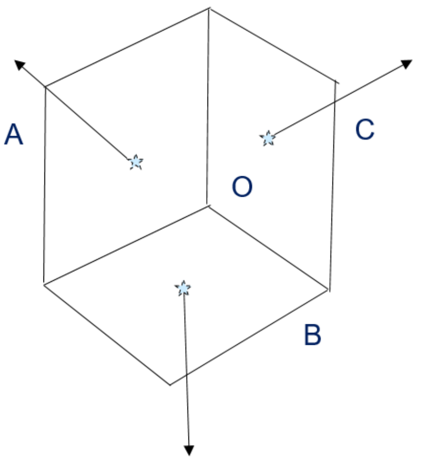


Figure 26: GDRF for 3 planes method

For one plane and two pins method:

I. The axis of both the cylindrical pins are calculated from the feature fitting algorithms and the resultant is computed. This resultant (X_vector) represents one of the axes of the coordinate system

II. A plane is fit to the point cloud data of the plane, say A

III. The cross product of the face normal of A and the X_vector gives the second axis (Y_vector) of the coordinate system

IV. The third axis is obtained by the cross product of X_vector and Y_vector

V. The point on the plane A obtained from the feature fitting algorithms is the origin of the coordinate system

The direction pertaining to each vector, which denotes either the positive or negative axis direction, can be selected once the computations are completed and the coordinate system set up. Upon reversing the sign, the coordinate axis is flipped and the transformations are also changed accordingly. Figure 27 shows the two cylinder axis (axis_1 & axis_2) and one planar datum target (plane A). Vectors x_vec, y_vec and z_vec represent the coordinate axes. The letter 'O' in the figure denotes the origin.

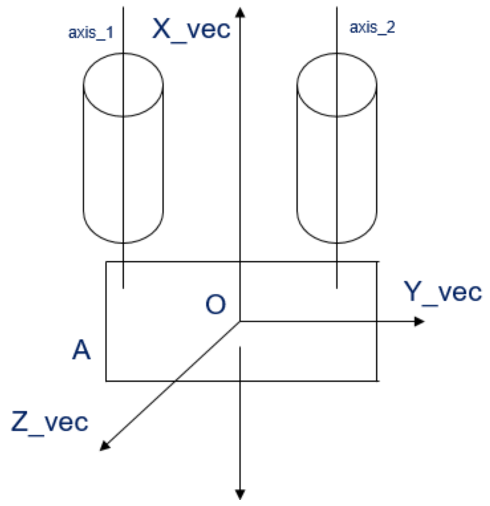


Figure 27: GDRF for one plane two pins method

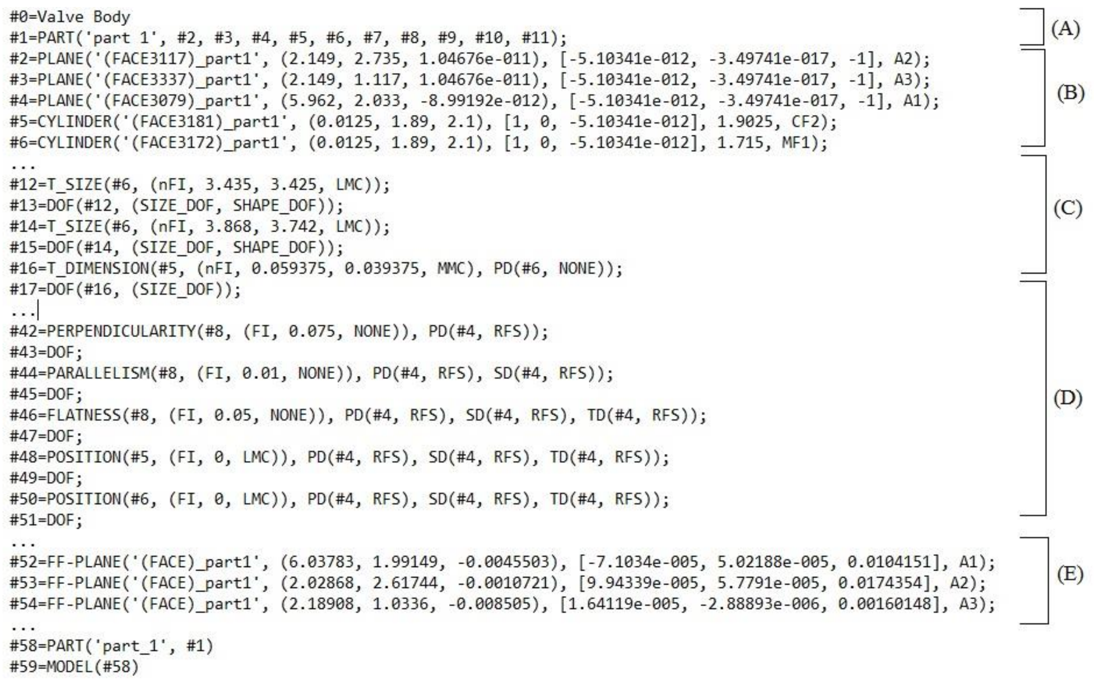
4.3.2 Transforming CAD Data & Feature Fits into GDRF

The transformation matrix multiplied to the fitted features and CAD model consists of two types of transformation, (i) rotation about x, y and z-axes and (ii) translation with respect to the origin. The transformation matrix is a 4×4 matrix where the first three rows and columns represent the rotation vectors for x, y and z-axes (represented by three column vectors) respectively and the fourth column represents the translation with respect to the origin. The fourth row in the matrix is used to specify the scaling factor (if any) of the entities during the transformation. Every single part has a set of point cloud data from which the GDRF is constructed and to which the entities are transformed. Upon completion of the transformation, an ACIS (*.sat) file is written out to view the nominal CAD geometry in the newly created GDRF. Upon visualizing, if the coordinate axis directions are to be reversed, the same can be done by simply specifying it during the GDRF creation.

4.4 Communicating Results (MCTF Graph)

The PMI information that is parsed and stored in a data structure along with the feature fitting results are used to write the MCTF graph. The MCTF graph writer writes the set of datum and TBM features in a part based on the datum/face label and assigns appropriate line numbers. Once the features are written, the tolerances are added to the graph along with its tolerance magnitudes, datum references, and other details. Every tolerance has a designated owner which is called out by the line number in which the feature was initially listed. After the set of size and geometric tolerances are inserted, the feature fitting data transformed into the GDRF, is added to the MCTF graph. These features are prefixed by “FF-“ denoting them as feature fit data. The MCTF graph is further communicated to perform additional computations to arrive at the most optimum fixture setup solution.

The Machining Constraint Tolerance Feature (M-CTF) Graph has a similar structure as a general CTF, but with additional data pertaining to the faces that are to be machined. This includes a point on the feature and an axis (face normal for a plane and cylinder axis for a cylinder). In addition, for planar machined features, a set of points are included in the MCTF to define the boundary while a cylindrical feature has the radius specified. Figure 28 shows a sample MCTF graph.



```
#0=Valve Body
#1=PART('part 1', #2, #3, #4, #5, #6, #7, #8, #9, #10, #11);
#2=PLANE('(FACE3117)_part1', (2.149, 2.735, 1.04676e-011), [-5.10341e-012, -3.49741e-017, -1], A2);
#3=PLANE('(FACE3337)_part1', (2.149, 1.117, 1.04676e-011), [-5.10341e-012, -3.49741e-017, -1], A3);
#4=PLANE('(FACE3079)_part1', (5.962, 2.033, -8.99192e-012), [-5.10341e-012, -3.49741e-017, -1], A1);
#5=CYLINDER('(FACE3181)_part1', (0.0125, 1.89, 2.1), [1, 0, -5.10341e-012], 1.9025, CF2);
#6=CYLINDER('(FACE3172)_part1', (0.0125, 1.89, 2.1), [1, 0, -5.10341e-012], 1.715, MF1);
...
#12=T_SIZE(#6, (nFI, 3.435, 3.425, LMC));
#13=DOF(#12, (SIZE_DOF, SHAPE_DOF));
#14=T_SIZE(#6, (nFI, 3.868, 3.742, LMC));
#15=DOF(#14, (SIZE_DOF, SHAPE_DOF));
#16=T_DIMENSION(#5, (nFI, 0.059375, 0.039375, MMC), PD(#6, NONE));
#17=DOF(#16, (SIZE_DOF));
...
#42=PERPENDICULARITY(#8, (FI, 0.075, NONE)), PD(#4, RFS));
#43=DOF;
#44=PARALLELISM(#8, (FI, 0.01, NONE)), PD(#4, RFS), SD(#4, RFS));
#45=DOF;
#46=FLATNESS(#8, (FI, 0.05, NONE)), PD(#4, RFS), SD(#4, RFS), TD(#4, RFS));
#47=DOF;
#48=POSITION(#5, (FI, 0, LMC)), PD(#4, RFS), SD(#4, RFS), TD(#4, RFS));
#49=DOF;
#50=POSITION(#6, (FI, 0, LMC)), PD(#4, RFS), SD(#4, RFS), TD(#4, RFS));
#51=DOF;
...
#52=FF-PLANE('(FACE)_part1', (6.03783, 1.99149, -0.0045503), [-7.1034e-005, 5.02188e-005, 0.0104151], A1);
#53=FF-PLANE('(FACE)_part1', (2.02868, 2.61744, -0.0010721), [9.94339e-005, 5.7791e-005, 0.0174354], A2);
#54=FF-PLANE('(FACE)_part1', (2.18908, 1.0336, -0.008505), [1.64119e-005, -2.88893e-006, 0.00160148], A3);
...
#58=PART('part_1', #1)
#59=MODEL(#58)
```

