

Semantic Model Difference Identification for Aerospace Sheet Metal Part CAD Models

by

Syedmorteza GHAFARISHAHRI

MANUSCRIPT-BASED THESIS PRESENTED TO ÉCOLE DE
TECHNOLOGIE SUPÉRIEURE IN PARTIAL FULFILLMENT FOR THE
DEGREE OF DOCTOR OF ENGINEERING
Ph.D.

MONTREAL, FEBRUARY 23, 2023

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC



Syedmorteza Ghaffarishahri, 2022



This Creative Commons licence allows readers to download this work and share it with others as long as the author is credited. The content of this work can't be modified in any way or used commercially.

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED
BY THE FOLLOWING BOARD OF EXAMINERS

Mr. Louis Rivest, Thesis Supervisor
Department of Systems Engineering, École de technologie supérieure

Mr. Souheil-Antoine Tahan, President of the Board of Examiners
Department of Mechanical Engineering, École de technologie supérieure

Mr. Éric Wagnac, Member of the Jury
Department of Mechanical Engineering, École de technologie supérieure

Mr. Jean-Christophe Cuillère, External Evaluator
Department of Mechanical Engineering, Université du Québec à Trois-Rivières

Mr. Fernando Mas, Independent External Evaluator
Department of Mechanical Engineering and Manufacturing, Universidad de Sevilla

THIS THESIS WAS PRESENTED AND DEFENDED
IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC
ON JANUARY 26, 2023
AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGMENTS

Firstly, I would like to express my sincere gratitude to Professor Louis Rivest, my doctorate supervisor, for his invaluable guidance and inputs both scholarly and beyond, that I received throughout this work. I am grateful to find him and be accepted as one of his doctoral students, which led to me completing this doctoral thesis.

Special thanks go to my family, both in Iran and in Canada, for their encouragement and love, which were essential to completing my Ph.D. program. To Kendall Behrsin, my partner, I cannot thank you enough for being there for me, your support and being you! To my parents, I am ever grateful for your support in this journey and in life. I thank you all from the bottom of my heart.

Identification sémantique des différences entre modèles CAO de pièces de métal en feuille aéronautiques

Seyedmorteza GHAFFARISHAHRI

RÉSUMÉ

De nombreux processus inhérents à la gestion du cycle de vie des produits, comme la réutilisation des informations, peuvent bénéficier d'une identification sémantique des différences entre modèles (*semantic Model Difference Identification* ou *semantic MDI*). Une identification sémantique des différences entre modèles peut être définie comme une solution qui identifie et représente les différences, entre deux modèles faisant l'objet de la comparaison, d'une manière significative au plan de l'ingénierie – par exemple : tel trou a été déplacé. L'identification sémantique des différences est difficile en raison de la variété des logiciels de modélisation, de la non-unicité des séquences de modélisation et de l'utilisation d'informations de bas niveau dans les échanges de données.

Cette thèse propose une méthode d'identification sémantique des différences entre modèles qui identifie et représente les différences entre modèles CAO (Conception assistée par ordinateur) 3D sur la base de la sémantique propre à l'ingénierie par le biais de caractéristiques (*Features*). Le domaine d'application exploré est celui des pièces de structures aéronautiques faites de métal en feuille et typiquement obtenues par pliage (*Brake-Forming*) ou hydroformage (*Hydro-Forming*).

Les échanges de modèles 3D étant souvent basés sur des informations de bas niveau, c'est-à-dire des représentations par les frontières (*Boundary Representation, B-rep*), une méthode de reconnaissance automatique de caractéristiques (*Automated Feature Recognition, AFR*) est établie en premier lieu. En effet, la reconnaissance automatique de caractéristiques permet de s'affranchir des représentations de bas niveau pour élever la sémantique des communications technique. Bien que les pièces de métal en feuille constituent une part importante des avions en service, il n'existe pas de méthode AFR spécialisée qui leur soit dédiée, et malgré la présence de plusieurs méthodes AFR pour les pièces de métal en feuille, aucune d'entre elles n'est adaptée pour reconnaître les caractéristiques particulières à l'industrie aéronautique.

Cette thèse propose donc la première méthode AFR pour reconnaître les caractéristiques de pièces de métal en feuille rencontrées au sein de structures aéronautiques afin d'élever le niveau d'abstraction de l'information manipulée à partir de modèles STEP 3D. La méthode fait d'abord un prétraitement du modèle STEP 3D afin de classer les éléments topologiques des modèles B-rep et de créer de nouveaux ensembles de faces, de sous-types de faces, de limites de faces et d'arêtes. Une démarche à base de règles est ensuite utilisée pour reconnaître les caractéristiques typiques du métal en feuille aéronautique. Les caractéristiques extraites sont décrites par leur géométrie, leurs relations avec d'autres caractéristiques et leurs paramètres pertinents.

Une fois les caractéristiques reconnues, l'identification des différences entre les modèles est effectuée. La méthode MDI proposée dans cette thèse consiste principalement en une étape de mise en correspondance sémantique (*Semantic Pose Registration*) des modèles suivie d'une étape d'identification de leurs différences. La méthode de mise en correspondance sémantique exploite le fait que chacune des caractéristiques d'une pièce satisfait des fonctions spécifiques,

dont certaines sont prioritaires dans l'intention de conception d'une pièce de type métal en feuille aéronautique. La mise en correspondance est réalisée en s'appuyant sur les caractéristiques préalablement identifiées au sein des modèles CAO 3D et selon des priorités établies entre elles dans ce domaine d'application spécifique. L'identification des différences se fait ensuite en identifiant d'abord les caractéristiques communes aux deux modèles CAO 3D comparés, puis en identifiant les différences. Les différences entre les caractéristiques des deux modèles sont classées comme des caractéristiques ajoutées, supprimées ou différentes. Globalement, la méthode proposée pour faire la mise en correspondance sémantique des modèles CAO 3D et pour identifier de manière sémantique les différences entre ces modèles exploite les caractéristiques identifiées au sein des pièces et, par extension, les intentions de conception sous-jacentes.

Les méthodes proposées pour la reconnaissance des caractéristiques et pour l'identification sémantique des différences entre modèles sont mises en œuvre pour être validées. Les prototypes implémentent des versions légèrement modifiées des algorithmes et les modifications sont mises en évidence. Pour valider la méthode AFR et en vérifier la mise en œuvre, une collection de 26 pièces de métal en feuille tirées de structures aéronautiques réelles a été utilisée pour créer des modèles CAO qui ont ensuite été convertis au format STEP. Les résultats confirment que la méthode AFR proposée fonctionne de manière très satisfaisante pour ce domaine d'application. Pour valider la méthode MDI et en vérifier la mise en œuvre, 3 des pièces de métal en feuille utilisées pour tester le prototype AFR ont été modifiées afin de refléter les différences possibles entre des pièces similaires dans des scénarios du monde réel. Les résultats obtenus sont parfaitement conformes à ceux attendus et confirment le potentiel des algorithmes d'identification sémantique des différences entre modèles de pièces basés sur des caractéristiques dans des domaines d'application spécialisés.

Mots-clés : Pièces de métal en feuille aéronautique, Reconnaissance des caractéristiques, Comparaison sémantique de modèles CAO, Identification des différences

Semantic Model Difference Identification for Aerospace Sheet Metal Part CAD Models

Seyedmorteza GHAFARISHAHRI

ABSTRACT

Many product lifecycle management's processes like information reuse can take a lot of advantage from a semantic model difference identification (MDI). A semantic MDI can be defined as a solution that identifies and represents the differences between two compared models in terms of meaningful engineering information – for instance: this hole was moved. Semantic difference identification is especially challenging due to the variety of modeling solutions, the non-uniqueness of modeling sequences and the use of low-level information in engineering communications.

This work proposes an MDI method that identifies and represents the differences between 3D Computer-Aided Design (CAD) models based on engineering semantics through features. Brake- and hydro-formed aerospace structural sheet metal parts are used as the application domain in which to propose and illustrate the method.

Because engineering communications of 3D models are commonly in low-level information i.e., B-rep models, an automated feature recognition (AFR) method is first needed. An AFR solution elevates semanticity of engineering communication information without being reliant on the suboptimal modeling solutions. Although structural sheet metal parts form a significant portion of airplanes, there is no specialized AFR method dedicated to them. Despite the presence of several AFR methods for sheet metal parts, none of them are tuned to recognize the design semantics of the aerospace industry.

This work proposes the first AFR method to recognize aerospace sheet metal features and design semantics to elevate the level of abstraction of the information from 3D STEP models. It starts with preprocessing the 3D STEP model in order to classify the topological elements of the B-rep models and create relevant novel face sets and subtypes of faces, face boundaries and edges. Then, rule-based steps are used to recognize aerospace sheet metal features. The extracted features are described by their geometry, their relationship with other features and their pertinent parameters.

Once the features are recognized, model difference identification is performed. This work's MDI method consists mainly of a pose registration stage and a difference identification stage. The pose registration method exploits the fact that all the features of a part serve specific functions, some of which are fundamental to the part's essential functionality, and that they are intertwined with the design intent of the part, which is particularly true for aerospace sheet metal parts. This provides the opportunity to semantically register feature-based 3D CAD models according to the unique purpose of the features in this specific domain of application. Difference identification is approached by primarily identifying and segregating the commonality between the compared 3D CAD models and then identifying the differences. The differences between 3D CAD models are classified as added, removed or differed features. The proposed MDI method describes a way to fully pose-register 3D CAD models and identify their differences semantically based solely on their features, and, by extension, their design intent.

Both the AFR and MDI methods are implemented to be validated. The prototypes are modified implementation of the original methods and the differences are outlined and pointed out. To validate the AFR method and verify its correct implementation, a collection of 26 real-world aerospace structural sheet metal parts was used to create CAD models that were subsequently converted to STEP models. The results show perfect accuracy and confirm that AFR works for this domain of application. To validate the MDI method and verify its correct implementation, 3 of the real-world ASM parts used in testing AFR prototype were modified to reflect the possible differences that could occur between similar parts in real-world scenarios. The results show perfect accuracy and confirm there is great potential for further development of MDI algorithms for feature-based models of parts from specialized domains of application.

Keywords: Aerospace sheet metal parts, Feature recognition, Semantic CAD model comparison, Model difference identification

TABLE OF CONTENTS

	Page
INTRODUCTION	1
CHAPTER 1 PROBLEM STATEMENT AND OBJECTIVES	6
1.1 Problem statement.....	6
1.2 Research objectives.....	9
1.3 Methodology	9
CHAPTER 2 RELATED LITERATURE	11
2.1 Automated feature recognition (AFR) for CAD models	11
2.1.1 Syntactic pattern recognition	12
2.1.2 Logic (if-then) rules and expert systems	13
2.1.3 Graph-based approach	14
2.1.4 Convex-hull and cell based volumetric decomposition approach	15
2.1.5 Hint-based approaches	17
2.2 AFR for sheet metal parts	17
2.3 Semantic CAD model comparison.....	19
2.3.1 CAD model comparison subjects.....	21
2.3.2 CAD model comparison interfaces	22
2.3.3 CAD model comparison mediums.....	24
2.3.4 CAD model comparison problems.....	30
2.3.5 CAD model comparison solving methods	33
2.3.6 CAD model comparison purposes	35
2.4 Synthesis	36
CHAPTER 3 FEATURE RECOGNITION FOR STRUCTURAL AEROSPACE SHEET METAL PARTS	37
3.1 Abstract	37
3.2 Introduction.....	38
3.3 Previous works.....	39
3.4 Term definitions and premises.....	40
3.4.1 Structure of B-rep models.....	41
3.4.2 Feature definition and description	41
3.4.3 ASM feature taxonomy.....	42
3.5 Proposed automated feature recognition method.....	49
3.5.1 Classifying and grouping of B-rep elements	50
3.5.2 Recognizing features.....	60
3.6 An example	74
3.7 Conclusion	77

CHAPTER 4	FEATURE-BASED MODEL DIFFERENCE IDENTIFICATION FOR AEROSPACE SHEET METAL PARTS	79
4.1	Abstract	79
4.2	Introduction	80
4.3	Related literature and positioning of proposal	82
4.3.1	Pose registration	82
4.3.2	Feature-based comparison	83
4.3.3	Feature definition and description	84
4.3.4	Aerospace sheet metal sample parts	87
4.4	Proposed method	92
4.4.1	Pose registration	95
4.4.2	Difference identification	99
4.4.3	Illustration of the proposed method	106
4.5	Discussion	111
4.6	Conclusion	113
CHAPTER 5	A PROTOTYPE OF AN AUTOMATED FEATURE RECOGNITION ALGORITHM FOR AEROSPACE SHEET METAL PARTS	115
5.1	Abstract	115
5.2	Introduction	116
5.3	Definitions and methodology	120
5.3.1	Definitions	120
5.3.2	3D models of sample parts	123
5.3.3	Modified steps of the AFR method	127
5.3.4	Data structures	129
5.3.5	The algorithms	135
5.4	Results	139
5.5	Discussion and conclusion	144
CHAPTER 6	RESULTS FROM THE MDI PROTOTYPE	147
6.1	Testing procedure	147
6.2	Results	152
6.3	Summary	157
CHAPTER 7	DISCUSSION OF THE RESULTS	159
7.1	Aerospace sheet metal feature taxonomy	159
7.2	Contributions	160
7.3	Assumptions and limitations	162
7.4	Future works	163
CONCLUSION	165
APPENDIX I	CLASSIFYING SHEET_FACES ADJACENT TO WEB_FACE	169

APPENDIX II	CLASSIFYING REMAINING SHEET_FACES	170
APPENDIX III	CHANGING BEND_FACES TO CONNECT_FACES.....	172
APPENDIX IV	CHANGING DETAINED_FACES TO CONNECT_FACES	173
APPENDIX V	CHANGING DETAINED_FACES TO BEND_FACES OR WALL_FACES.....	174
APPENDIX VI	CHANGING WALL_FACES TO BEND_FACES	175
APPENDIX VII	RECOGNIZING TWIN JOGGLES.....	176
APPENDIX VIII	RECOGNIZING THE REMAINING FLANGES.....	178
APPENDIX IX	RECOGNIZING COMBINED FLANGES	180
APPENDIX X	RECOGNIZING STRINGER CUTOUTS AND BEND RELIEFS ...	182
APPENDIX XI	RECOGNIZING CORNERS	185
APPENDIX XII	RECOGNIZING HOLES, CUTOUTS, LIGHTENING HOLES, LIGHTENING CUTOUTS AND BEADS	186
APPENDIX XIII	RECOGNIZING LIPS	188
APPENDIX XIV	POSE REGISTRATION DETAILS	189
APPENDIX XV	MODEL DIFFERENCE IDENTIFICATION DETAILS	192
	LIST OF REFERENCES.....	203

LIST OF TABLES

	Page
Table 2.1	The rule-based AFR methods and their relevant references12
Table 2.2	The comparison interfaces and their references*23
Table 2.3	The comparison interfaces and comparison mediums*25
Table 2.4	The comparison mediums and comparison problems*30
Table 2.5	The comparison problems and the comparison solving methods proposed for them34
Table 5.1	AFR implementation means used in the literature.....119