Figure 28: Sample MCTF Graph

CHAPTER 5

VALIDATION & VERIFICATION

The algorithms proposed in this research work were validated and verified for accuracy and robustness with the help of a sample part. Robustness was measured by using the data obtained for multiple metal castings, of the same demonstration article. The obtained values for the location of GDRF and fitted features were then verified manually in a commercial CAD system to verify the results for accuracy.

5.1 Test Case I

The demonstration article was provided by PDA (LLC). It consisted the CAD geometry with GD&T information for it, in CREO (*.prt) format and the point cloud data associated with each feature in ASCII file format (*.asc). The input CAD geometry with GD&T information for it is shown in Figure 29 (a & b).

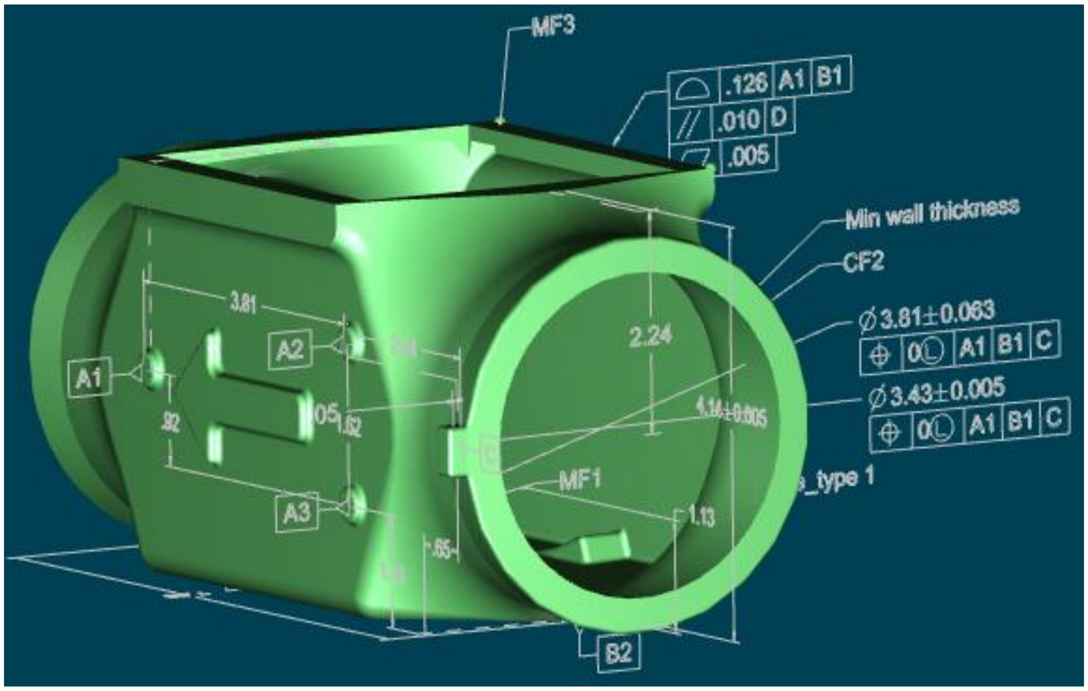


Figure 29 (a): CAD geometry with GD&T information

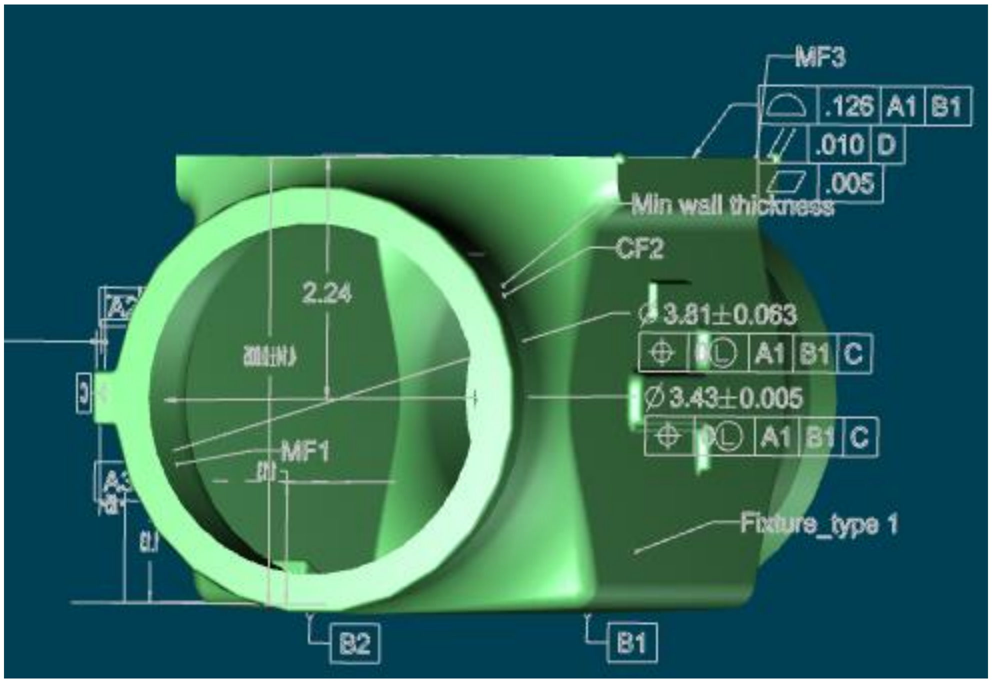
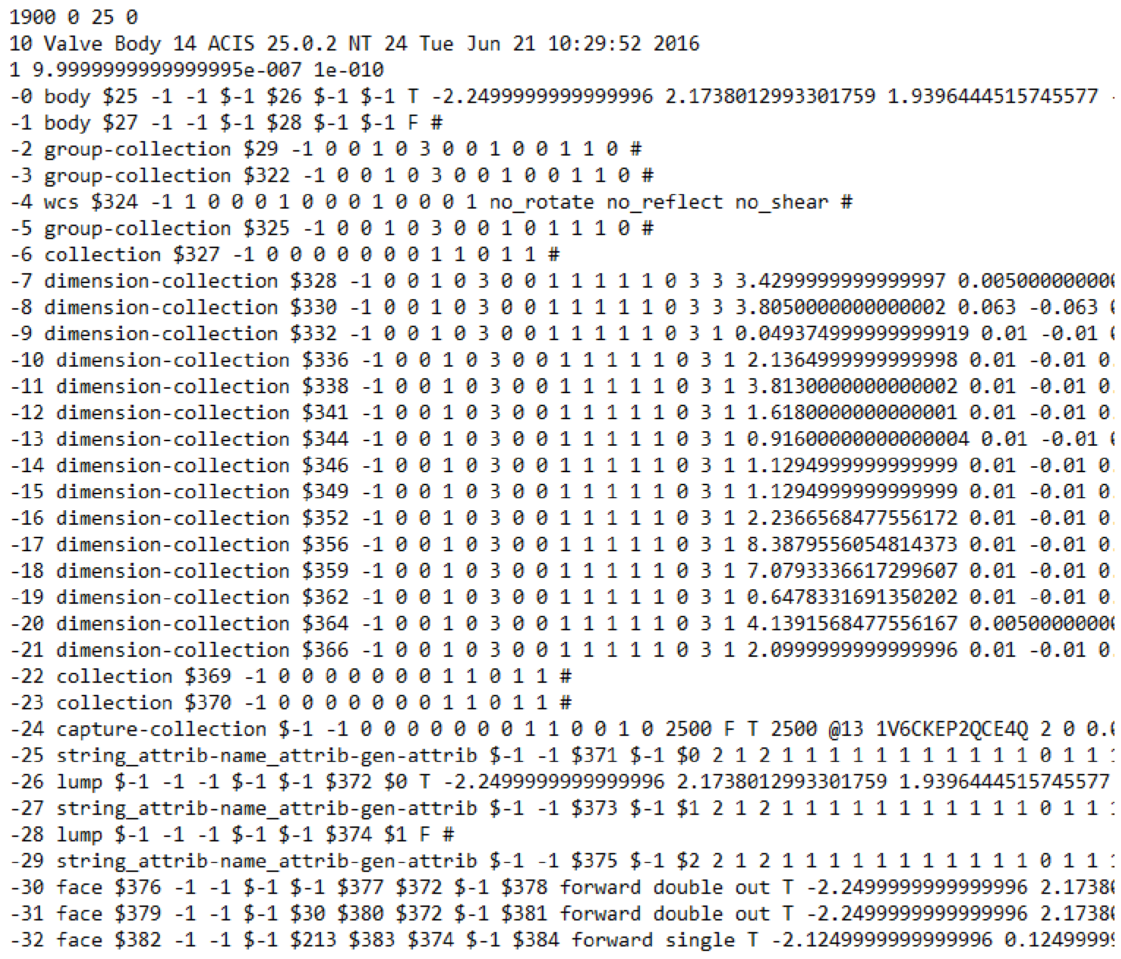


Figure 29 (b): CAD geometry with GD&T information

In this part, the critical features are the six datum targets (A1, A2, A3, B1, B2, & C), two machined features (MF1 & MF3) and one critical feature (CF2). The point cloud data for these critical features are saved with the same filename as the datum/face labels assigned to it in the CAD model. The datum/face label has to match the point cloud file name in order to extract the correct point cloud to perform the feature fitting. The folder structure for each test case is present and must remain the same. The folder structure is as shown below:

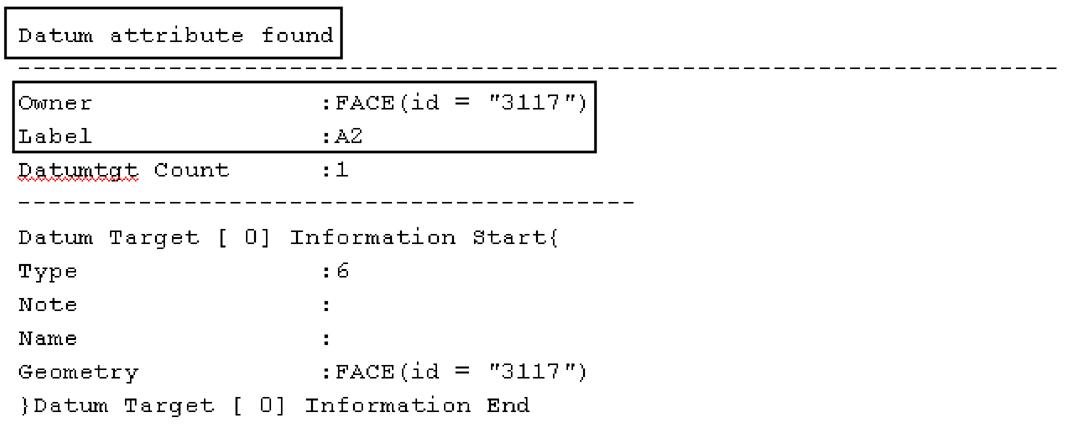
```
Project Folder (inside Documents)
    Input file (CREO *.prt file)
    Point Cloud Data
        SN<number>
            Datums
                A1.asc
                B1.asc
                …
            Machined Features
                MF1.asc
                CF2.asc
                …
```

The CAD file is read into the system and translated to ACIS (*.sat) using InterOp. The options used in the translation are very important and must specify what details are to be extracted from the input file. In this case, the option for representation was set to "BRep+PMI". The output file name is the same as input file name with "*.sat" concatenated at the end. Figure 30 shows a part of the ACIS (*.sat) file. This file is later read again and all the PMI information is extracted and dumped into a text file. Figure 31 (a, b & c) shows an extract from the PMI dump text file.



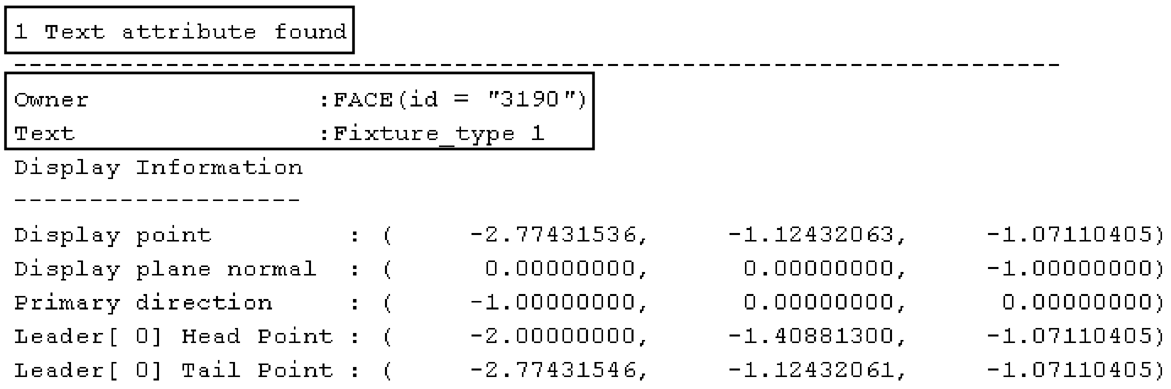
```
1900 0 25 0
10 Valve Body 14 ACIS 25.0.2 NT 24 Tue Jun 21 10:29:52 2016
1 9.9999999999999995e-007 1e-010
-0 body $25 -1 -1 $-1 $26 $-1 $-1 T -2.2499999999999996 2.1738012993301759 1.9396444515745577
-1 body $27 -1 -1 $-1 $28 $-1 $-1 F #
-2 group-collection $29 -1 0 0 1 0 3 0 0 1 0 0 1 1 0 #
-3 group-collection $322 -1 0 0 1 0 3 0 0 1 0 0 1 1 0 #
-4 wcs $324 -1 1 0 0 0 1 0 0 0 1 0 0 0 1 no_rotate no_reflect no_shear #
-5 group-collection $325 -1 0 0 1 0 3 0 0 1 0 1 1 1 0 #
-6 collection $327 -1 0 0 0 0 0 0 0 1 1 0 1 1 #
-7 dimension-collection $328 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 3 3.4299999999999997 0.00500000000000
-8 dimension-collection $330 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 3 3.8050000000000002 0.063 -0.063
-9 dimension-collection $332 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 0.049374999999999919 0.01 -0.01
-10 dimension-collection $336 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 2.1364999999999998 0.01 -0.01 0
-11 dimension-collection $338 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 3.8130000000000002 0.01 -0.01 0
-12 dimension-collection $341 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 1.6180000000000001 0.01 -0.01 0
-13 dimension-collection $344 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 0.91600000000000004 0.01 -0.01
-14 dimension-collection $346 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 1.1294999999999999 0.01 -0.01 0
-15 dimension-collection $349 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 1.1294999999999999 0.01 -0.01 0
-16 dimension-collection $352 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 2.2366568477556172 0.01 -0.01 0
-17 dimension-collection $356 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 8.3879556054814373 0.01 -0.01 0
-18 dimension-collection $359 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 7.0793336617299607 0.01 -0.01 0
-19 dimension-collection $362 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 0.6478331691350202 0.01 -0.01 0
-20 dimension-collection $364 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 4.1391568477556167 0.00500000000
-21 dimension-collection $366 -1 0 0 1 0 3 0 0 1 1 1 1 1 0 3 1 2.0999999999999996 0.01 -0.01 0
-22 collection $369 -1 0 0 0 0 0 0 0 1 1 0 1 1 #
-23 collection $370 -1 0 0 0 0 0 0 0 1 1 0 1 1 #
-24 capture-collection $-1 -1 0 0 0 0 0 0 0 1 1 0 0 1 0 2500 F T 2500 @13 1V6CKEP2QCE4Q 2 0 0.
-25 string_attrib-name_attrib-gen-attrib $-1 -1 $371 $-1 $0 2 1 2 1 1 1 1 1 1 1 1 1 1 1 0 1 1
-26 lump $-1 -1 -1 $-1 $-1 $372 $0 T -2.2499999999999996 2.1738012993301759 1.9396444515745577
-27 string_attrib-name_attrib-gen-attrib $-1 -1 $373 $-1 $1 2 1 2 1 1 1 1 1 1 1 1 1 1 1 0 1 1
-28 lump $-1 -1 -1 $-1 $-1 $374 $1 F #
-29 string_attrib-name_attrib-gen-attrib $-1 -1 $375 $-1 $2 2 1 2 1 1 1 1 1 1 1 1 1 1 1 0 1 1
-30 face $376 -1 -1 $-1 $-1 $377 $372 $-1 $378 forward double out T -2.2499999999999996 2.1738
-31 face $379 -1 -1 $-1 $30 $380 $372 $-1 $381 forward double out T -2.2499999999999996 2.1738
-32 face $382 -1 -1 $-1 $213 $383 $374 $-1 $384 forward single T -2.1249999999999996 0.1249999
```