LIST OF FIGURES

	Page
Figure 0.1 The basic phases of a product throughout its lifecycle	1
Figure 2.1 Presentation of a native 3D CAD model and its corresponding graph	26
Figure 2.2 Conversion of a 3D CAD model to 2D information.....	28
Figure 2.3 Conversion of 3D CAD models to complete and partial shape descriptor histograms	29
Figure 3.1 The proposed taxonomy of features in the ASM parts	43
Figure 3.2 Illustration of the web as the feature with the highest surface area	44
Figure 3.3 Illustration of trim features	45
Figure 3.4 Illustration of deformation features (a, b, c, d, e, f)	47
Figure 3.5 Illustration of deformation features (g, h).....	48
Figure 3.6 Illustration of features	49
Figure 3.7 Flowchart of the proposed method	50
Figure 3.8 Illustration of the process of classifying faces (a, b, c, d, e, f).....	55
Figure 3.9 Illustration of the process of classifying faces (g, h, i, j)	56
Figure 3.10 Illustration of the faces included in creating joggle_face_sets	57
Figure 3.11 A fabricated example to illustrate subtypes of face_bounds	59
Figure 3.12 A fabricated example to illustrate subtypes of edges	60

Figure 3.13	A fabricated example of how the presence of a shared edge between the web_face and the connect_faces of a joggle_face_set distinguishes a joggle on a flange from a joggle on the web (a) and (b); and the parameters of joggles (c).....	63
Figure 3.14	An example of two joggles merged into a twin joggle and a deformed web merged into the web	65
Figure 3.15	Illustration of the geometry of temp-flange	66
Figure 3.16	Two examples of determining the order in which the parent feature of the joggle, the joggle and the child feature of the joggle occur, to distinguish flange and deformed flange	67
Figure 3.17	Illustration of recognizing and characterizing (a) a combined flange and (b) the parameters of a flange	69
Figure 3.18	Analyzing edges to recognize stringer cutouts (a) and (b), bend reliefs (c) and corners (d)	71
Figure 3.19	Parameters of a hole, a lightening hole, a cutout, a lightening cutout and a bead	74
Figure 3.20	Illustration of feature recognition process for a real part model (a, b, c, d)	75
Figure 3.21	Illustration of feature recognition process for a real part model (e, f, g, h, i, j)	76
Figure 3.22	Illustration of feature recognition process for a real part model (k, l)	77
Figure 4.1	A representation of feature information and the illustration of its attributes	86
Figure 4.2	Examples of features in real-world ASM parts.....	88
Figure 4.3	Variations* of attachment flanges and stiffening flanges.....	89
Figure 4.4	The taxonomy of feature types in the studied ASM parts in terms of their functions and purposes.....	91

Figure 4.5	Representation of the web, an attachment flange, a lightening hole and a corner of an actual ASM part according to the feature definition92
Figure 4.6	Overview of the proposed pose registration and difference identification stages.....94
Figure 4.7	Illustration of the proposed semantic registration method through an example (a, b, c, d).....98
Figure 4.8	Illustration of the proposed semantic registration method through an example (e, f, g)99
Figure 4.9	The reference (a) and target (b) models for the proposed difference identification method103
Figure 4.10	Illustrations of key steps of the proposed difference identification method.....104
Figure 4.11	Illustrations of key steps of the proposed difference identification method.....105
Figure 4.12	Illustrations of a key step of the proposed difference identification method.....106
Figure 4.13	Semantic registration of two versions of a 3D CAD model of an ASM part107
Figure 4.14	Difference identification of two versions of a 3D CAD model of an ASM part (a, b, c, d)109
Figure 4.15	Difference identification of two versions of a 3D CAD model of an ASM part (e, f, g, h, i, j).....110
Figure 4.16	Difference identification of two versions of a 3D CAD model of an ASM part (k, l, m, n).....111
Figure 5.1	CAD models of the sample aerospace sheet metal parts (a, b, c, d, e, f, g)124

Figure 5.2	CAD models of the sample aerospace sheet metal parts (h, i, j, k, l, m, n, o, p)125
Figure 5.3	CAD models of the sample aerospace sheet metal parts (q, r, s, t, u, v, w, x, y, z).....126
Figure 5.4	Flowchart of the modified method implemented in the prototype129
Figure 5.5	Class diagram representing the B-rep elements of the STEP files132
Figure 5.6	Class diagram representing the further abstracted and enhanced data structure.....133
Figure 5.7	Class diagram representing the features and their data structure.....134
Figure 5.8	Flowchart representing the algorithm to identify sheet_faces and trim_faces.....137
Figure 5.9	Flowchart representing the algorithm to recognize some joggles and flanges138
Figure 5.10	Flowchart representing the algorithm to create joggles and temporary flanges and recognize the deformed flange from the temporary flanges....139
Figure 5.11	The content of the feature file for the part represented in Figure 5.1 (a)....140
Figure 5.12	A rather typical design of a hydroformed aerospace sheet metal part annotated with its features (a) as well as the content of its feature file (b).....141
Figure 5.13	A rather common design of a hydroformed aerospace sheet metal part annotated with its features (a) as well as the content of its feature file (b).....143
Figure 5.14	A less common design of a hydroformed aerospace sheet metal part annotated with its features (a) as well as the content of its feature file (b).....144

Figure 6.1	A 3D model representation of an ASM part (a).....	147
Figure 6.2	Two other representations of an ASM part.....	148
Figure 6.3	The part models used to test the MDI prototype and a relatively similar real ASM part for each.....	151
Figure 6.4	The color-coded representation of (a) the volumetric comparison of the part model presented in Figure 6.3 (a) and its modified version, and (b) the MDI prototype's output.....	153
Figure 6.5	The color-coded representation of (a) the volumetric comparison of the part model presented in Figure 6.3 (c) and its modified version, and (b) the MDI prototype's output.....	155
Figure 6.6	The color-coded representation of (a) the volumetric comparison of the part model presented in Figure 6.3 (e) and its modified version, and (b) the MDI prototype's output.....	157

LIST OF ABBREVIATIONS AND ACRONYMS

AAG	attributed adjacency graph
AFR	automated feature recognition
ASM	aerospace sheet metal
B-rep	boundary representation
CAD	computer aided design
D2	distance distribution
DF	deformed flange
EBC	edge boundary classification
HAAG	holistic attributed adjacency graph
ICP	iterative closest point
MDI	model difference identification
MLAG	multi-level attributed graph-based method
OOFF	object-oriented feature finder
PCA	principal component analysis
PLM	product lifecycle management
STEP	standard for the exchange of product

INTRODUCTION

A product's lifecycle includes all the phases that the product goes through, from design to retirement. Product lifecycle management (PLM) can be defined as a product-centric lifecycle-oriented approach to sharing data among the actors involved in all of the phases of a product's lifecycle to achieve desired goals (Terzi, Bouras, Dutta, Garetti, & Kiritsis, 2010). A product's lifecycle is classically divided into beginning of life, middle of life and end of life (Figure 0.1). One of the main tools used at the beginning of a product's life is computer aided design (CAD), and it helps with both defining the product using CAD models and downstream processes including process planning and manufacturing.

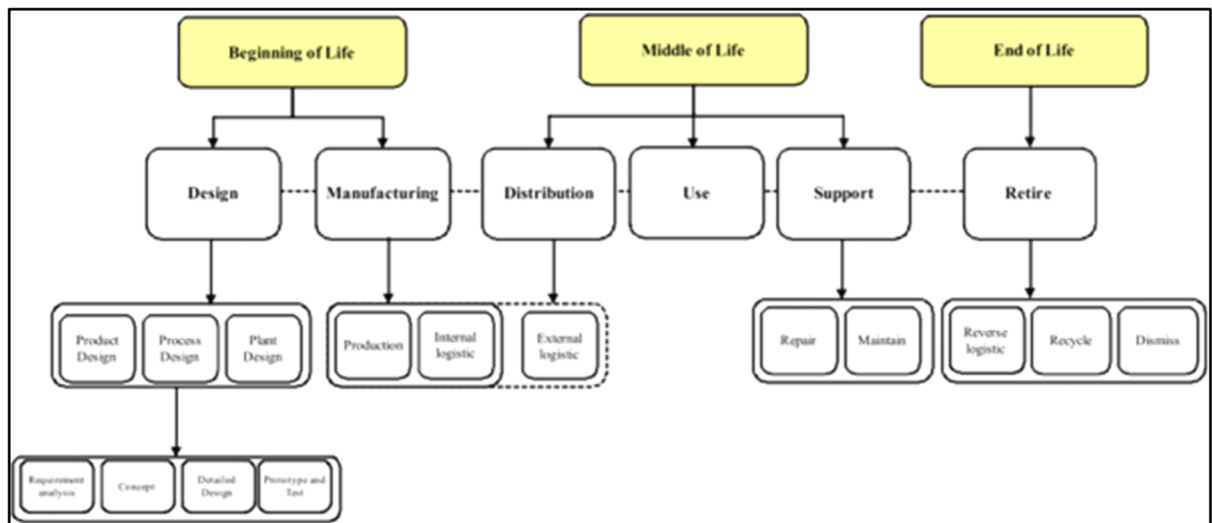


Figure 0.1 The basic phases of a product throughout its lifecycle
From Terzi et al. (2010)

One way that a PLM solution can improve a product in its beginning of life is information reuse. Information reuse boosts the product development process by reducing repetitive activities. It can be implemented in a PLM solution by searching for relevant information and identifying the differences between pieces of information. Given that CAD models are pivotal to the development of modern products, their reuse, and consequently the search for information and difference identification, are very important.

Finding similar CAD models makes it possible to reuse relevant information, for example, manufacturing sequences, material-related remarks and pertinent costs, and avoid the need to repeat the associated activities in the new product's development process. Once the similar models have been identified, finding the differences between them helps to filter out the relevant pieces of information about the reference product to reuse in the development of the new product. In general, CAD model searching and model difference identification (MDI) facilitate not only product information reuse but also product rationalization and standardization, CAD model management, CAx model authorization, CAD data translation/premastering, and engineering change management (Brière-Côté, Rivest, & Maranzana, 2013). From an execution point of view, CAD model searching and MDI are both based on CAD model comparison; however, MDI requires detailed comparison results. MDI can be divided into two basic steps: (1) identifying and measuring the relative differences between a reference model and the model being compared to it (the CAD model comparison part), and (2) representing the differences identified with respect to the reference model.

CAD model comparison methods have been categorized in different ways in previous studies.

1. Some comparison methods are based on the geometric information of the model.
 - a. For example, global geometrical properties such as volume, surface area, and moment of inertia have been used to compare CAD models; however, they do not provide information about position or the nature of differences. Still, due to their very high computational efficiency, these properties are used to detect duplicates (Gear & Visser, 2003).
 - b. Another way of comparing CAD models based on their geometric information is the point-to-part method. Point-to-part is based on the Hausdorff metric, which measures how far two subsets of two CAD models are from each other. The reference CAD model is discretized, and the distance between each of the points created on the reference model and on the compared model is calculated and called the forward Hausdorff metric. The backward Hausdorff metric is also calculated, by discretizing the compared model instead and repeating the same calculation process. The higher value of the two (forward and backward) Hausdorff metrics is called the Hausdorff metric (Aspert, Santa-Cruz, & Ebrahimi, 2002).

- c. CAD model comparison technologies have also implemented spatial occupancy-based comparison methods, which involve superposing the CAD models and using Boolean operations to calculate the differences between them (Brière-Côté, Rivest, & Maranzana, 2012b; V5-R2016, 2016). The differences are calculated in terms of added volumes, removed volumes and common volumes, and represented with color coding.
- 2. Other comparison methods are based on the data structure of the models.
 - a. Persistent identity-based matching is a method that has been implemented by some CAD model comparison technologies. It involves comparing CAD models' constituting elements based on the "persistent identity" that was assigned to them when they were created or modified (PRO/Engineer Wildfire 5.0, 2009).
 - b. Signature-based model comparison, on the other hand, uses a wide variety of signatures that correspond to the constituting elements of a CAD model (Msaaf, Maranzana, & Rivest, 2007). The signatures could be geometric (determined based on the geometric properties of the element), descriptive, such as the name of the element, or a combination of several geometric and descriptive signatures.
 - c. Syntax-based matching has also been proposed as a comparison method, and its main contribution has been improving computational efficiency.
- 3. Yet other comparison methods are based on the representation scheme of the CAD models.
 - a. Procedural representation is the most widely implemented representation scheme in CAD technologies (Brière-Côté, Rivest, & Maranzana, 2012a). However, due to the non-uniqueness of the procedural representation of CAD models, the comparison results could be irrelevant.
 - b. Boundary representation (B-rep) relies on vertices, edges, faces, loops and shells to explicitly represent a CAD model. B-rep based comparison methods compare CAD models based on their geometric and topological information.
 - c. CAD model decomposition representation uses uniformly sized voxels to form the spatial occupancy of a model. Decomposition representation-based CAD model comparison methods basically compare the presence of voxels.

4. And finally, there are also feature-based comparison methods that are best equipped to return detailed information that is engineeringly meaningful. Despite its advantages, feature-based CAD model comparison has remained a rather neglected area of research.
 - a. It is commonly proposed to convert feature-based CAD models to graphs and use graph comparison methods to compare the CAD models. The many methods proposed to convert features' structures to graphs include various levels of detail in an attempt to improve the comparison results. However, certain limitations exist that have not been addressed. For instance, Smit & Bronsvort (2007) presented a way to describe the differences between two models in terms of their features and concluded that the problem is resolvable only if the models' features can be mapped.

This study proposes feature-based comparison for semantic CAD MDI. A new approach is proposed to avoid the drawbacks of the previous research, which are primarily a lack of semantics or details in the comparison results and unestablished semantic pose registration. To address these drawbacks, this research proposes innovative use of feature recognition for semanticity, feature-structure-based pose registration and domain-specific feature-based MDI. The outcome is a proof of concept for the use of feature-based CAD MDI to facilitate reuse of product information. Feature-based CAD MDI can represent the differences between compared CAD models with the semantics needed for the engineering activities that benefit from it. MDI-beneficiary engineering activities range from design (3D modeling, detailed design and engineering modeling, etc.) to downstream manufacturing (process planning, tooling, etc.) activities. A semantic CAD MDI improves product development by shortening development time, and reducing design iterations and efforts, and is a valuable capability for future PLM solutions.

This thesis is paper-based, and Chapters 3, 4 and 5 present the published papers. Chapter 1 presents the problem statement. Chapter 2 presents a review of the literature related to the chapters that follow it. Chapter 3 presents the paper on the automated feature recognition method proposed for aerospace sheet metal parts, which is validated in the paper presented in Chapter 5. Chapter 4 presents the paper on the MDI method proposed for aerospace sheet metal

parts, which is validated in Chapter 6. (The validation of the MDI method presented in Chapter 4 has not been published yet.) This thesis comes to an end with a discussion of the results in Chapter 7 and the conclusion thereafter.

CHAPTER 1

PROBLEM STATEMENT AND OBJECTIVES

1.1 Problem statement

In an industrial scenario in which a process planner, for example, benefits from reusing or taking inspiration from the process plans of parts having similar shapes to the current project's parts, a 3D shape search solution makes it possible to search through a repository of part models from previous projects. The solution makes it possible to retrieve and reuse the process plans of the similar previous parts through a model comparison mechanism that is based on the models' similarities. However, the full potential of reusing the existing information, in our example, the process plans, cannot be reached without clarifying which portions of the information are usable. Hence, the process planner would benefit greatly from a model difference identification (MDI) solution that complements the 3D shape search solution. The MDI solution would identify and characterize the differences between the current project's parts and the similar models found in the repository.

In another scenario, a model of a part is distributed, for example, between design partners, and modified unilaterally. An MDI solution could identify and measure the differences between the various versions of the model that exist to validate undocumented modifications, which are basically differences between the versions, for evaluation and approval. In both of the above scenarios, and many others, it is important that the differences between part models are identified and represented at an appropriate semantic level to support rationalizing their impact on their related downstream engineering tasks and to facilitate user interactions.