Figure 30: Partial ACIS (*.sat) file for test part



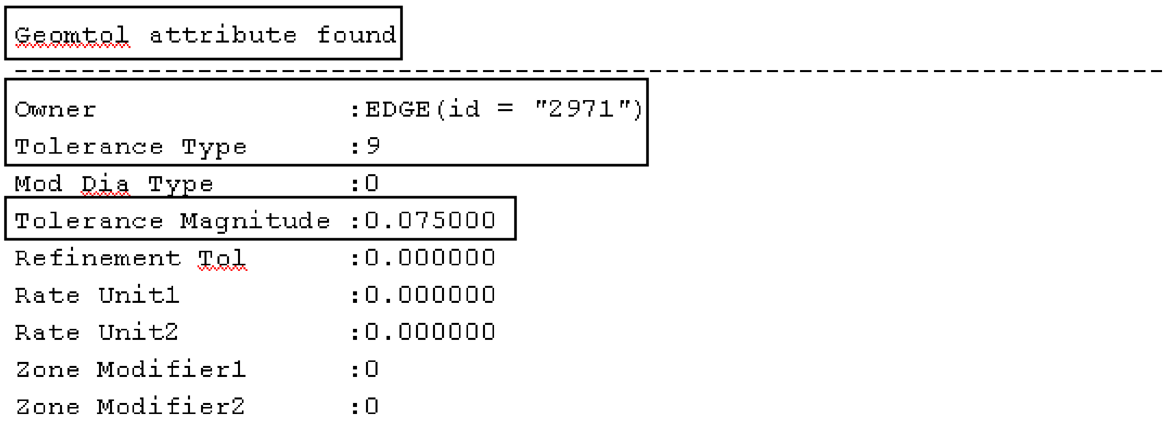
```
Datum attribute found
---------------------------------------------------------------------
Owner                   :FACE(id = "3117")
Label                   :A2
Datumtgt Count          :1
-----------------------------------------
Datum Target [ 0] Information Start{
Type                    :6
Note                    :
Name                    :
Geometry                :FACE(id = "3117")
}Datum Target [ 0] Information End
```

Figure 31 (a): Extract from PMI text file of Datum attribute



```
1 Text attribute found
---------------------------------------------------------------------
Owner                   :FACE(id = "3190")
Text                    :Fixture_type 1
Display Information
-------------------
Display point           : (     -2.77431536,     -1.12432063,     -1.07110405)
Display plane normal    : (      0.00000000,      0.00000000,     -1.00000000)
Primary direction       : (     -1.00000000,      0.00000000,      0.00000000)
Leader[ 0] Head Point   : (     -2.00000000,     -1.40881300,     -1.07110405)
Leader[ 0] Tail Point   : (     -2.77431546,     -1.12432061,     -1.07110405)
```

Figure 31 (b): Extract from PMI text file of Text attribute

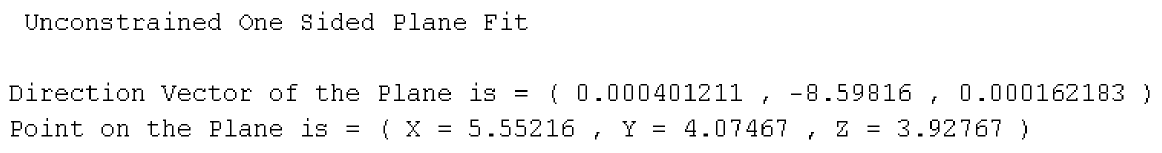


```
Geomtol attribute found
---------------------------------------------------------------------
Owner                   :EDGE(id = "2971")
Tolerance Type          :9
Mod Dia Type            :0
Tolerance Magnitude     :0.075000
Refinement Tol          :0.000000
Rate Unit1              :0.000000
Rate Unit2              :0.000000
Zone Modifier1          :0
Zone Modifier2          :0
```

Figure 31 (c): Extract from PMI text file of tolerance attribute

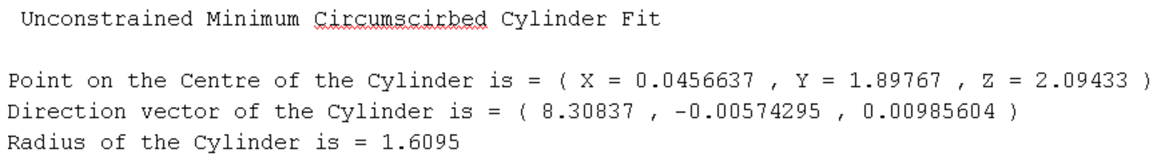
The data from the PMI text file is extracted using a parser. The keywords used to extract these details like “Datum attribute found”, “Text attribute found” & “Geomtol attribute found” are highlighted in the figures above. This information is stored in a custom created class structure. The structure is created in such a way that the feature is the top node which contains the face geometry, its face id, face type (planar or cylindrical) and machine tag (which states if it is a machined face or not). There are two subclasses for planar and cylindrical faces. For a planar face, the centroid and face normal are computed and stored. While for a cylindrical face, the root point, cylinder axis and radius information is stored. The tolerance values, are stored in two separate classes. Each tolerance value being stored points to its owner, based on the face id, in the feature class. The objects used to store these values are stored in a separate vector.

Once the PMI extraction is complete, the nominal data is used as input to the feature fitting algorithms. In the existing algorithm, the face normal is used as the initial direction which was modified to fit a plane using least square method and then obtain the initial direction instead. For a cylindrical face, the cylindrical axis and radius are inputs to the feature fitting algorithm. Planar fits give out a point on the face and its face normal while a cylindrical fit gives a point on the axis, a position vector of the axis and the radius of the cylinder. The output files from the feature fitting algorithms have the same name as the face label concatenated with “-FF.dat”. In this case, the feature fitting outputs are obtained for A1, A2, A3, B1, B2, C, MF1, CF2 & MF3 faces. Figure 32 (a & b) shows the feature fitting output for a planar and cylindrical face.



```
 Unconstrained One Sided Plane Fit

Direction Vector of the Plane is = ( 0.000401211 , -8.59816 , 0.000162183 )
Point on the Plane is = ( X = 5.55216 , Y = 4.07467 , Z = 3.92767 )
```

Figure 32 (a): Feature fitting output for a planar feature



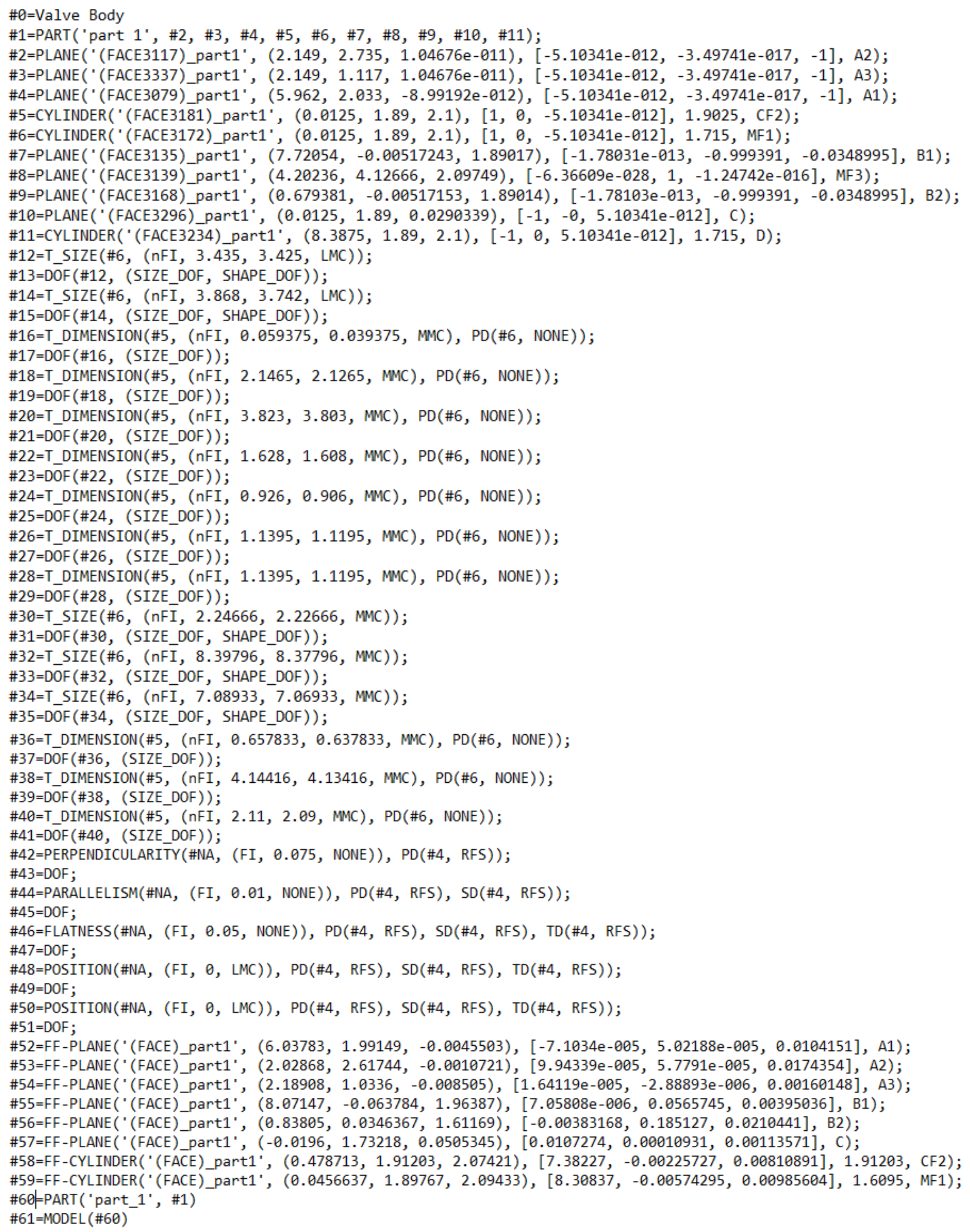
```
 Unconstrained Minimum Circumscirbed Cylinder Fit

Point on the Centre of the Cylinder is = ( X = 0.0456637 , Y = 1.89767 , Z = 2.09433 )
Direction vector of the Cylinder is = ( 8.30837 , -0.00574295 , 0.00985604 )
Radius of the Cylinder is = 1.6095
```

Figure 32 (b): Feature fitting output for a cylindrical feature

The initial feature fitting results for datum targets B1 and B2, on the demonstration article, had an error of the order 60/1000s. The features simulated, at these locations, using the point cloud data were not at the actual location, position or orientation. To better understand this error, a virtual point cloud was created at the nominal position (from the CAD model). The results obtained for the virtual point cloud did not include the 60/1000s error. A detailed analysis of the results led to the source of the error being the location of the datum targets (which were present on the parting line of the casting). The presence of draft (amount of taper for castings perpendicular to the parting line) prevents the surfaces from being exactly perpendicular to datum targets A1, A2, A3, and C; and the point cloud generated was positioned in such a way that, when a feature is fit to it, the feature remains perpendicular to datum targets A1, A2, A3, and C. This did not represent the actual feature since it was not perpendicular to any of the other datum targets. Hence it was identified that the scanning of such features must be done with care and must capture the draft at these locations, thereby eliminating errors like the 60/1000s (in this case).

The fixture type (or locating method) is obtained from the PMI text file as shown in Figure 31(b). In the test case, the method selected is of type 1 (3-2-1 method). The 6 datum target feature fitting results are selected and the user is asked to categorize them into the three, two and one. Once this is done, the three planes at A (A1, A2, and A3) are used to construct one single plane. Similarly, the data from feature fitting of B1 and B2 are used to construct a single plane called plane B. The third plane, plane C, is used as obtained from the feature fitting output. These three planes are used to set up a coordinate system and the intersection of the planes are used as the three axes. The vectors representing the coordinate axes and the origin are represented in the existing DRF. These vectors are then used to create a transformation matrix. The three vectors, representing x, y and z-axes, are the first three column vectors respectively, in the transformation matrix and the vector to the origin is the fourth column that represents the translation vector.

The entities involved, the CAD geometry and the feature fits, are transformed using the transformation matrix obtained. These results are then sent to the MCTF graph writer. The features are first printed and corresponding line numbers are assigned. Once the features are printed, the tolerance values are printed with the respective tolerance values and corresponding information with a pointer to the owner feature. After the tolerance data, the feature fitting data is printed, where each feature is prefixed with an “FF” to denote a feature fit face. The MCTF graph obtained for the test case is shown in Figure 33.



```
#0=Valve Body
#1=PART('part 1', #2, #3, #4, #5, #6, #7, #8, #9, #10, #11);
#2=PLANE('(FACE3117)_part1', (2.149, 2.735, 1.04676e-011), [-5.10341e-012, -3.49741e-017, -1], A2);
#3=PLANE('(FACE3337)_part1', (2.149, 1.117, 1.04676e-011), [-5.10341e-012, -3.49741e-017, -1], A3);
#4=PLANE('(FACE3079)_part1', (5.962, 2.033, -8.99192e-012), [-5.10341e-012, -3.49741e-017, -1], A1);
#5=CYLINDER('(FACE3181)_part1', (0.0125, 1.89, 2.1), [1, 0, -5.10341e-012], 1.9025, CF2);
#6=CYLINDER('(FACE3172)_part1', (0.0125, 1.89, 2.1), [1, 0, -5.10341e-012], 1.715, MF1);
#7=PLANE('(FACE3135)_part1', (7.72054, -0.00517243, 1.89017), [-1.78031e-013, -0.999391, -0.0348995], B1);
#8=PLANE('(FACE3139)_part1', (4.20236, 4.12666, 2.09749), [-6.36609e-028, 1, -1.24742e-016], MF3);
#9=PLANE('(FACE3168)_part1', (0.679381, -0.00517153, 1.89014), [-1.78103e-013, -0.999391, -0.0348995], B2);
#10=PLANE('(FACE3296)_part1', (0.0125, 1.89, 0.0290339), [-1, -0, 5.10341e-012], C);
#11=CYLINDER('(FACE3234)_part1', (8.3875, 1.89, 2.1), [-1, 0, 5.10341e-012], 1.715, D);
#12=T_SIZE(#6, (nFI, 3.435, 3.425, LMC));
#13=DOF(#12, (SIZE_DOF, SHAPE_DOF));
#14=T_SIZE(#6, (nFI, 3.868, 3.742, LMC));
#15=DOF(#14, (SIZE_DOF, SHAPE_DOF));
#16=T_DIMENSION(#5, (nFI, 0.059375, 0.039375, MMC), PD(#6, NONE));
#17=DOF(#16, (SIZE_DOF));
#18=T_DIMENSION(#5, (nFI, 2.1465, 2.1265, MMC), PD(#6, NONE));
#19=DOF(#18, (SIZE_DOF));
#20=T_DIMENSION(#5, (nFI, 3.823, 3.803, MMC), PD(#6, NONE));
#21=DOF(#20, (SIZE_DOF));
#22=T_DIMENSION(#5, (nFI, 1.628, 1.608, MMC), PD(#6, NONE));
#23=DOF(#22, (SIZE_DOF));
#24=T_DIMENSION(#5, (nFI, 0.926, 0.906, MMC), PD(#6, NONE));
#25=DOF(#24, (SIZE_DOF));
#26=T_DIMENSION(#5, (nFI, 1.1395, 1.1195, MMC), PD(#6, NONE));
#27=DOF(#26, (SIZE_DOF));
#28=T_DIMENSION(#5, (nFI, 1.1395, 1.1195, MMC), PD(#6, NONE));
#29=DOF(#28, (SIZE_DOF));
#30=T_SIZE(#6, (nFI, 2.24666, 2.22666, MMC));
#31=DOF(#30, (SIZE_DOF, SHAPE_DOF));
#32=T_SIZE(#6, (nFI, 8.39796, 8.37796, MMC));
#33=DOF(#32, (SIZE_DOF, SHAPE_DOF));
#34=T_SIZE(#6, (nFI, 7.08933, 7.06933, MMC));
#35=DOF(#34, (SIZE_DOF, SHAPE_DOF));
#36=T_DIMENSION(#5, (nFI, 0.657833, 0.637833, MMC), PD(#6, NONE));
#37=DOF(#36, (SIZE_DOF));
#38=T_DIMENSION(#5, (nFI, 4.14416, 4.13416, MMC), PD(#6, NONE));
#39=DOF(#38, (SIZE_DOF));
#40=T_DIMENSION(#5, (nFI, 2.11, 2.09, MMC), PD(#6, NONE));
#41=DOF(#40, (SIZE_DOF));
#42=PERPENDICULARITY(#NA, (FI, 0.075, NONE)), PD(#4, RFS));
#43=DOF;
#44=PARALLELISM(#NA, (FI, 0.01, NONE)), PD(#4, RFS), SD(#4, RFS));
#45=DOF;
#46=FLATNESS(#NA, (FI, 0.05, NONE)), PD(#4, RFS), SD(#4, RFS), TD(#4, RFS));
#47=DOF;
#48=POSITION(#NA, (FI, 0, LMC)), PD(#4, RFS), SD(#4, RFS), TD(#4, RFS));
#49=DOF;
#50=POSITION(#NA, (FI, 0, LMC)), PD(#4, RFS), SD(#4, RFS), TD(#4, RFS));
#51=DOF;
#52=FF-PLANE('(FACE)_part1', (6.03783, 1.99149, -0.0045503), [-7.1034e-005, 5.02188e-005, 0.0104151], A1);
#53=FF-PLANE('(FACE)_part1', (2.02868, 2.61744, -0.0010721), [9.94339e-005, 5.7791e-005, 0.0174354], A2);
#54=FF-PLANE('(FACE)_part1', (2.18908, 1.0336, -0.008505), [1.64119e-005, -2.88893e-006, 0.00160148], A3);
#55=FF-PLANE('(FACE)_part1', (8.07147, -0.063784, 1.96387), [7.05808e-006, 0.0565745, 0.00395036], B1);
#56=FF-PLANE('(FACE)_part1', (0.83805, 0.0346367, 1.61169), [-0.00383168, 0.185127, 0.0210441], B2);
#57=FF-PLANE('(FACE)_part1', (-0.0196, 1.73218, 0.0505345), [0.0107274, 0.00010931, 0.00113571], C);
#58=FF-CYLINDER('(FACE)_part1', (0.478713, 1.91203, 2.07421), [7.38227, -0.00225727, 0.00810891], 1.91203, CF2);
#59=FF-CYLINDER('(FACE)_part1', (0.0456637, 1.89767, 2.09433), [8.30837, -0.00574295, 0.00985604], 1.6095, MF1);
#60=PART('part_1', #1)
#61=MODEL(#60)
```