The above-mentioned scenarios reveal the importance of MDI in effectively reusing information and avoiding engineering rework and non-value-added activities. The MDI solution's ability to represent differences in a way that is informative for engineering is key to its successful application. Although many MDI solutions and methods exist, they have not yet fully succeeded at representing differences in an engineering semantic way. For example, when the difference between two models is the displacement of a hole, available solutions identify

it as the addition of material at the original place of the hole and removal of material from the new place of the hole (Brière-Côté et al., 2013), while the engineering semantic is the displacement of the hole.

The semantic representation of differences between part models is difficult due to the diversity of the CAD solutions used across industries, the non-uniqueness of modeling sequences, and the use of low-level information, e.g., STEP files, in engineering communications. Various CAD solutions, and different versions of each one, are used for the same application domain in different industries. The differences that exist in the data structure of the CAD models produced by each solution (the native models) prevent their direct applicability in MDI. Moreover, the data structures of the native models produced by latter versions of a modeling solution are unlikely to be identical.

A part can also be modeled in multiple ways within a single modeling solution and using the same modeling operations; this is known as the *non-uniqueness of modeling operation sequences* (Brière-Côté et al., 2012a). Hence, even if the same modeling solution is used and the same part is modeled, two models of that part may be considered different because the modeling operation sequences may be non-unique. Because of this inevitable “*false positive*,” even if all the mainstream modeling solutions were to use the same relatively high-level and engineeringly meaningful modeling operations, utilizing them in MDI would be impractical.

Even if neither the data structure differences between modeling solutions or their versions nor the non-uniqueness of modeling operation sequences hindered successful MDI for native models, using modeling operations to identify differences between models does not necessarily express engineering semantics. For example, a shaft could be modeled by the *pad* operation in CATIA V5, which is not engineeringly semantic.

Moreover, native models are likely to be converted into low-level information like B-rep models to facilitate engineering communication. When they are, the concern for using native models in MDI is even less relevant. Given that implementing MDI with native CAD models

is far from ideal, and that B-rep models and other low-level modeling schemes, which are popular and largely supported by CAD solutions, lack the semantic representation needed to enable a semantic MDI, it seems necessary to look elsewhere for a solution.

A sensible solution would be to take measures to elevate the level of the information in low-level CAD models, like B-rep models, that are already commonplace in engineering communications. A measure that invariably elevates the level of the information in, for example, B-rep models, would solve all of the aforementioned hindrances to a semantic MDI. Automated feature recognition (AFR) has been extensively researched to elevate the level of information of CAD models (Y. Shi, Zhang, Xia, & Harik, 2020). Feature models created by the AFR process are composed of features that represent high-level engineering semantics.

Although AFR seems at first glance to be the answer to elevate the level of the information in low-level models for a semantic MDI, AFR poses its own challenges. Both the AFR research and existing AFR solutions are suboptimal. Developing a comprehensive approach to AFR is a huge undertaking, and AFR has hence suffered from insufficiently precise outcomes. Even the domain of application specific to AFR approaches is challenged by disparities in the feature taxonomy used within a given domain of application across different industries. It seems that the challenges that are specific to a given AFR solution need to be addressed first to successfully implement AFR in an MDI solution. One option is to narrow the scope of the AFR solution in question to a specific domain of application in a specific industry, for example, structural sheet metal parts in the aerospace industry, which is explored in this work.

Sheet metal parts are generally widely used to produce various lightweight structures for office furniture, server management housing, computer casings, etc. Similarly, aerospace sheet metal (ASM) parts are very important in the aerospace industry as an established way of creating lightweight airframes. Although these parts are gradually being replaced by composite parts in new aircraft design, the engineering maintenance and modification work that older aircraft designed using sheet metal require as well as established fabrication know-how keep them

relevant. The comparison of ASM parts is highly valuable to be able to reuse relevant information from previous designs.

1.2 Research objectives

The general objective of this research is to propose a model difference identification method for structural aerospace sheet metal parts that represents the differences between 3D CAD models in a meaningful way through design features. Our specific objectives are to:

1. Propose an automated feature recognition method for structural aerospace sheet metal parts to elevate the level of information of the parts' B-rep CAD models.
2. Propose a CAD model difference identification method that uses the feature models of structural aerospace sheet metal parts to represent the differences between their CAD models semantically.

1.3 Methodology

In short, this study first proposes a novel AFR method for ASM parts with a focus on manufacturing features like shearing features and deforming features. Next, it proposes a novel semantic MDI method that encompasses both semantic pose registration and semantic difference identification. The proposed methods are then verified and validated by prototyping them and testing the prototype programs with real-world samples. The samples are fabricated part models that were created from actual parts and altered in their proportions to prevent infringing upon intellectual property rights.

CHAPTER 2

RELATED LITERATURE

The fundamental literature that is pertinent to this research is reviewed in this chapter. First, the AFR literature is reviewed to introduce major AFR methods and the latest research on them. Next, the AFR literature related to sheet metal part CAD models is briefly reviewed. Then, the main literature that has contributed to CAD model comparison over the last decade is reviewed. Note that Chapters 3, 4 and 5 also each contain their own review of the literature specific to their respective papers.

Before taking a look at the AFR literature, it is best to briefly review the concept of features and feature-based CAD models. In this research, features are considered portions of a CAD model that are significant for at least one of the phases of a product's lifecycle and have attributes to be described by. For example, a hole is a feature that could be part of the attachment method incorporated in the design and needs a manufacturing operation to be created in the part. A hole is described by its attributes: diameter, depth, type (e.g., threaded, non-threaded), etc. Feature-based CAD models are preferred over technical drawings and geometric models, which have traditionally been the mainstream means of engineering communication, for engineering communications due to their ability to convey semantic information. One example of semantic information is design intent, and geometric CAD models are unable to convey it. Feature technology's ability to convey semantic information as well as its ease of design and modification, self-adequacy to contain part definitions (called model-based definition), and ability to be converted to technical drawings and geometric models if need be, among many other reasons, make it important in engineering.

2.1 Automated feature recognition for CAD models

The AFR literature can essentially be divided into rule-based methods (Bojan Babic, Nesic, & Miljkovic, 2008) and artificial neural networks (Babić, Nešić, & Miljković, 2011). The former is of utmost relevance to the approach used in this research and the type focused on here.

The rule-based methods can be further sub-categorized as (1) syntactic pattern recognition methods, (2) logic (if-then) rules and expert systems (Bojan Babic et al., 2008), (3) graph-based approaches, (4) convex hull and cell-based volumetric decomposition methods, and (5) hint-based approaches (Y. Shi et al., 2020). Table 2.1 shows the references selected (considered relevant to this work) for each of the rule-based sub-categories.

Table 2.1 The rule-based AFR methods and their relevant references

AFR methods	References
Syntactic pattern recognition methods	(Ismail, Bakar, & Juri, 2002, 2005; Jain & Kumar, 1998)
Logic (if-then) rules and expert systems	(B Babic, 1996; BR Babic & Miljkovic, 1997; Bouzakis & Andreadis, 2000; Henderson & Anderson, 1984)
Graph-based approaches	(Campana & Mele, 2020; Y. G. Li, Ding, Mou, & Guo, 2009; Malyshev, Slyadnev, & Turlapov, 2017; Sun, Huang, Chen, Wang, & Wan, 2012; V. B. Sunil, R. Agarwal, & S. S. Pande, 2010)
Convex-hull and cell-based volumetric decomposition methods	(Y. S. Kim, 1992; E. Wang & Kim, 1998; Woo, 2003; Jian Zhang & Li, 2016)
Hint-based approaches	(J. Han & Requicha, 1997; Vandenbrande & Requicha, 1993)

2.1.1 Syntactic pattern recognition methods

Syntactic pattern recognition methods are based on the CAD model representation that is initially translated into a set of primitives (a string) written in a descriptive language (Bojan Babic et al., 2008). Sets of patterns in the syntax are associated with shape features.

The method presented by Jain et Kumar (1998) is especially interesting because it uses a wireframe (3D) part representation model. It is developed only for prismatic parts. The wireframe model is translated in a 2D vertices-edges graph for each of the six boundary planes

of a parallelepiped. This method can recognize form features like holes, steps, slots, and protrusions with orthogonal boundary faces. A hole is considered a basic feature, and all the others are derived from it – steps are holes without two faces, slots are holes without one face, and protrusions are treated like a combination of slots and steps manufacturing.

The edge boundary classification (EBC) method proposed by Ismail et al. (2002, 2005) uses spatial addressability information from solid models and identifies the solid and outer sides of a boundary element. For each edge loop identified in the part's B-rep model, an EBC pattern can be formed by classifying a set of test points (located in proximity to the edges that form the loop) with reference to the solid model. Depending on whether these test points correspond to a solid or outer space, they are coded, and the code string for each edge in the loop forms the pattern that can be used for form feature recognition. This approach can be applied to recognize some features in parts to be produced using 2D numerical control (NC) machines – pockets, slots, and steps consisting of planar or semi-cylindrical faces (Ismail et al., 2002) – as well as cylindrical and truncated conical features in both prismatic and rotational parts (Ismail et al., 2005). The advantage of this method is that it is unaffected by geometric and topological variations, except when they affect primary faces. This means no post-processing is needed. The shortcoming of this method is its complicated pre-processing: extracting relevant geometric and topological data, presenting them in a format that is suitable for the EBC algorithms, and creating the spatial addressability information and then the EBC patterns.

2.1.2 Logic (if-then) rules and expert systems

Henderson et Anderson (1984) introduced the logic approach as a set of production rules written in the format *IF C1, C2, C3 . . . Cn THEN A* that define form features and provide the patterns for AFR. If the conditions (*C1, C2, C3*) that represent some patterns are satisfied, then the structure in the part representation that corresponds to them is recognized as the corresponding form feature *A*. This approach specifically implements logics in the geometric information extraction level; hence, no particular part representation needs to be defined for geometric feature extraction, contrary to all the other methods.

Other similar methods are presented in studies published by, in which a model of the part is translated from a 3D solid modeler to initial graphics exchange specification (IGES) and then the IGES data are converted to Prolog facts using a utility program. The first stage of the recognition process is face extraction and base face determination. A base face is a feature face that is concave adjacent to at least one feature face. Then, the boundary faces are determined. Apart from holes, the main criterion for form feature matching is the number and type of boundary faces. The features that a system may recognize are pockets, slots, blind slots, steps, corner steps, holes, blind holes, and countersunk holes. A very similar system that also applies logic rules to a set of data obtained from a neutral IGES file was developed by Bouzakis et Andreadis (2000) for the computer aided process planning (CAPP) of prismatic parts.

2.1.3 Graph-based approaches

In the mid-1980s, Joshi (1987) developed a graph-based approach to form a representation model in which the topological information and some geometric information about the part are preserved. The author proposed an attributed adjacency graph (AAG) in which a B-rep model of the part (designed in a solid modeler) is transformed. An AAG is a type of graph in which every arc has the attribute “0” if its nodes are concave adjacent or “1” if they are convex adjacent.

Y G Li, Ding, Mou, et Guo (2010) proposed a graph-based method that incorporates a variety of geometric and topological information from a product’s B-rep model. They called their graph a holistic attributed adjacency graph (HAAG) and suggested that it contains all of the information necessary to recognize both generic and freeform features; however, they did not provide any example of freeform AFR. In addition, it seems that they did not follow the conventional terminology used by other researchers, as they used “hint” instead of “rule” and vice versa. Nonetheless, this study is interestingly aeronautic industry-oriented.

In an outstanding work, Sun et al. (2012) proposed a multi-level attributed graph (MLAG)-based method that represents faces and edges at a low level and simple shape features at a high level. Both the low-level graphs and the high-level simple feature graphs are

attributed. The nodes of the high-level graphs are also referred to as “subgraphs” because they represent a subgraph of the low-level graphs. The high-level graph’s representation of the model is used to identify complex shapes.

However, graph-based AFR methods are inherently inaccurate when it comes to recognizing intersecting features. A number of works have been published attempting to address this downside. For example, V. B. Sunil et al. (2010) proposed a method to augment graph-based AFR methods by first differentiating between the features that have a planar base, like pockets, from those that do not, like passages, and then using geometric reasoning to recognize interacting features. Graph-based feature recognition methods have been applied to recognize features at various degrees of specificity depending on the application. Malyshev et al. (2017) used graph-based AFR for solid model suppression by recognizing only the main features: holes, pockets, bosses, and general holes (complex depressions). Such a simplistic approach could be further expanded to the point that graph-based AFR is used to recognize only *islands*, which encompass both protrusions and depressions (Campana & Mele, 2020).

2.1.4 Convex-hull and cell-based volumetric decomposition methods

The convex-hull and cell-based volumetric decomposition methods approach feature recognition as a process in which a model’s volume is decomposed into intermediate volumes and features are recognized from them. The convex-hull volumetric decomposition methods date back as far as the early 1990s (Y. S. Kim, 1992). In their final version, E. Wang et Kim (1998) proposed:

1. First, the cylindrical shapes of the 3D model are abstracted to polyhedral volumes.
2. Then, the convex-hull volume of the 3D model is calculated.
3. The difference between the convex-hull and polyhedral volumes of the 3D model is calculated and decomposed until it becomes convex.
4. Form features are recognized from the decompositions.
5. Machining features are recognized from the form features based on being a positive or negative form.
6. Cylindrical shapes are returned as machining features.

In a more recent application of convex-hull decomposition for feature recognition, Jian Zhang et al. (2016) proposed a method that can automatically detect convex and concave regions of a tessellated CAD model and further characterise their shapes. The authors' method starts with clustering triangular facets based on each facet's local convexity with its neighbourhood. Triangular facets are classified into local convex groups, local concave groups and mixed groups based on each facet normal's relationship with its neighbouring facets. Subsequently, shape recognition is conducted on the detected regions using Gaussian image formation and the point distribution distance algorithm.

Like convex-hull decomposition methods, cell-based decomposition methods also propose to recognize machining features in the form of machining paths. Bojan Babic et al. (2008) reported that the basic methodology consists of three steps:

1. First, the removed volume is identified as the difference between the blank and the 3D model of the part.
2. Then, the removed volume is decomposed into unit volumes by using the extended boundary faces as cutting planes (cell decomposition).
3. Lastly, all the unit volumes that have common or co-planar faces are merged to form the maximum number of cells that can be removed in a single tool path (cell composition).

In another work, Woo (2003) proposed a volume decomposition method to quickly decompose a solid model into maximal unit volumes. Although maximal volume decomposition is considered an effective way to model and recognize intersecting machining features, its scalability in practical applications is doubted. The fast volume decomposition method presented in Woo's work addresses "the global effect of local geometry" (a problem encountered by its predecessor methods) by using the localized face extension and the seed cells. The global effect of local geometry happens when the areas into which machining features would not extend in a reasonable machining sequence are reached (in the last step). As a result, many unnecessary cells are created, and multiple interpretations are suggested.

2.1.5 Hint-based approaches

Hint-based approaches have been proposed to resolve the problem of arbitrary feature intersection, which graph-based approaches and volumetric decomposition and other methods cannot cope with. Hint-based approaches are a combination of logic approaches and volumetric or graph-based approaches (Bojan Babic et al., 2008). In this type of approach, the topological, geometric and heuristic information about the envisaged part are hints that certain form features are present. The largest volumetric feature that can be identified from a hint is then recognized.