Figure 33: MCTF Graph obtained for test case

The MCTF Graph is then communicated to further compute the adjustment required for the fixture setups for the part prior to the machining operations.

5.2 Test Case II

A second test case was developed at ASU to test and validate the algorithms. It consisted the CAD geometry with GD&T information for it, in CREO (*.prt) format and the point cloud data associated with each feature in ASCII file format (*.asc). The input CAD geometry with GD&T information for it is shown in Figure 34 (a & b).

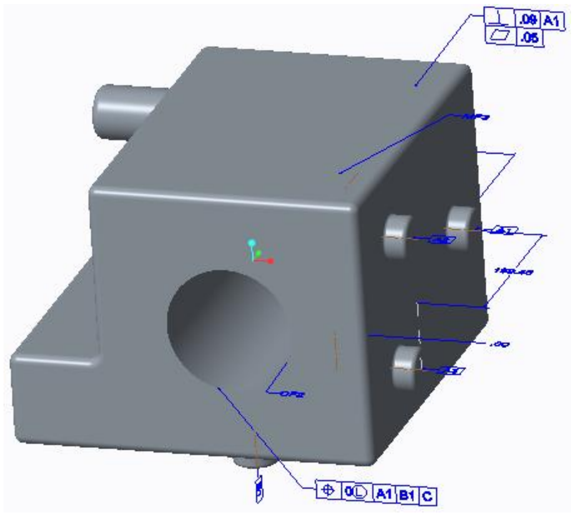


Figure 34 (a): CAD geometry with GD&T information

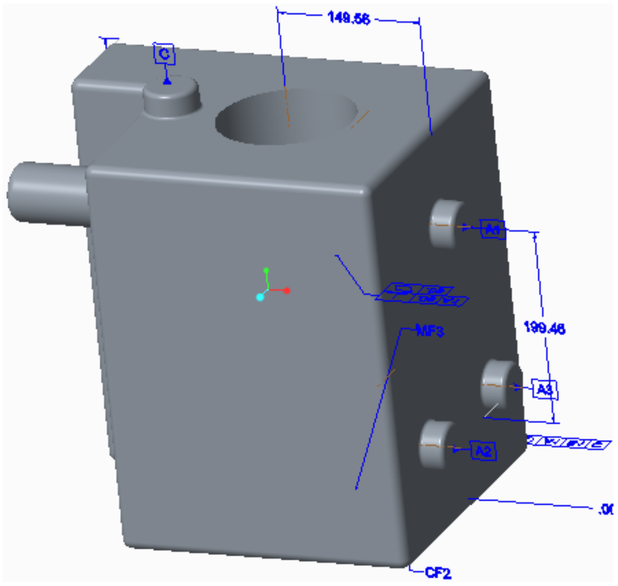
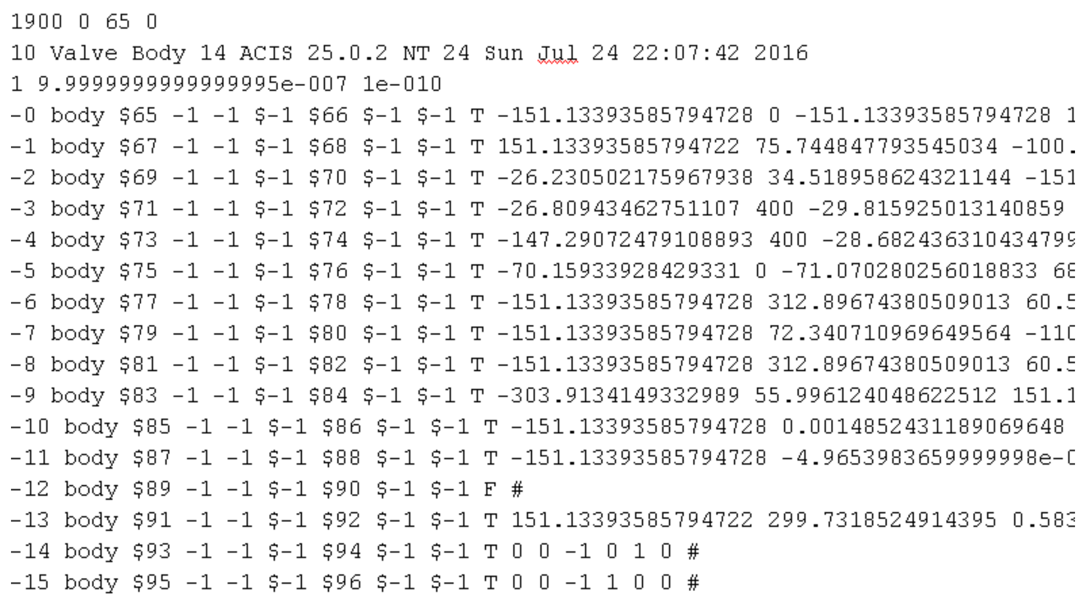


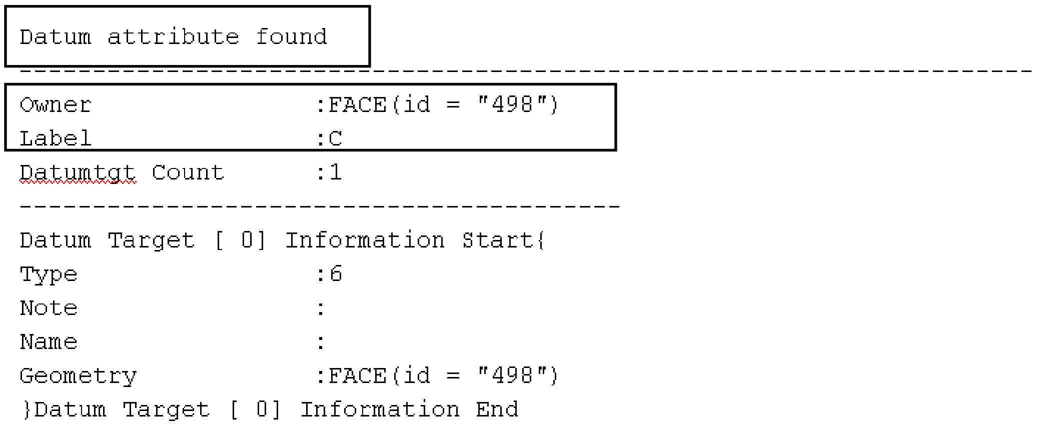
Figure 34 (b): CAD geometry with GD&T information

In this part, the critical features are the six datum targets (A1, A2, A3, B1, B2, & C), one machined feature (MF3) and one critical feature (CF2). The point cloud data for these features are saved with the same filename as the datum/face labels assigned to it in the CAD model. The folder structure must remain similar as shown in the previous case. Figure 35 shows the *.sat for the part and Figure 36 (a & b) are samples from the PMI text file.



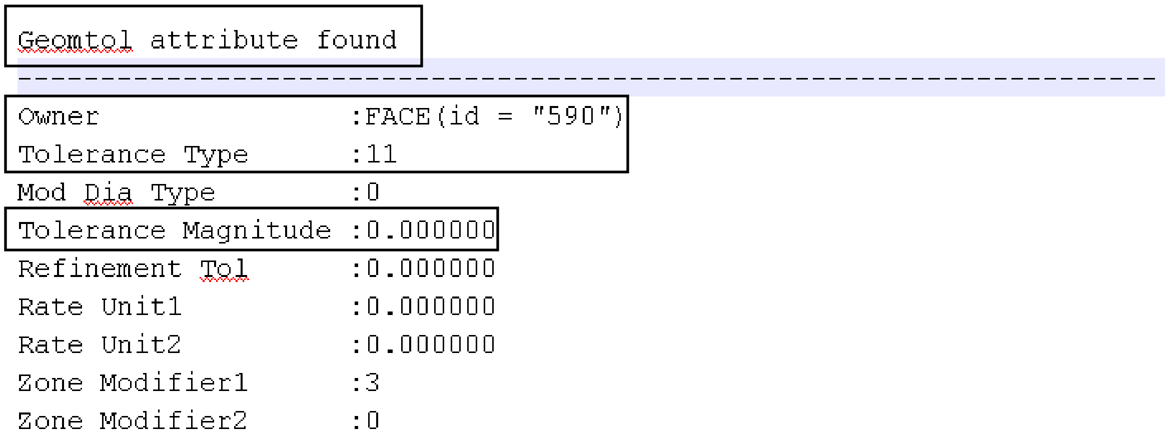
```
1900 0 65 0
10 Valve Body 14 ACIS 25.0.2 NT 24 Sun Jul 24 22:07:42 2016
1 9.9999999999999995e-007 1e-010
-0 body $65 -1 -1 $-1 $66 $-1 $-1 T -151.13393585794728 0 -151.13393585794728 1
-1 body $67 -1 -1 $-1 $68 $-1 $-1 T 151.13393585794722 75.744847793545034 -100.
-2 body $69 -1 -1 $-1 $70 $-1 $-1 T -26.230502175967938 34.518958624321144 -151
-3 body $71 -1 -1 $-1 $72 $-1 $-1 T -26.80943462751107 400 -29.815925013140859
-4 body $73 -1 -1 $-1 $74 $-1 $-1 T -147.29072479108893 400 -28.682436310434799
-5 body $75 -1 -1 $-1 $76 $-1 $-1 T -70.15933928429331 0 -71.070280256018833 68
-6 body $77 -1 -1 $-1 $78 $-1 $-1 T -151.13393585794728 312.89674380509013 60.5
-7 body $79 -1 -1 $-1 $80 $-1 $-1 T -151.13393585794728 72.340710969649564 -110
-8 body $81 -1 -1 $-1 $82 $-1 $-1 T -151.13393585794728 312.89674380509013 60.5
-9 body $83 -1 -1 $-1 $84 $-1 $-1 T -303.9134149332989 55.996124048622512 151.1
-10 body $85 -1 -1 $-1 $86 $-1 $-1 T -151.13393585794728 0.0014852431189069648
-11 body $87 -1 -1 $-1 $88 $-1 $-1 T -151.13393585794728 -4.9653983659999998e-0
-12 body $89 -1 -1 $-1 $90 $-1 $-1 F #
-13 body $91 -1 -1 $-1 $92 $-1 $-1 T 151.13393585794722 299.7318524914395 0.583
-14 body $93 -1 -1 $-1 $94 $-1 $-1 T 0 0 -1 0 1 0 #
-15 body $95 -1 -1 $-1 $96 $-1 $-1 T 0 0 -1 1 0 0 #
```

Figure 35: Partial ACIS (*.sat) file for test part



```
Datum attribute found
---------------------------------------------------------------------
Owner                   :FACE(id = "498")
Label                   :C
Datumtgt Count          :1
-----------------------------------------
Datum Target [ 0] Information Start{
Type                    :6
Note                    :
Name                    :
Geometry                :FACE(id = "498")
}Datum Target [ 0] Information End
```

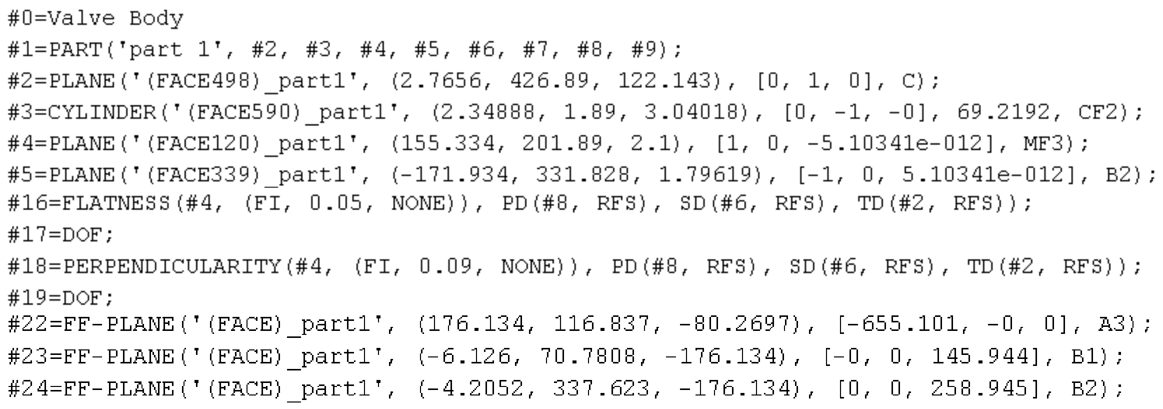
Figure 36 (a): Extract from PMI text file of Datum attribute



```
Geomtol attribute found
----------------------------------------------------------------------
Owner                    :FACE(id = "590")
Tolerance Type           :11
Mod Dia Type             :0
Tolerance Magnitude      :0.000000
Refinement Tol           :0.000000
Rate Unit1               :0.000000
Rate Unit2               :0.000000
Zone Modifier1           :3
Zone Modifier2           :0
```

Figure 36 (a): Extract from PMI text file of tolerance attribute

The PMI is read and parsed from the text file into a custom class structure. The data stored in the class structure is used to fit various features using the feature fitting algorithms. In this case, the feature fitting outputs are obtained for A1, A2, A3, B1, B2, C, CF2 & MF3 faces. The fixture type used in the test part is type 1. Hence the GDRF is constructed based on the algorithm for fixture type 1. The entities are then transformed into the GDRF and the MCTF is written accordingly. Figure 37 shows a sample of the MCTF written for this part.



```
#0=Valve Body
#1=PART('part 1', #2, #3, #4, #5, #6, #7, #8, #9);
#2=PLANE('(FACE498)_part1', (2.7656, 426.89, 122.143), [0, 1, 0], C);
#3=CYLINDER('(FACE590)_part1', (2.34888, 1.89, 3.04018), [0, -1, -0], 69.2192, CF2);
#4=PLANE('(FACE120)_part1', (155.334, 201.89, 2.1), [1, 0, -5.10341e-012], MF3);
#5=PLANE('(FACE339)_part1', (-171.934, 331.828, 1.79619), [-1, 0, 5.10341e-012], B2);
#16=FLATNESS(#4, (FI, 0.05, NONE)), PD(#8, RFS), SD(#6, RFS), TD(#2, RFS));
#17=DOF;
#18=PERPENDICULARITY(#4, (FI, 0.09, NONE)), PD(#8, RFS), SD(#6, RFS), TD(#2, RFS));
#19=DOF;
#22=FF-PLANE('(FACE)_part1', (176.134, 116.837, -80.2697), [-655.101, -0, 0], A3);
#23=FF-PLANE('(FACE)_part1', (-6.126, 70.7808, -176.134), [-0, 0, 145.944], B1);
#24=FF-PLANE('(FACE)_part1', (-4.2052, 337.623, -176.134), [0, 0, 258.945], B2);
```