Vandenbrande et Requicha (1993) proposed a method that was originally called Object-Oriented Feature Finder (OOFF) and was based on the notion that the machining operation used to produce a feature leaves a trace in the part boundary even when that feature intersects with another feature. This same method was called “trace-based” in later works. J. Han et Requicha (1997) improved upon the trace-based method by incorporating direct user input, tolerances and attributes, and design features in the hints, and consequently made it a time consuming and computationally expensive process. This improved version of the trace-based method was called IF² for Integrated Incremental Feature Finder.

2.2 AFR for sheet metal parts

Numerous feature recognition methods have been reviewed so far (Babić et al., 2011; Langerak, 2010; Y. G. Li et al., 2009; Verma & Rajotia, 2010; Q. Wang & Yu, 2014). There are comparatively far fewer studies on sheet metal AFR, and they are very specialized. While there exist some rare general AFR methods for sheet metal models, such as the one proposed by Nnaji, Kang, Yeh, et Chen (1991), the majority of the methods in the literature can be categorized as shear AFR methods, generic deformation AFR methods and freeform AFR methods. Shear AFR methods focus on recognizing features that have been sheared off a piece of sheet metal to form a portion of the part. Generic deformation AFR methods focus on recognizing features that have been formed by a forming mechanism, i.e., bending, drawing, or a mix of both. Freeform AFR methods are focused on recognizing complicated form features

that have been created by complex drawing. The literature on freeform AFR methods (Gupta & Gurumoorthy, 2012; Sunil & Pande, 2008; Chunjie Zhang, Zhou, & Li, 2009) is not relevant to the scope of this work and hence not reviewed.

Jagirdar et al. (1995) proposed an early AFR method for recognizing shear features based on geometric and topological reasoning. In general, shear features are created by shearing operations like blanking, notching, piercing and cutting off. Although the method in Jagirdar et al.'s work focused on recognizing shear features, it also recognized features that were formed by shear and deformation operations, such as bridges. In another work, Devarajan et al. (Devarajan, Kamran, & Nnaji, 1997) proposed an AFR method for shear features to offset their profiles for layout punching tool path specification. Their method recognized the portions of parts that cannot be punched (removed by punching) and therefore need to be cut out using special tools. A change in punch diameter alters the features recognized and would thereby alter the outputs. Also, Kannan and Shunmugam (Kannan & Shunmugam, 2009b) used a center-plane model to calculate the shear layout of sheet metal parts, which could then be used to recognize shear features.

Generic sheet metal deformation features are the result of pure deformation or shear and deformation operations; hence, they always have traces of deformation. Liu et al. (2004) presented a fundamental study that addressed some of the main issues associated with sheet metal parts, including feature intersection and array features. Feature intersection occurs when a few features intersect in such a way that one of them is split or misses certain topological entities. An array feature is made up of repeated features. Remarkably, these issues have not been considered in subsequent works.

Kannan and Shunmugam (Kannan & Shunmugam, 2009a, 2009b) took two significant steps in sheet metal AFR: proposing a method based on STEP AP 203 to improve its applicability, and proposing to calculate and utilize a center-plane model. Their proposition to use a center-plane model has two elements to it: it proves that a reduction in topographical information improves computation performance and does not necessarily reduce the useful

information available; however, it adds a step to the AFR method. Before Kannan and Shunmugam, Jagirdar et al. (Jagirdar, Jain, & Batra, 2001) assumed that a sheet metal model can be represented by its center-plane and accordingly proposed their own AFR method. They considered the intersection of features in sheet metal models to be limited to “cross-bend” features (features that pass through a bend) and proposed a technique to identify them. Even though their proposed technique is not general, their research is quite unique in that subsequent studies have neglected to consider the intersecting features problem entirely. In one of the most recent studies, Gupta and Gurumoorthy (Gupta & Gurumoorthy, 2013) propose a general solution to recognize generic deformation features. In their proposed method, cylindrical, conical, spherical and toroidal faces are considered transitive entities to characterize deformation. This method is the most geometrically general of all the methods published. All in all, the literature on sheet metal AFR was particularly inspiring for the method proposed in this research in that it clarified the applicability of representing a part by its center plane. The center-plane representation of a sheet metal part was further simplified in this work to just one side of the part.

2.3 Semantic CAD model comparison

A novel way of categorizing the previous works is proposed here to take a more constructive approach to reviewing the literature on CAD model comparison, particularly with regard to semanticity. The literature on CAD model comparison can be broken down into the following categories:

1. *Similarity identification*, which focuses on comparing CAD models to identify or quantify their commonalities.
2. *Difference identification*, which focuses on comparing CAD models to identify or quantify their differences.
3. *Part model comparison*, which compares the CAD models of individual components.
4. *Assembly model comparison*, which compares CAD models of assemblies of components.
5. *Graph comparison*, which involves converting CAD models to graphs and comparing the graphs to identify the models’ similarities or differences.

6. *Shape feature comparison*, which involves converting CAD models to shape features and descriptors, and comparing the features and descriptors to identify the models' similarities or differences.
7. *Topology comparison*, which involves comparing CAD models based on the topological information about their shape.
8. *Geometric comparison*, which involves comparing CAD models based on their geometric information.
9. *Global comparison*, which involves comparing multiple CAD models to identify the ones with the most similarity.
10. *Partial/local comparison*, which involves comparing multiple CAD models to identify the ones with a specific subset of their constituting elements in common.

In this work, the possibility of comparing CAD models at a higher abstraction level than is generally found in the literature, such as models' geometry and topology, is explored. This higher abstraction level is considered semantic comparison. The following viewpoints are taken for the literature review to be able to analyze semanticity in the CAD model comparison literature and position this work relative to pertinent previous works:

1. *Comparison subject*, which is the type of information that a comparison solution compares, i.e., a part or assembly model.
2. *Comparison interface*, which is the type of data that the user interacts with the comparison solution, e.g., sketches, 3D CAD models. A comparison interface may be converted to a different type of data.
3. *Comparison medium*, which is the type of data that the comparison interface is converted to mainly for comparison, e.g., graph, code, sketch, B-rep model, 2D model. In some cases, no conversion occurs, and the comparison interface and the comparison medium are the same.
4. *Comparison problem*, which is the how the problem of comparing comparison mediums is formulated, e.g., graph matching, code comparison.
5. *Comparison solving method*, which is how the comparison problem is solved. Which method was used to solve, for example, the graph matching problem?

6. *Comparison purpose*, which is the application of the comparison, e.g., difference identification, similarity assessment.

In this thesis, “semantic comparison” is considered a form of comparison that returns comparison results that are meaningful to the user, i.e., engineeringly meaningful. Essentially, a semantic comparison method is distinguished from a non-semantic comparison method by its comparison medium, problem formulation and comparison solving method; however, it is important to look at the comparison subject and interface for context. These viewpoints are reviewed in the following sections.

2.3.1 CAD model comparison subjects

The two most common subjects of CAD model comparison are part models and assembly models. The research on comparing assembly CAD models can be divided into two categories: (i) comparison of only the shape of the assemblies’ constituent parts, and (ii) comparison of the shape of the assemblies’ constituent parts as well as the assembly relationships between the parts (Lupinetti, Pernot, Monti, & Giannini, 2019). Both categories of assembly CAD model comparison include part CAD model comparison. In some works assessing the similarities between assembly CAD models, the researchers (S. Q. Tao & He, 2012; S. Q. Tao & Huang, 2012) used *shape features* like surface properties, contact surface properties, component area, and volume to represent the parts of the compared assemblies as directed graphs. Chen, Guo, Bai, and Gao (2010) chose to convert the parts’ CAD models into simplified shapes and then represent the compared assembly CAD models as graphs. What these works have in common is that their assembly comparison problems are reduced to graph matching, which is not effective for semantic comparison. Graph matching based on the parts in an assembly fails to factor in the parts’ semantics like function and importance. It is worth noting that graph matching is not the only way to turn assembly CAD model comparison into a non-semantic problem. For example, shape distribution-based methods are another type of non-semantic comparison solving method that are used with assembly CAD models (H. Kim, Cha, & Mun, 2017). However, not all of the works on assembly CAD model comparison convert the comparison problem to a non-semantic problem or suggest non-semantic solving

methods (Z. Han, Mo, & Hao, 2019; Z. Han, Mo, Yang, & Hao, 2018; Jie Zhang, Zuo, Wang, Yu, & Li, 2016). The research on part CAD model comparison also includes both semantic and non-semantic routes. Both types of routes are further discussed in the following sections.

2.3.2 CAD model comparison interfaces

In the literature, the interfaces that the user interacts with in the comparison solution vary and could be confused with comparison mediums. For example, in one work, the user could interface with the proposed CAD comparison solution using a 2D model, and in another, the user could interface with the proposed CAD comparison method using a 3D model from which 2D models are extracted for comparison. The comparison interface is the 2D model in the first case and the 3D model in the second, whilst the comparison medium is the 2D model in both cases. Many CAD model comparison interfaces are mentioned in literature, and Table 2.2 presents the main ones: sketches, unspecified CAD models, native 3D CAD models, and B-rep 3D CAD models. The “unspecified CAD models” category refers to the works that do not indicate any specific type of CAD model and describe the CAD model used as, for example, a feature-based CAD model.

Table 2.2 The comparison interfaces and their references*

Comparison interface	References
<i>Sketches</i>	(Y.-J. Liu et al., 2013; X. Wang, Wang, & Pan, 2014; Chao Zhang, Zhou, Yang, Xiao, & Yang, 2019)
<i>Unspecified CAD models</i>	(Brière-Côté, Rivest, & Maranzana, 2011; Brière-Côté et al., 2012b; Ding, Zhang, Yu, & He, 2013; X.-Y. Gao, Li, & Zhang, 2020; X.-Y. Gao, Zhang, & Lu, 2015; Hou & Ramani, 2006; Jeon, Lee, Hahm, & Suh, 2016; M. Li, Zhang, Fuh, & Qiu, 2009; S. Tao, 2018; Jiale Wang, Jiang, & He, 2010; Chao Zhang & Zhou, 2019; Chao Zhang et al., 2019; Zhao, Liu, & Zhang, 2017; Zhuang, Zhang, Hou, Zuo, & Liu, 2017)
<i>Native 3D CAD models</i>	(Chatelain, Maranzana, & St-Martin, 2002; Chowdhury & Siddique, 2009; Dawei, Guangrong, Yi, & Zhang, 2012; Harik & Barakat, 2010; M. Li, Fuh, Zhang, & Qiu, 2008; M. Li, Zhang, & Fuh, 2010; M. Li et al., 2009; M. Li, Zhang, Fuh, & Qiu, 2011; Min, 2011; Qin, Gao, Yang, Li, & Bai, 2016)
<i>B-rep 3D CAD models</i>	(Cuillière, François, Souaissa, Benamara, & Bel Hadj Salah, 2011; Wei Gao, Gao, & Liu, 2005; W Gao, Gao, Liu, Bai, & Hu, 2006; Huangfu, Zhang, & Yan, 2017; Z. Li, Zhou, & Liu, 2015; Ma, Wang, Cai, & Wang, 2019; Paterson & Corney, 2016; Reddy, Adithan, & Radhakrishnan, 2011; M. Shi & Zhang, 2013; S. Tao, 2014; S. Tao et al., 2013; S. Tao, Wang, & Chen, 2017; Wei & Yuanjun, 2006; Xiaoliang, Shusheng, & Kaixing, 2010; Yin & Guo, 2018; Zehtaban, Elazhary, & Roller, 2016; Zhu, San Wong, Loh, & Lu, 2012)

* The references in which they are used

Even though the comparison interfaces have different levels of semanticity, they may not have an impact on the comparison semanticity because they may be converted to a different type of information to be used as the comparison medium. Comparison interfaces that have a high level of semanticity may be converted to comparison mediums that have lower levels of

semanticity to increase efficiency. Also, comparison interfaces with low levels of semanticity may be converted to comparison mediums with higher levels of semanticity to improve overall comparison semanticity. Therefore, the choice of comparison interface does not correlate consistently with the semanticity of the comparison results.

2.3.3 CAD model comparison mediums

A lot of attention has been paid to the mediums in which CAD models are compared. Table 2.3 lists the comparison mediums mentioned in the literature, the comparison interfaces they appeared with, and the references in which they were found. The comparison medium might be: (1) similar to the comparison interface, for example, if the interface is native feature-based CAD models and the medium is the features (Dawei et al., 2012), (2) a conversion of the comparison interface, for example, if B-rep CAD models are converted to graphs (Ma et al., 2019), or (3) an abstraction of the comparison interface, for example, if semantic information is abstracted from 3D CAD models (Qin et al., 2016). Given the fact that the comparison medium can be an abstraction of the comparison interface, meaning it has the potential for semanticity, it has an important role to play in semantic CAD model comparison. The comparison medium could also determine whether partial/local or global CAD model comparison is used, which have their own applications (M. Li et al., 2009; Ma et al., 2019). Partial/local comparison considers only a predetermined portion of the reference CAD model and compares it to the most similar portion of the target model, whereas global comparison considers the reference and target CAD models in their entirety. This work focuses on the comparison mediums, i.e., graphs, topological information, semantic information, 2D information and shape descriptors. “Semantic information” refers to the comparison mediums that are engineeringly meaningful and significant, like design features.

Table 2.3 The comparison interfaces and comparison mediums*

Comparison interface	Comparison medium	References
<i>Sketches</i>	2D information	(X. Wang et al., 2014; Chao Zhang et al., 2019)
	Semantic information	(Y.-J. Liu et al., 2013)
<i>Unspecified CAD models</i>	Graphs	(Ding et al., 2013; X.-Y. Gao et al., 2015; S. Tao, 2018; Jiale Wang et al., 2010)
	Shape information	(X.-Y. Gao et al., 2020)
	Semantic information	(Brière-Côté et al., 2011, 2012b; Jeon et al., 2016; Zhao et al., 2017)
	2D information	(Jiale Wang et al., 2010; Chao Zhang & Zhou, 2019)
	Shape descriptors	(Cuillière et al., 2011; Hou & Ramani, 2006; M. Li et al., 2009; Zhuang et al., 2017)
<i>Native 3D CAD models</i>	Graphs	(M. Li et al., 2010; M. Li et al., 2011; Min, 2011; S. Tao, 2018)
	Semantic information	(Chatelain et al., 2002; Chowdhury & Siddique, 2009; Dawei et al., 2012; Harik & Barakat, 2010; Qin et al., 2016)
	Shape descriptors	(M. Li et al., 2008)
<i>B-rep 3D CAD models</i>	Graphs	(W Gao et al., 2006; Z. Li et al., 2015; Ma et al., 2019; Paterson & Corney, 2016; M. Shi & Zhang, 2013; S. Tao, 2014; S. Tao et al., 2013; S. Tao et al., 2017; Xiaoliang et al., 2010; Yin & Guo, 2018)
	Semantic information	(Huangfu et al., 2017)
	Shape descriptors	(Reddy et al., 2011; Wei & Yuanjun, 2006; Zehtaban et al., 2016)

* The comparison mediums with which they appear in the literature

Graphs are one of the comparison mediums most commonly referred to in the literature. Figure 2.1 shows an example of a native 3D CAD model and a graphic representation of its features. The CAD models could be represented in graphs by directly converting their content into graphs (M. Li et al., 2010) or abstracting information from their content and representing the abstracted information via graphs (Z. Li et al., 2015). The graphs per se have been used to refine and further abstract the information from the CAD models to enrich the comparison outcome (Harik & Barakat, 2010).

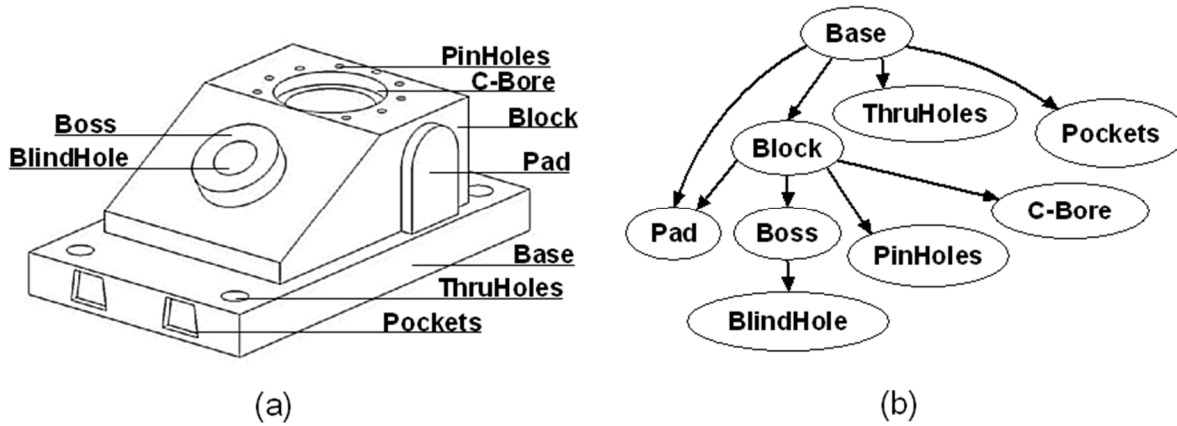


Figure 2.1 Presentation of a native 3D CAD model and its corresponding graph (M. Li et al., 2010)

CAD models' geometric and topological information is commonly represented by graphs for CAD model comparison (X.-Y. Gao et al., 2015; Z. Li et al., 2015; Paterson & Corney, 2016; S. Tao, 2014, 2015, 2018; S. Tao et al., 2013; Xiaoliang et al., 2010; Xu & Jiang, 2018; Yin & Guo, 2018). In the studies listed above, the surfaces, faces or surface regions of the CAD models' boundary representations are represented by vertices in the graphs, with their adjacencies represented by edges. The vertices are usually attributed to geometric information about the faces (Yin & Guo, 2018), their type (convex/concave/planar) (S. Tao, 2015, 2018) and edge count (X.-Y. Gao et al., 2015), the type and direction of the surfaces or surface regions (S. Tao, 2014; S. Tao et al., 2013; Xiaoliang et al., 2010; Xu & Jiang, 2018), topological information about the faces, and geometric information about the underlying surface (Paterson

& Corney, 2016). The edges are usually attributed to the concavity/convexity between adjacent faces or surfaces (Paterson & Corney, 2016; Xu & Jiang, 2018), the type of adjacent faces and the concavity/convexity between them (S. Tao, 2018), the type of shared edges and the concavity/convexity between adjacent faces (S. Tao, 2014; S. Tao et al., 2013; Xiaoliang et al., 2010), the type and relative length of shared edges (Yin & Guo, 2018), the type of shared edges, and the relative length of and concavity/convexity between adjacent faces (S. Tao, 2015).

CAD models' more semantic information other than geometric and topological information has also been represented by graphs for CAD model comparison (Ding, Yu, & Liu, 2014; Ding et al., 2013; Huang, Zhang, Bai, Xu, & Huang, 2015; M. Li et al., 2010; M. Li et al., 2009; M. Li et al., 2011; Min, 2011; M. Shi & Zhang, 2013; Zhao et al., 2017). In those studies, design features acquired from feature-based CAD models (Ding et al., 2013; M. Li et al., 2010; M. Li et al., 2009; M. Li et al., 2011; Min, 2011), design features recognized from CAD models (M. Li et al., 2011; M. Shi & Zhang, 2013; Zhao et al., 2017), and machining features recognized from CAD models (Huang et al., 2015) are represented by vertices in graphs.

The use of topological information as a comparison medium is not popular in the literature. In general, graphs that bear a variety of topological information are preferred. In a rather recent paper, however, X.-Y. Gao et al. (2020) investigated comparing parts using face similarity based on the faces' edge count and adjacency.

Semantic information is not quite commonplace in the CAD model comparison literature due to how difficult it is to extract this information from CAD models. This is evident in the work of Jeon et al. (2016), who proposed to utilize the words in the CAD model (technical terms for tire design) in the first step of the comparison. In later steps, the "CAD objects" in the CAD model are incorporated in concept extraction. Zhao et al. (2017) proposed feature recognition as part of their comparison method to extract manufacturing features as semantic information. They proposed to represent the semantic information as a set of nodes that each represent a piece of semantic information about the CAD model. Qin et al. (2016) took that idea one step

further and proposed to extract modeling features from CAD models and utilize multi-facet ontology mapping to convert them to domain features. This results in uniformly capturing design semantics from CAD models that were potentially generated in heterogeneous modeling solutions. Brière-Côté et al. (2012b) took an entirely different approach and proposed to utilize geometric constraints as a comparison medium. This approach has not received much attention yet.

Two-dimension (2D) information mediums like sketches and drawings introduce interesting possibilities and challenges to the CAD model comparison research. Figure 2.2 shows an example of how a 3D CAD model can be converted to 2D information for comparison. On one hand, 2D information is used for model retrieval, which requires that the CAD models in the database be translated into 2D representations. On the other hand, the query 2D information could be provided by the user (Y.-J. Liu et al., 2013; X. Wang et al., 2014) or provided by the user and/or by translating a 3D model into a 2D representation (Chao Zhang & Zhou, 2019). It is worth noting one important contribution of the work by Y.-J. Liu et al. (2013) is that it accounts for user drawing habits and proposes a statistical user profile to adapt to them.

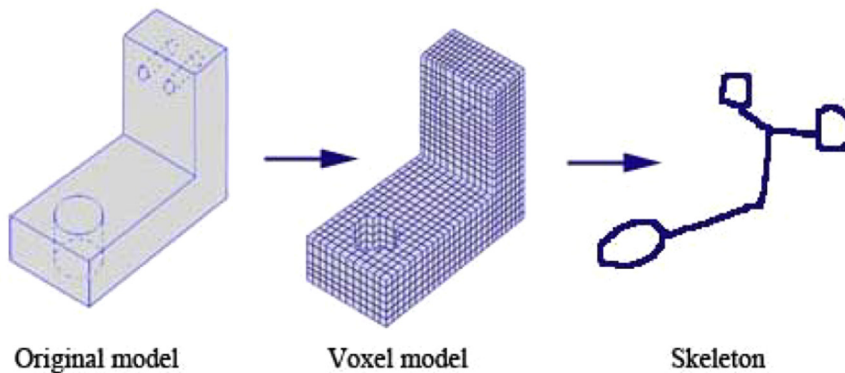


Figure 2.2 Conversion of a 3D CAD model to 2D information
(Jiale Wang et al., 2010)

A variety of comparison interfaces, including unspecified CAD models, native CAD models and B-rep CAD models, have been converted into shape descriptors to be used as the comparison medium. The majority of the works in the literature that focus on converting CAD

information to shape descriptors use distance distribution (D2) histograms to represent their comparison medium and ultimately compare CAD models (Hou & Ramani, 2006; M. Li et al., 2008; M. Li et al., 2010; M. Li et al., 2009; Zhuang et al., 2017). Figure 2.3 shows a few examples of shape conversions to complete and partial shape descriptor histograms. D2 is a common shape function that computes a distance histogram between random points on a model's surface. In more efficient comparison methods, code vectors such as Opitz code vectors (Zehtaban et al., 2016) are used as the comparison medium. Numerical shape descriptors like the eigenvalue of a normalized form of the adjacency matrix of the CAD models are also proposed as comparison mediums for efficient CAD model retrieval (Reddy et al., 2011).

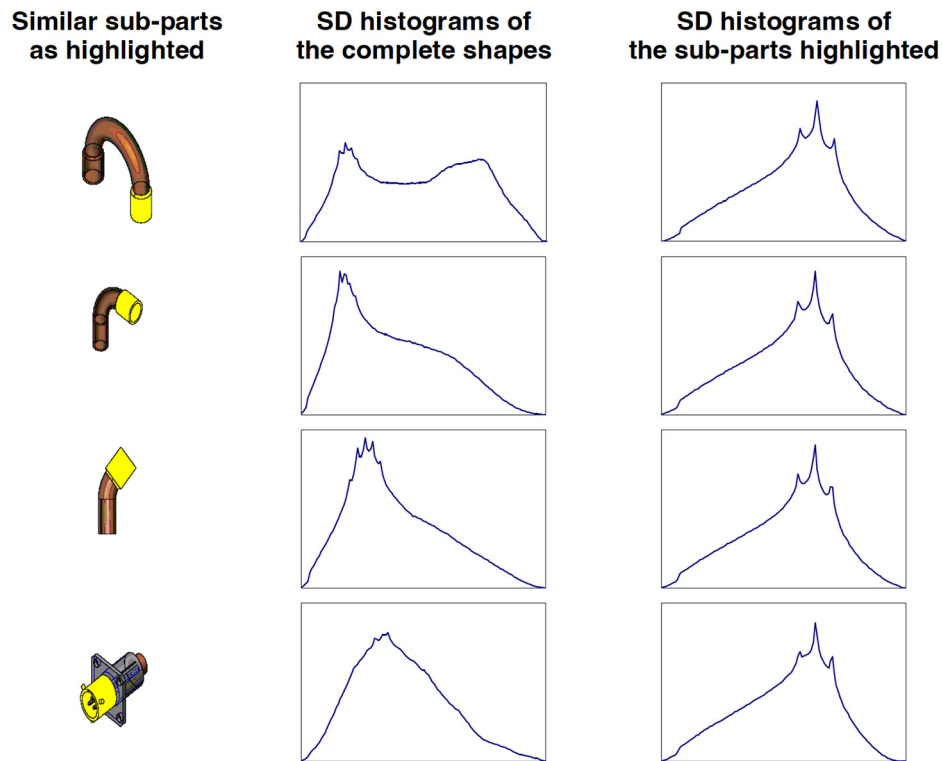


Figure 2.3 Conversion of 3D CAD models to complete and partial shape descriptor histograms (M. Li et al., 2008)

2.3.4 CAD model comparison problems

Different comparison problems have been formulated to compare each of the comparison mediums named in the previous section, i.e., graphs, topological information, semantic information, and 2D information. Table 2.4 lists the comparison mediums and the comparison problems and references in which they appeared in the literature.

Table 2.4 The comparison mediums and comparison problems*

Comparison medium	Comparison problem	References
Graphs	Graph matching	(Ding et al., 2013; W Gao et al., 2006; X.-Y. Gao et al., 2015; Huang et al., 2015; Z. Li et al., 2015; Ma et al., 2019; Paterson & Corney, 2016; S. Tao et al., 2017; Jiale Wang et al., 2010; Xiaoliang et al., 2010; Xu & Jiang, 2018)
	Shape descriptor comparison	(M. Li et al., 2010; M. Li et al., 2009; M. Li et al., 2011; Min, 2011)
	Code (index) comparison	(S. Tao, 2014; S. Tao et al., 2013; S. Tao et al., 2017)
	Finding the optimal mapping matrix	(S. Tao, 2018)
Semantic information	String comparison	(Jeon et al., 2016; Y.-J. Liu et al., 2013)
	Feature comparison	(Dawei et al., 2012; Harik & Barakat, 2010; Huangfu et al., 2017; Qin et al., 2016)
	Explicit geometric constraint re-evaluation	(Brière-Côté et al., 2011, 2012b)
	Commonality index calculation	(Chowdhury & Siddique, 2009)
	(Tree) structure comparison	(Chatelain et al., 2002)
	Optimal matching of complete bipartite graphs	(Zhao et al., 2017)
2D information	Calculating the minimal dissimilarity distance between pieces of 2D information	(Jiale Wang et al., 2010; X. Wang et al., 2014)
	View recognition	(Chao Zhang & Zhou, 2019)

Comparison medium	Comparison problem	References
Shape descriptors	Shape descriptor histogram comparison	(Hou & Ramani, 2006; M. Li et al., 2008; M. Li et al., 2010; Zhuang et al., 2017)
	Numeric value comparison	(Reddy et al., 2011)
	Opitz code vector comparison	(Zehtaban et al., 2016)
	Voxelated shape feature vector comparison	(Wei & Yuanjun, 2006)
	Vector-based geometric representation comparison	(Cuillière et al., 2011)
Shape information	Face similarity computation	(X.-Y. Gao et al., 2020; Yin & Guo, 2018)

* The comparison problems which they are formulated as

The graph comparison mediums reviewed in the literature were generally formulated as graph matching problems, shape descriptor comparison problems, code (index) comparison problems and finding-the-optimal-mapping-matrix problems. Graph matching is a popular comparison problem type for CAD model comparison. The researchers who took this avenue to formulate their CAD model comparisons, for example, Z. Li et al. (2015), enhanced the graphs with topological and geometric information as well as convexity and concavity information. The CAD models with the most matched portions were considered similar and retrieved.

Comparison problems are for the most part directly influenced by the comparison medium. Graph matching is not computationally efficient; hence, some researchers have chosen to formulate their comparison problems differently. Some researchers have chosen to structure their comparison medium as graphs but use shape descriptors for CAD model comparison. Various strategies have been implemented to simplify semantic features, particularly when they are represented by graphs. M. Li et al. (2010); M. Li et al. (2009); M. Li et al. (2011); Min (2011) proposed to simplify the graph and the features to shape descriptors and then calculate the shape descriptors' similarities.

However, formulating graph comparison as code comparison is an even more efficient avenue for CAD model comparison that is extensively explored by Tao and colleagues. S. Tao et al.

(2013) simplified a face attributed relational graph representing a B-rep CAD model into convex, concave and planar regions. Each region was abstracted by an index called *region header code*, each edge on a region boundary was abstracted by a *region relation code* and each face in a region was abstracted by a *face context code*. The comparison problem was formulated as a header code comparison followed by a relation code comparison and face context code comparison once the header code was matched. This comparison problem was extended to general and partial comparison (S. Tao, 2014). The general comparison problem was formulated as the calculation of the number of regions having the same relation and face context codes. The partial comparison problem was formulated as the matching of surface region codes.

The comparison of semantic information can be formulated as string comparison, feature comparison, explicit geometric constraint re-evaluation, commonality index calculation, (tree) structure comparison, and optimal matching of complete bipartite graphs. String comparison formulation of comparing the semantic information of CAD models could yield very efficient solutions. However, since the links between pieces of semantic information are overlooked, this formulation is unable to support high-level semantic comparison. The feature comparison formulation of comparing the semantic information of CAD models, on the other hand, leverages feature occurrence frequency and feature position (Dawei et al., 2012), the feature tree structure (Qin et al., 2016) and manufacturing features (Harik & Barakat, 2010). When it comes to using explicit geometric constraints as the comparison medium (Brière-Côté et al., 2011, 2012b), the comparison problem involves two sub-problems: accurate topological mapping and the successful transposition and comparison of explicit geometric constraints. In other words, the first sub-problem would be efficiently and accurately mapping the topological elements of the compared shapes to create a shape difference model, and then the second sub-problem would be re-evaluating the validity of the compared models' geometric constraints in light of the shape differences. When each CAD model being compared is represented in a medium that incorporates a set of semantic information units about the model, comparison of the two models' similarity can be formulated as an optimal matching problem of a complete bipartite graph consisting of two groups of nodes (Zhao et al., 2017). It is worth noting that

commonality index calculation appears to be the least semantically rich way to formulate semantic information comparison.