Figure 37: Partial MCTF Graph obtained for test case

CHAPTER 6

CONCLUSION

The research work described in this thesis creates a seamless digital thread for data communication between various systems. The hope is that such a software package will help digitally connect the designing and manufacturing processes, thereby help produce better-machined castings. Having weighed all the available options of data transfer, the most efficient method to transfer data has been incorporated. A file parser has been developed in order to scan and compartmentalize the PMI information specified in the CAD model, into a text file using ACIS. Further, to communicate the data pertaining to the machining process, and the data required to devise methods to automate the fixture setups, a separate data structure (MCTF) that includes the GD&T information, geometry of the part and data pertaining to the features to be machined, has been developed. The data stored in MCTF is further used to compute the fixture adjustments.

Multiple file formats we tested during the course of the research work. The status of various file formats that could or could not, communicate the PMI along with the geometry, are listed in Table 11.

Table 11: File Formats to Transfer GD&T

Annotated CAD Model Format	**Comments**	**Testing Status**
Siemens NX	Directly translated to ACIS (*.sat)	Success
SolidWorks 2014	Directly translated to ACIS (*.sat)	Fail[#]
SolidWorks	Translated to AP242 (intermediate *.sat)	Success[*]
PTC Creo	Directly translated to ACIS (*.sat)	Success

* Requires parsing a separate file (apart from SolidWorks file) to get the GD&T information

# Spatial claims the possibility of extracting GD&T information from a SolidWorks 2015 part file using InterOp R25 – 64bit (Not tested)

6.1 Limitations

In this research, all the proposed methods are general and expandable. This research is divided into three steps, (i) CAD data translation (ii) automating feature fitting and the creation of a GDRF and (iii) transforming data into new GDRF and creating an MCTF. The algorithm in place for translating the CAD geometry and PMI using InterOp® is limited to part files created in CREO®. The algorithm must be modified to enable it to read other CAD data formats. The nFF library is currently limited to planar and cylindrical features only. Any other type of feature cannot be handled using the current tool. Finally, the fixture library used is limited and does not cover all types of fixture methods used in various machining processes. The current tool can handle only three types of fixture methods. The fixture library can be studied and expanded to add more fixture types, with the required vector algebraic calculations to construct the GDRF accordingly.

6.2 Improvements & Future Work

There are tools that can be used to create a STEP AP242 file from the CAD geometry and GD&T information provided as input. Once STEP AP242 becomes available in commercial CAD systems, this tool can be modified to read STEP AP242 file directly using InterOp®. There are many other CAD file formats that can be tested and used, apart from the file formats listed in Table 11, to communicate the CAD geometry and PMI.

The fixture library specified here is very limited and can be studied and expanded to include many other fixture methods used in machining. An expanded fixture library will help generalize the process of obtaining fixture adjustments for parts involving complex designs and various machining processes. Since the GDRF is constructed based on the fixture method used as part of the machining process, the expansion of the fixture library is vital for making the whole process more robust. Once additional fixture methods are added to the library, the algorithms and codes to construct a GDRF based on each of the fixture methods, are required to be added to the tool.

Lastly, the MCTF is an open data structure that can be modified to add or remove data from it. The details pertaining to fitted features can also be modified and written in a different format if required.

REFERENCES

[1] "Straight to Sand" [Online]. Available: http://www.mmsonline.com/videos/straight-to-sand. [Accessed: 20-Jul-2016].

[2] ASME Y14.5, 2009, "Dimensioning and Tolerancing," NY Am. Soceity Mech. Eng., p. 704.

[3] Mohan, P., 2013, "Development and Verification of a Library of Feature Fitting Algorithms for CMMs," Arizona State University.

[4] Wong, A., "Review • Orthographic Projection • Dimensioning," pp. 5–70.

[5] Bi, Z. M., and Zhang, W. J., 2001, "Flexible fixture design and automation: Review, issues and future directions," Int. J. Prod. Res., **39**(13), pp. 2867–2894.

[6] 2012, "Royal Machine & Tool Introduces Universal Valve Fixture" [Online]. Available: http://www.mfgnewsweb.com/archives/4/35985/General-jan12/Royal-Machine-and-Tool-Introduces-Universal-Valve-Fixture.aspx. [Accessed: 20-Jul-2016].

[7] Rong, Y. (Kevin), and Zhu, Y. (Stephens), 1999, Computer-Aided Fixture Design, New York : Marcel Dekker, c1999.

[8] Haghighi, P., 2015, "3-D Conformance Analysis of Manufacturing Plans Using M-Maps , by Explicating Formal GD & T Schema from the Process Plan," Arizona State University.

[9] Shen, Z., 2005, "DEVELOPMENT OF A FRAMEWORK FOR A SET OF COMPUTER-AIDED TOOLS FOR TOLERANCE ANALYSIS," Arizona State University.

[10] Xu, X., 2011, Integrating advanced computer-aided design, manufacturing, and numerical control: principles and implementations, by X. Xu.

[11] Venkiteswaran, A., Hejazi, S. M., Biswas, D., Shah, J. J., and Davidson, J. K., 2016, "Semantic Interoperability of GD&T Data Through ISO 10303 STEP AP242," IDETC/CIE 2016, pp. 1–11.

[12] "CAD data exchange" [Online]. Available: https://en.wikipedia.org/wiki/CAD_data_exchange. [Accessed: 22-Jul-2016].

[13] Europe, A. and D. I. A. of, "PMI interoperability based on ISO STEP standards" [Online]. Available: http://www.asd-ssg.org/pmi-interoperability. [Accessed: 17-Jul-2016].

[14] Overmars, M., 1996, Fundamentals of Tool Design Study Guide.

[15] McKechnie, G., 2005, "Adjustable angle plate" [Online]. Available: https://en.wikipedia.org/wiki/Angle_plate. [Accessed: 20-Jul-2016].

[16] "VISE-154S-20-1 Vise Action Fixture" [Online]. Available: http://www.starrett-

precision.co.uk/en/metrology/product-detail/7-Testing-Equipment/72-Grips-Fixtures/7208-Vise-Action-Fixtures/VISE-154S-20-1. [Accessed: 20-Jul-2016].

[17] Rong, Y., Zhu, J., and Li, S., 1993, “Fixturing Feature Analysis for Computer-aided Fixture Design,” Intell. Des. Manuf. ASME, **64**, pp. 267–271.

[18] Creative Tools, “CreativeTools.se - VIUscan - Laser-scanned - ZPrinter - 3D printed - Viking Belt Buckle 24” [Online]. Available: https://www.flickr.com/photos/creative_tools/4530599701/. [Accessed: 07-Jul-2016].

[19] Autodesk, 2016, “Point Cloud Object” [Online]. Available: http://help.autodesk.com/view/3DSMAX/2016/ENU/?guid=GUID-49CE0ACB-1345-4D50-B6E5-361DBFDB5B33. [Accessed: 07-Jul-2016].

[20] Shen, Z., Shah, J. J., and Davidson, J. K., 2008, “Analysis neutral data structure for GD&T,” J. Intell. Manuf., **19**(4), pp. 455–472.

[21] Wu, Y., Shah, J. J., and Davidson, J. K., 2003, “Computer Modeling of Geometric Variations in Mechanical Parts and Assemblies,” J. Comput. Inf. Sci. Eng., **3**(1), p. 54.

[22] Dassault Systèmes, S. C., 2016, “3D ACIS Modeler” [Online]. Available: https://doc.spatial.com/get_doc_page/articles/3/d/_/3D_ACIS_Modeler_9aee.html. [Accessed: 16-Jun-2016].