The literature on CAD model comparison using 2D information, shape descriptors and shape information suggests the comparison problem can be formulated a variety of ways. The comparison of 2D information mediums can be formulated as calculating the minimal dissimilarity distance between pieces of 2D information or view recognition. The comparison of shape descriptors has been formulated as histogram shape descriptor comparison, numeric value comparison, Opitz code vector comparison, voxelated shape feature vector comparison and vector-based geometric representation comparison. In general, when shape information is directly used to formulate comparison problems, it takes the form of face comparison.

2.3.5 CAD model comparison solving methods

A variety of comparison solutions are proposed in the literature for the comparison problems mentioned. In general, researchers have either adopted an adapted version of a solution for a similar problem in a different domain or invented a solution from scratch. Table 2.5 lists a few solutions identified for some of the comparison problems listed in Table 2.4. Although some of these solving methods have been adopted from other domains of research, they did appear at least once in our review.

A few of the solving methods listed in Table 2.5 for CAD model comparison problems were chosen repeatedly by researchers. The graph representation is a very popular medium for many different types of information, and graph matching is a very common problem formulation. Isomorphism is proposed by (Jiale Wang et al., 2010) to solve a graph matching problem when it entails finding the maximal sub-graph between two graphs. In general, graph isomorphism algorithms are NP-complete; they are known to be feasible for small graphs. To avoid this limitation, Xiaoliang et al. (2010) proposed to use a genetic algorithm, which is a polynomial time solving method. Similarly, Xu et Jiang (2018) proposed to use an ant colony algorithm, which is another polynomial time solving method. A combination of a genetic algorithm and an ant colony algorithm is also adopted to obtain sub-optimal retrieval and retrieval refinement

(Ding et al., 2013). The comparison of D2 histogram shape descriptors is another problem formulation commonly encountered in the CAD model comparison literature. Authors have chosen Manhattan distance calculation or weighted Manhattan distance calculation to resolve this problem.

Table 2.5 The comparison problems and the comparison solving methods proposed for them

Comparison problem	Comparison solving method	References
Graph matching	Isomorphism algorithms	(Huang et al., 2015; Z. Li et al., 2015; Paterson & Corney, 2016; Jiale Wang et al., 2010)
	Isomorphism algorithm developed by Messmer and Bunke (Messmer & Bunke, 1995)	(W Gao et al., 2006)
	Genetic algorithms	(Ding et al., 2013; Xiaoliang et al., 2010)
	Ant colony algorithms	(Ding et al., 2013; Ma et al., 2019; Xu & Jiang, 2018)
	Bipartite graphs	(X.-Y. Gao et al., 2015)
	Softassign quadratic assignment algorithm	(S. Tao, 2018)
Face similarity computation	Ant colony algorithm and Hopfield neural network	(X.-Y. Gao et al., 2020)
Feature comparison	Bag-of-words schema	(Huangfu et al., 2017; Qin et al., 2016)
View recognition	ResNet	(Chao Zhang & Zhou, 2019)
Calculating the minimal dissimilarity distance between pieces of 2D information	Angular radial partitioning	(Jiale Wang et al., 2010; X. Wang et al., 2014)
Shape descriptor histogram comparison (for both graph and shape descriptor mediums)	Manhattan distance	(M. Li et al., 2008; M. Li et al., 2010; M. Li et al., 2009; M. Li et al., 2011; Min, 2011)
	Weighted sum of Manhattan distances	(Hou & Ramani, 2006; Zhuang et al., 2017)
Opitz code vector comparison	Cosine coefficient	(Zehtaban et al., 2016)

2.3.6 CAD model comparison purposes

The purpose of CAD model comparison is either identifying similar models or identifying differences between models. Similarity and difference are two sides of the same coin; however, there is a subtle distinction. Similarity, which is a representation of commonality between two objects, is commutative, whereas difference, which is a representation of the subtraction of objects from one another, is not commutative. This means that the similarity of \underline{a} to \underline{b} is equal to the similarity of \underline{b} to \underline{a} , but the difference of \underline{a} from \underline{b} is *not* equal to the difference of \underline{b} from \underline{a} . Cardone, Gupta, et Karnik (2003) published an insightful survey paper on shape similarity assessment methods with a focus on shape signatures. They provided an inventory of applications for similarity assessment: cost estimation, part family formation, and information reuse. Note that assessing both similarity and difference could be prioritized in some of these applications. For example, similarity assessment would be of priority in cost estimation and part family formation, but in information reuse, both similarity and difference would be equally necessary. Similarity assessment makes it possible to find similar designs, and difference identification is subsequently used to create, modify, remove or add the differences.

CAD model comparison is generally used for the following applications: product information reuse, product rationalization and standardization, CAD modeling management, CAD data translation/remastering, CAx model authoring, and engineering change management (Brière-Côté et al., 2012a). Brière-Côté et al. (2012a) argue that similarity assessment is fundamental to all of these applications except engineering change management. They also argue that difference identification is fundamental to product information reuse, product rationalization and standardization, and engineering change management.

Semantic CAD model comparison could mean different things depending on what the purpose of comparison is – similarity assessment or difference identification. Semantic CAD model comparison to identify the differences between models needs to return semantic results for the differences identified. On the other hand, semantic CAD model comparison to identify the

similarities between models needs to measure similarity using semantic information or methods.

2.4 Synthesis



The work presented in this thesis can be divided into two parts: automated feature recognition (AFR) and CAD model comparison. The AFR portion is focused on sheet metal parts and specifically aerospace structural applications. The CAD model comparison part is focused on semantic model difference identification (MDI).

The AFR method proposed in this work is close to logic rules methods, with a specialized domain of application. Since it is focused on aerospace sheet metal part models, it has three advantages: (1) a very clear-cut set of rules are followed to design such parts, (2) feature taxonomy and feature definition are constrained, and (3) features have distinct design intentions. The rules that are followed when designing such parts are leveraged to facilitate the identification of AFR logic and conditions. AFR is facilitated by the constraining of feature taxonomy and definition. The distinct design intention of features makes semantic CAD model comparison by means of feature comparison possible.

The MDI method proposed in this thesis uses feature models to interface and uses the feature models directly as the comparison medium. Although the features are structured as acyclic graphs in the feature files, the comparison problem is not formulated as a graph-related problem. The comparison problem could be categorized as a feature comparison problem; however, it is unlike any of the feature comparison problems found in the literature. The solving method used for the feature comparison problem was invented from scratch and leverages the distinct design intentions conveyed by each feature in the CAD models' corresponding feature models.

CHAPTER 3

FEATURE RECOGNITION FOR STRUCTURAL AEROSPACE SHEET METAL PARTS

Syedmorteza Ghaffarishahri¹ , Louis Rivest¹ 

¹ Department of Systems Engineering, École de Technologie Supérieure,
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3

Paper published in *Computer-Aided Design and Applications*, February 2020

3.1 Abstract

With the advances in the aerospace industry, specialized tools, e.g. specialized modelling tools for aerospace sheet metal, have been developed to help at various stages of the product lifecycle. Although structural sheet metal parts form a significant portion of airplanes, there is no specialized automated feature recognition (AFR) method dedicated to them. AFR provides unparalleled contributions to various tasks in product lifecycle management, e.g. computer aided process planning, data retrieval and model difference identification. Despite the presence of a number of AFR methods for sheet metal parts, none of them are tuned to recognize the design semantics of the aerospace industry. This work proposes the first AFR method to recognize aerospace sheet metal features and design semantics to elevate the level of abstraction of the information from 3D STEP models. The proposed approach is to first preprocess the 3D STEP model in order to classify the topological elements of the boundary model (B rep model) and create relevant novel face sets and subtypes of faces, face boundaries and edges. Then, rule-based steps are used to recognize aerospace sheet metal features. The extracted features are described by their geometry, their relationship with other features and their pertinent parameters. As a result, the engineering semantics of 3D B-rep models of aerospace sheet metal parts are extracted and could be used for many applications like design reuse and model comparison.

3.2 Introduction

Numerous CAD solutions, including CATIA (Dassault Systems, 2017), SOLIDWORKS (Dassault Systems, 2018), IronCAD (IronCAD LLC, 2017), NX 12 for Design (Siemens PLM Software, 2017a), Solid Edge (Siemens PLM Software, 2017b), BricsCAD Platinum (Bricsys, 2017), Creo Parametric (PTC, 2017), TurboCAD (IMSI/Design, 2017), Onshape (Onshape), Alibre Design (Alibre LLC, 2017), Autodesk Inventor (Autodesk, 2018) and Ansys SpaceClaim (SpaceClaim Corporation, 2017) provide feature-based CAD to model sheet metal parts. Amongst the aforementioned CAD solutions, CATIA and NX 12 for Design provide modelling tools specialized for aerospace sheet metal (ASM) design. These modelling tools provide exclusive features that are commonly used in the design of ASM parts. Such exclusive features include curved flanges, joggles and stringer cutouts, to name a few. In addition, these feature-based ASM modelling tools are specialized to facilitate the design process in the aerospace industry.

The ASM models themselves, however, are not necessarily exchanged in native formats to avoid cross-platform and cross-version conflicts. Instead, they are exchanged via their boundary representation (B-rep) standardized by using Standard for the Exchange of Product (STEP) model data, one of the most effective and efficient information exchange methods (Pratt, 2001). However, the specific features commonly used in the design of ASM parts require the use of feature-based ASM design tools, which cannot be exchanged via STEP. An ASM feature-recognition method for STEP models – a B-rep model – could therefore significantly raise the level of design information exchanged via STEP. This work aims to propose an automated feature recognition (AFR) method for recognizing features from ASM B-rep models based on the definition of non-facetted manifold solid B-rep in ISO 10303-42 (*Industrial automation systems and integration -- Product data representation and exchange - - Part 42: Integrated generic resource: Geometric and topological representation - STEP 10303-42*, 2003). Our long-term goal is to use features to elevate the level of information provided for downstream applications such as 3D model difference identification. This way, we will be able to provide the user with, for example, the difference between two models in terms of their flange length.

3.3 Previous works

While numerous feature recognition/extraction methods have been proposed over the last decades (Babić et al., 2011; Langerak, 2010; Y. G. Li et al., 2009; Verma & Rajotia, 2010; Q. Wang & Yu, 2014), the ones dedicated to sheet metal features are comparatively far less numerous. While there are AFR methods for sheet metal models such as the one proposed by Nnaji et al. (Nnaji et al., 1991), the majority of the methods can be divided into shear AFR methods, generic deformation AFR methods and freeform AFR methods. Being far from the scope of our work, the previous work on freeform AFR methods (Gupta & Gurumoorthy, 2012; Sunil & Pande, 2008; Chunjie Zhang et al., 2009) is not reviewed here. Jagirdar et al. (Jagirdar et al., 1995) proposed an AFR method for identifying shear features based on geometric and topological reasoning. Shear features are sheet metal parts features that are created by shearing operations like blanking, notching, piercing and cutoff. Although the method is focused on the recognition of shear features, it also recognizes features that are formed by shear and deformation operations, such as bridges. Devarajan et al (Devarajan et al., 1997) proposed an AFR method for shear features that is based on profile offsetting for layout punching tool path specification. The offsetting method reveals the portions of parts that cannot be removed by punching, so these portions would need specific tools to be produced. A change in punch diameter could alter the recognized features and thereby introduce inconsistent output. Kannan and Shunmugam (Kannan & Shunmugam, 2009b) used a center-plane model to calculate the shear layout of sheet metal parts. The shear layout could then be used for recognizing shear features.

Sheet metal generic deformation features could be the result of either pure deformation or shear and deformation operations; hence, there are always footprints of deformation in their structures. Liu et al. (Z. Liu et al., 2004) presented a fundamental study that addressed some of the main issues, including feature intersection and array features, although the geometry was limited to cylindrical and planar faces. Feature intersection occurs when a few features intersect such that one of them is split or misses certain topological entities. An array feature

is made up of repeated features. Noticeably, these issues have not been considered in the subsequent works.

Kannan and Shunmugam (Kannan & Shunmugam, 2009a, 2009b) took two significant steps in sheet metal AFR: resting the method on STEP AP 203 for promoting its applicability and proposing the calculation and utilization of a center-plane model. Their proposition to use a center-plane model is twofold: it proves that a reduction in topographical information improves computation performance and that it does not necessarily reduce useful information; however, it adds a step to the AFR method. Before Kannan and Shunmugam, Jagirdar et al. (Jagirdar et al., 2001) assumed that a sheet metal model is represented by its center-plane and accordingly proposed their AFR method. They considered the intersection of features in sheet metal models, as limited to only “cross-bend” features (features that pass through a bend), and proposed a technique to identify the intersecting features.

Even though their proposed technique was not general, subsequent studies have detained the problem of intersecting features entirely; making their research quite unique. Gupta and Gurumoorthy (Gupta & Gurumoorthy, 2013), in one of the most recent studies, propose a general solution for the recognition of generic deformation features. In their paper, cylindrical, conical, spherical and toroidal faces are considered as transitive entities to characterize deformation. This method is more geometrically general than any other published works.

3.4 Term definitions and premises

Before describing the proposed AFR method, there are a number of concepts to be explained. The structure of B-rep models and the ASM feature taxonomy explain the input and output of this work, respectively. The feature definition and description briefly explain the authors’ viewpoint on features to facilitate a better understanding of the proposed method.

3.4.1 Structure of B-rep models

The structure of B-rep models that is used in this study is included in the definition provided by ISO 10303-42 (*Industrial automation systems and integration -- Product data representation and exchange -- Part 42: Integrated generic resource: Geometric and topological representation - STEP 10303-42*, 2003). Here, the B-rep model refers to an exact explicit boundary representation, and thus, a non-facetted representation of an ASM part. The B-rep CAD model **manifold_solid_brep** contains geometric elements, including **surface**, **curve**, and **point**, as well as topological elements, including **connected_face_set**, **closed_shell**, **face**, **face_surface**, **face_bound**, **face_outer_bound**, **loop**, **edge_loop**, **path**, **edge**, **oriented_edge**, **edge_curve**, **vertex**, and **vertex_point**. The **curve** is assumed, based on studying real ASM parts, to be **lines**, elementary **conics** and general parametric polynomial curves, such as **b_spline_curve**. The **surface** is assumed to be only an **elementary_surface** which could be a **plane**, **cylindrical_surface**, **conical_surface**, **spherical_surface**, **toroidal_surface** and **b_spline_surfaces**. A **connected_face_set** is a set of faces such that the domain of the faces together with their bounding edges and vertices is connected (*Industrial automation systems and integration -- Product data representation and exchange -- Part 42: Integrated generic resource: Geometric and topological representation - STEP 10303-42*, 2003). A **face_surface** is a subtype of a **face** in which the geometry is defined by an associated surface. A **face_bound** is a topological entity, constructed by stringing together connected (oriented) edges beginning and ending at the same vertex, and is intended to be used for bounding a face. A **face_outer_bound** is a special subtype of **face_bound** that carries the additional semantics of defining an outer boundary on the face. An **edge_curve** is a special subtype of an **edge**, which has its geometry fully defined. A **vertex_point** is a subtype of **vertex** that has its geometry defined by a point.

3.4.2 Feature definition and description

In this work, we define a feature as: a portion of a geometry model that is significant in at least one of the phases of the product's lifecycle and that can be described by its attributes. Accordingly, the specifications of a part and the information conveyed by its model that are

not reflected through its geometry, i.e. material, material properties, color and coating, are not features, but rather are the characteristics of the part itself.

The attributes of a feature are geometry, feature relationships and parameters. For example, the geometry of a hole is its faces, and its relation to its parent feature is represented by the edges connecting it to its parent feature. The parameters of the hole are its location (defined by an axis) and diameter (derived from its geometry). The geometry therefore links features with their B-rep. The feature relationships expose the feature structure of geometry models. In this work, the feature relationships are demonstrated to be rooted in the topological adjacency of the related features. In addition, the parameters of features carry the design intent or engineering semantics related to the features. The parameters can be numerical or non-numerical information and must be extracted from the geometry model, e.g. a bead's height and its type (straight or curved). It should be noted that some features have different types (e.g. flanges can be curved or planar and closed, open or perpendicular) and the types of each feature are considered as one of their parameters.

Parent-child relationships has been implemented to rationalize the relation of features in sheet metal parts (Z. Liu et al., 2004; Sunil & Pande, 2008). There is a *parent-child* relationship between the features of ASM parts. Child features are created based on their parent feature. A child feature could be the parent feature of another child feature. For example, a flange and a hole that are made on a web are the child features of the web, whilst the flange per se can be the parent feature of another hole on itself.

3.4.3 ASM feature taxonomy

The features in ASM parts included in this study were observed by studying design guidelines and 168 diverse structural sheet metal parts of fuselage and cockpit. A structural system is comprised of a thin-skinned shell which is stiffened by longitudinal stringers supported by transverse frames to form a semi-monocoque structure (Niu, 1999). This semi-monocoque is very efficient and has a high strength to weight ratio. The parts that we have studied for this paper were all produced by brake-forming or hydro-forming, thus the skin panel parts and

stringers were omitted. Brake-formed and hydro-formed parts are used to form frames, bulkheads, passengers and cargo floor structures (Niu, 1999) and cockpit components.

An aerospace sheet metal part is manufactured in two main steps: trimming the blank of the part from a sheet of metal and deforming the blank where needed. The generic features of ASM parts are organized, in this paper, into *web*, *trim features* and *deformation features*. Figure 3.1 shows the proposed taxonomy of ASM model features.

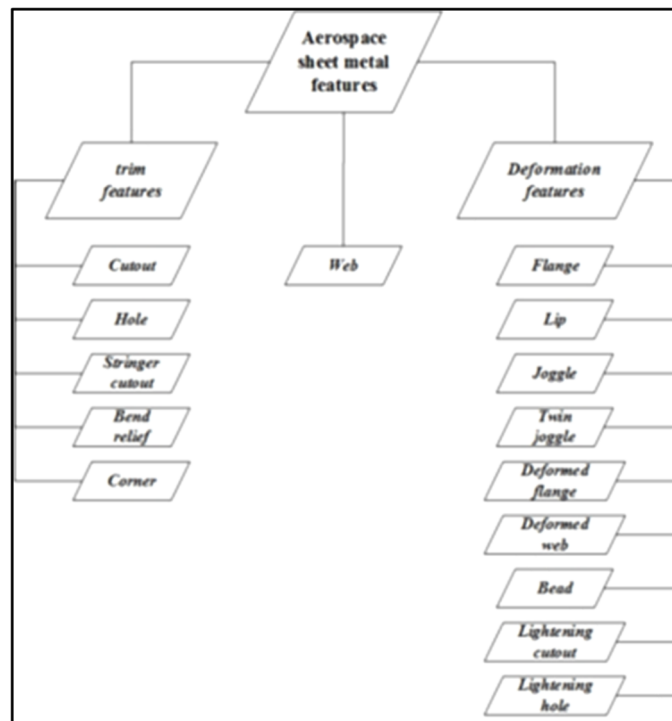


Figure 3.1 The proposed taxonomy of features in the ASM parts

The *web* is distinct among ASM features, shaped by cutting the blank and by the deforming operations that create other features on it. Figure 3.2 illustrates an example where a web boundary is defined by cutting the blank, indicated by the dashed-blue line, and the deforming operations creating the other features on it, indicated by the solid-red line. It should be noted that at a later step of the AFR process the web boundary is examined so as to yield other features such as corners and corner reliefs. A web is assumed to be the planar portion of an

ASM part with the highest surface area (Figure 3.2), and for 165 of our 168 samples (98%) the assumption is valid.

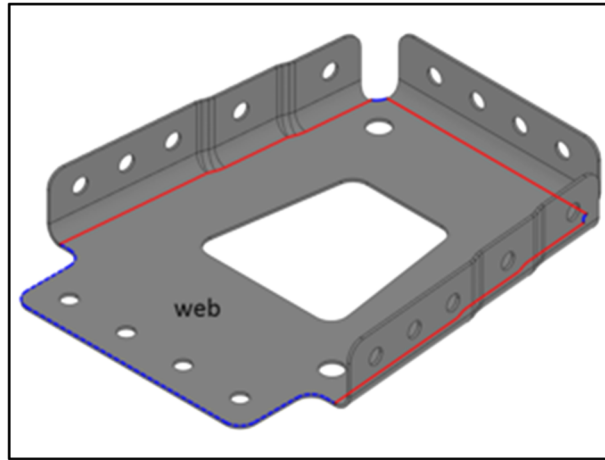


Figure 3.2 Illustration of the web as the feature with the highest surface area

The trim features include *cutout*, *hole*, *stringer cutout*, *bend relief* and *corner*, as illustrated in Figure 3.3, and are the result of trimming. *Cutouts* are formed by removing a portion of their parent feature, provided that the boundary of the parent feature is not changed. *Holes* are circular cutouts. *Stringer cutouts* are formed by modifying the boundary of the parent feature, i.e. the web, and splitting the flanges or the deformed flanges resulted from a twin joggle, as illustrated by Figure 3.3 (b) and (c). Stringer cutouts are created to make room for a stringer to pass through. The *bend reliefs* are cutouts to avoid sharp adjacency between flanges, which causes cracking. Given that bend reliefs are designed based on design guidelines, the length of the relief cuts is calculable. The *corners* are formed by rounding off sharp convex corners, often at the vicinity of holes; corners are concentric with their corresponding hole. Although both corner reliefs and corners could be considered as parts of their parent features' boundary, they convey design intentions, necessitating distinguishing them as features.

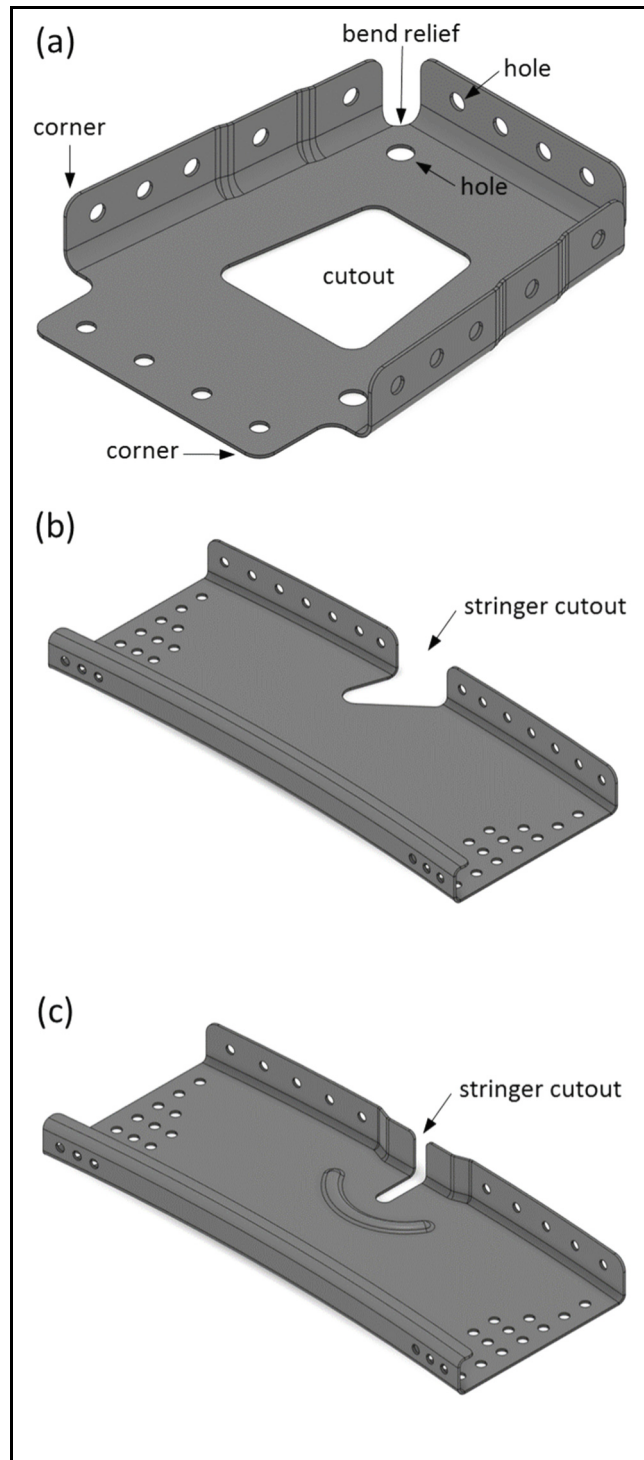


Figure 3.3 Illustration of trim features

The deformation features include *lightening cutout*, *lightening hole*, *flange*, *lip*, *joggle*, *twin joggle*, *deformed flange*, *deformed web* and *bead*, as illustrated in Figures 3.4, 3.5 and 3.6. They are created by deforming a portion of the parent feature. It is assumed in this study that all of the bends are created with a constant bend radius. The *lightening cutouts* and *lightening holes* are shaped by removing a portion of the parent feature and forming stiffening lips at the boundary of the removed portion, illustrated in Figure 3.4 (a) and (b). The *flange* is materialized on the external boundary of its parent feature and is always the child of a web or of another flange. It can be characterized as planar or curved; assembly or stiffening; immediate or return; single or combined; and perpendicular, open or closed, as illustrated in Figure 3.4 (c) to (f) and Figure 3.5 (g) and (h). Flange types are defined according to the usual practice, except single and combined types of flanges which will help understanding the proposed feature recognition method.

Assembly flanges and stiffening flanges differ due to their functionality, which is reflected by the presence or absence of holes for joining them to other parts. The immediate flanges are materialized on the web, and the return flanges are materialized on their immediate parent flange. A single flange is a flange that is modeled resting on a unique supporting geometry. On the other hand, a combined flange is materialized from distinct portions resting on distinct supporting geometries that combine to form it. Perpendicular, open or closed flanges are determined based on the angle between the flange and its parent feature.

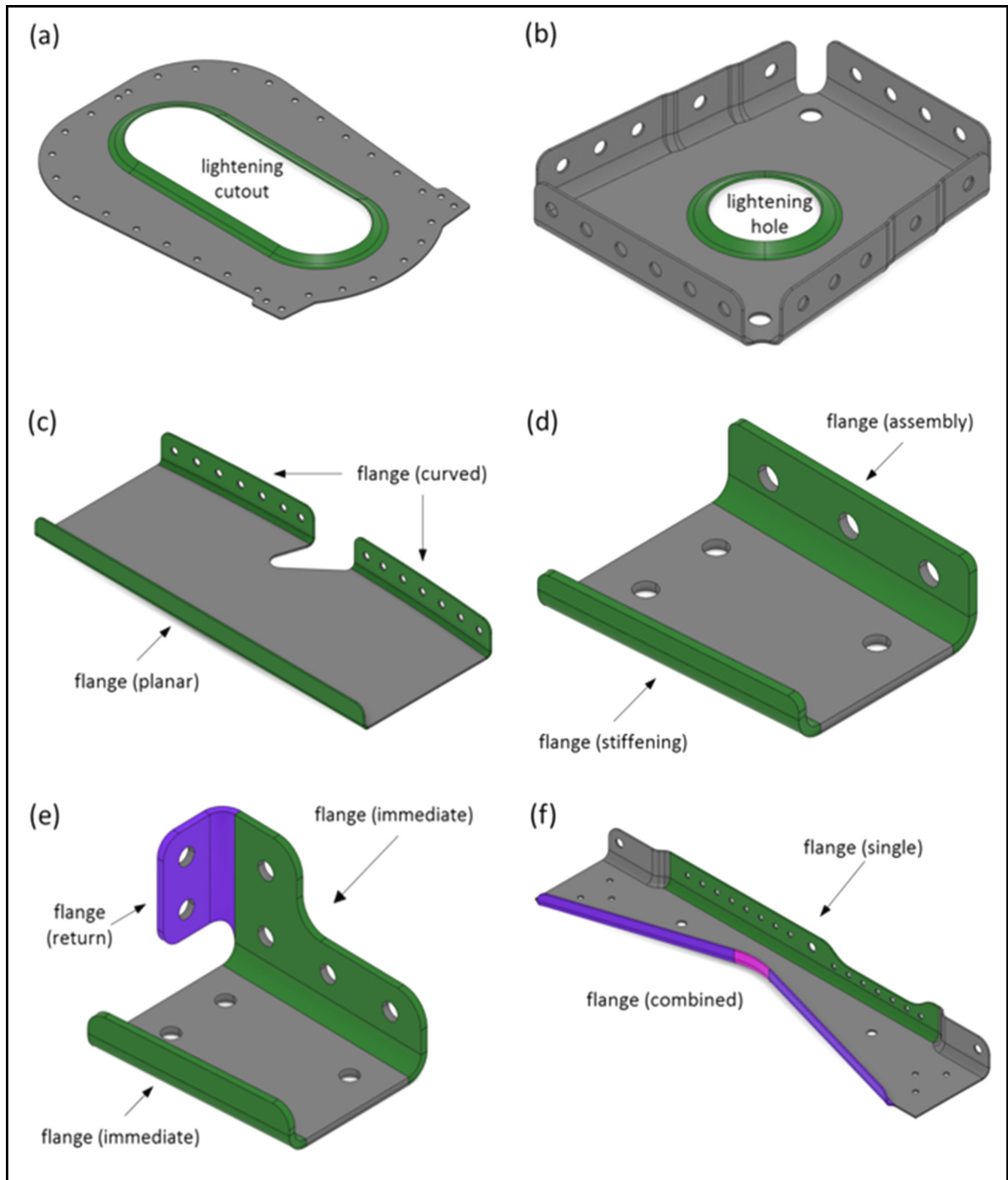


Figure 3.4 Illustration of deformation features (a, b, c, d, e, f)
 (a) a lightening cutout and (b) a lightening hole; (c) a planar and a curved flange; (d) an assembly and a stiffening flange; (e) an immediate and a return flange; (f) a single and a combined flange

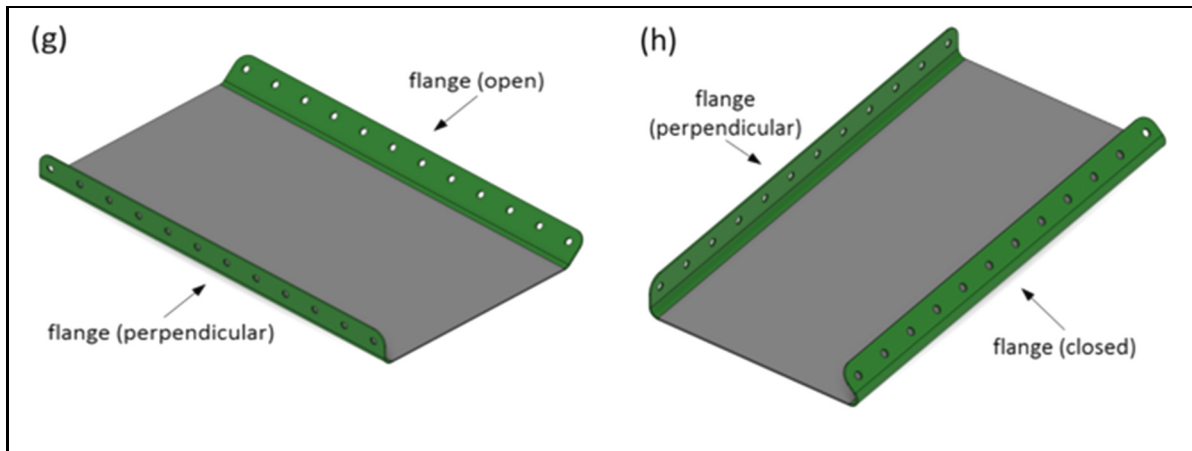


Figure 3.5 Illustration of deformation features (g, h)

(g) a perpendicular and an open flange; and (h) a perpendicular and a closed flange

Lips are shape-wise similar to combined-open-immediate-stiffening flanges, as illustrated in Figure 3.6 (a). *Joggles* and *twin joggles* are step-deformations that produce recesses on the web or on the flanges. The recessed portions of the parent feature's themselves are considered as features that are called either a *deformed web* or a *deformed flange* depending on the parent feature of the joggle. Figure 3.6 (a) and (b) provide examples of joggles and deformed flanges, and twin joggles and a deformed web, respectively. *Beads* result from protruding a portion of the web, and can be characterized as straight or curved as illustrated in Figure 3.6 (c) and (d).

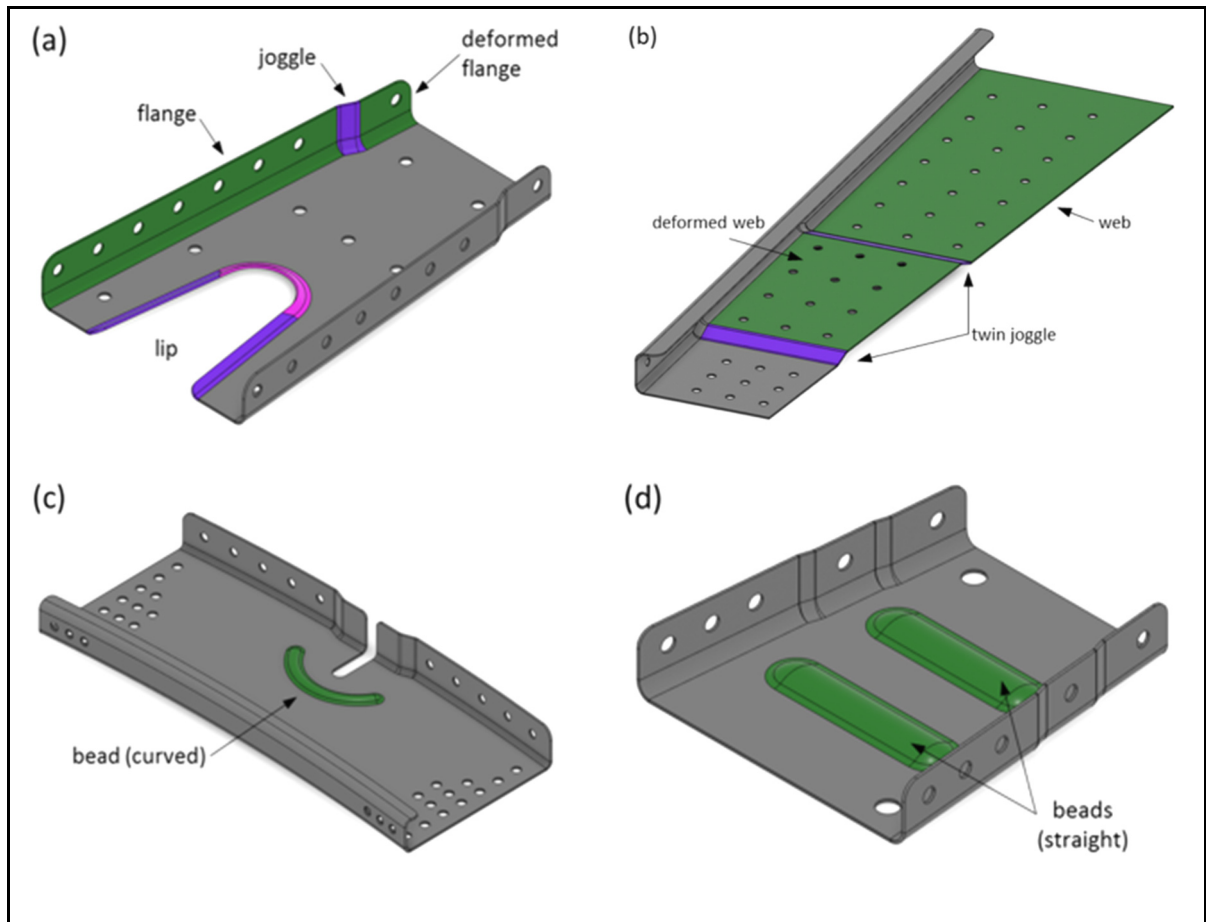


Figure 3.6 Illustration of features
 (a) a lip, a joggle and a deformed flange; (b) a twin joggle and a deformed web; and (c) and (d) a straight bead and a curved bead

3.5 Proposed automated feature recognition method

The aerospace sheet metal (ASM) automated feature recognition (AFR) method described in this work is comprised of two major steps: (1) classifying and grouping the elements of 3D B-rep model of the ASM parts and (2) recognizing the ASM features. Figure 3.7 provides a flowchart of the proposed method.

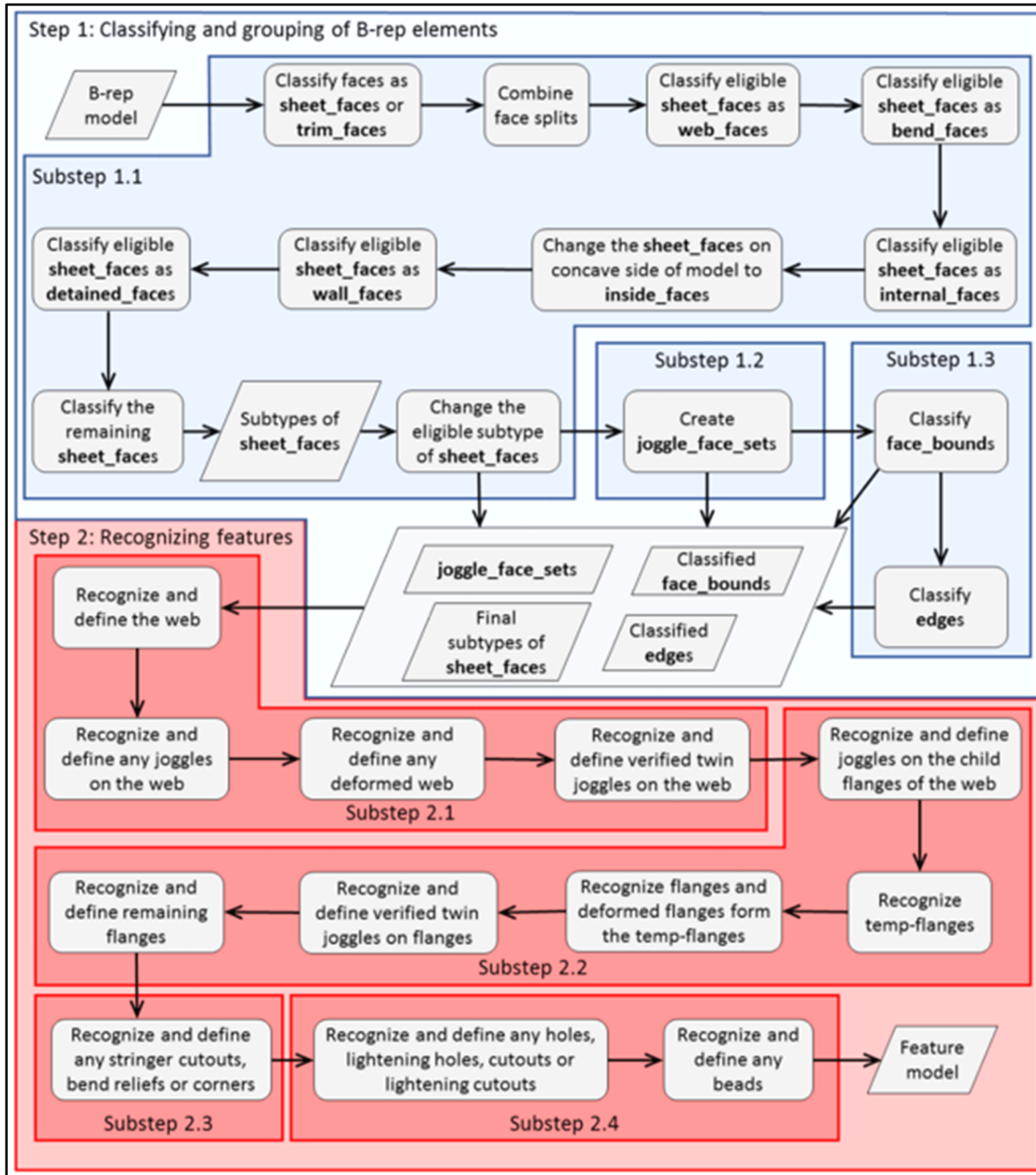


Figure 3.7 Flowchart of the proposed method

3.5.1 Classifying and grouping of B-rep elements

To prepare the 3D B-rep models for feature recognition, specific B-rep topological elements must be classified into novel subtypes and eligible faces are grouped into novel sets that will

be used in feature recognition process. The classification and grouping of **faces**, **face_bounds** and **edges** is explained in the following section. After classifying **faces** (substep 1.1), they are grouped into **joggle_face_sets** (substep 1.2), which are sets of faces relating to joggles. At the end, **face_bounds** and **edges** are classified into their subtypes (substep 1.3). The output of this first major step thereafter becomes the input of the second major step, feature recognition.

Substep 1.1: Classifying faces

Classifying faces includes a series of classification actions on faces according to their continuity at their shared edges and their adjacency, and a series of faces class alteration actions if certain conditions are met.

First, every **face** from the B-rep model is classified as either **sheet_faces** or **trim_faces**. Figure 3.8 (a) shows an example to illustrate the distinction between **sheet_faces** (gray) and **trim_faces** (red). The **trim_faces** (red) are related to the faces of an ASM part that are created by trimming, and one of their dimensions is always equal to the part thickness. The **sheet_faces** (gray) are related to the faces of an ASM part that are not being trimmed, and they constitute the two sides of an ASM part.

A **sheet_face** is G1 continuous to all its adjacent **sheet_faces**. Any face that is not G1 continuous with a **sheet_face** is a **trim_face**. Two internally continuous C1 faces are geometrically continuous (G1) if their tangent planes coincide at every point along their shared edge (Watkins, 1988). Since the B-rep model is created by a CAD modelling solution, as opposed to being constructed from 3D scanning, it can be assumed that all of the faces are internally continuous C1, and if the two faces are geometrically continuous at any arbitrary point of their shared edge, they have G1 continuity along the entirety of their shared edge.

Classifying faces as sheet_faces or trim_faces:

In a very first step, the two largest faces of the B-rep model are classified as **sheet_faces**. All the adjacent faces of these faces are subsequently evaluated to identify if they are in G1 continuity with these **sheet_faces**. The G1 continuous ones are classified as **sheet_faces**. The

adjacent faces without G1 continuity are classified as **trim_faces**. This G1 continuity evaluation continues until all faces in the B-rep model are classified as either **sheet_faces** or **trim_faces**.

Combining face splits:

Before the next step, the **sheet_faces** must be checked for any face splitting. All of the **sheet_faces** must be evaluated to identify the adjacent ones that are associated to the same **surface**. These **sheet_faces** must be combined by removing one of their shared edges. Figure 3.8 (b) shows how the combination of split faces affects the faces of the lightening hole.

Classifying eligible sheet_faces as web_faces:

The **sheet_faces** are primarily classified as **web_faces**, **bend_faces**, **wall_faces**, **detained_faces** and **internal_faces**. The **sheet_faces** with the two highest surface area are classified as **web_faces**. **Web_faces** are a pair of planar **sheet_faces** that have the highest surface area. Figure 3.8 (c) shows the **web_faces** of the example in Figure 3.8 (a). The classification of the remaining **sheet_faces** to the other subtypes is performed by starting from the **web_faces** and traversing to **trim_faces** until all **sheet_faces** are classified appropriately.

Classifying eligible sheet_faces as bend_faces:

All the **sheet_faces** adjacent to the **web_faces**, via their **face_outer_bounds**, are then classified as **bend_faces**, as illustrated in yellow in Figure 3.8 (d).

Classifying eligible sheet_faces as internal_faces:

The **sheet_faces** adjacent to **web_faces**, via their non-**face_outer_bounds**, are classified as **internal_faces**. The **sheet_faces** adjacent to **internal_faces**, and that are not yet classified, are also classified as **internal_faces**, as illustrated in pink in Figure 3.8 (e).

Changing the sheet_faces on concave side of model to inside_faces:

At this stage, the areas of each of the preceding **web_faces** and their adjacent **bend_faces** and **internal_faces** are calculated. The **web_face** and its adjacent **bend_faces** and **internal_faces**

that have the smallest combined area are changed – or reclassified – to **inside_faces**. The **inside_faces**, as illustrated in black in Figure 3.8 (f), correspond with the side of the ASM part that is concave. **Inside_faces** are not used in the rest of the process, while the remaining **sheet_faces**, on the convex side, are used in the following steps.

Classifying eligible sheet_faces as wall_faces:

All the **sheet_faces** adjacent to the preceding **bend_faces** are checked to verify that they are only adjacent to one **bend_face**. If so, they are classified as **wall_faces**. The **wall_faces** are the **sheet_faces** that are adjacent to only one previously classified **bend_face**, as illustrated in Figure 3.9 (g). **Wall_faces** must not be assumed to be linked to flanges, as they could be involved in forming the geometry of various features.

Classifying eligible sheet_faces as detained_faces:

If there are any **sheet_faces** adjacent to more than one preceding **bend_faces**, they are classified as **detained_faces**, as indicated in Figure 3.9 (h). **Detained_faces** are the **sheet_faces** that share edges with more than one previously classified **bend_face**.

Classifying the remaining sheet_faces:

At this stage, all the **sheet_faces** that are not classified as their subtypes and are adjacent to **wall_faces** via their **face_outer_bounds** are classified as **bend_faces**. The **sheet_faces** adjacent via the **wall_faces'** non-**face_outer_bounds** are classified as **internal_faces**. The remaining **sheet_faces** are evaluated according to the rules for **wall_face**, **detained_face**, **bend_face**, and **internal_face** to be classified consecutively until no **sheet_face** remains (Figure 3.9 (i)). In general, **bend_faces** are the **sheet_faces** that are adjacent to previously classified **wall_faces**, at the **face_outer_bound** of the **wall_faces** and **internal_faces** are the **sheet_faces** that are adjacent to **web_faces** or **wall_faces** at their non-**face_outer_bounds**.

Changing the subtype of eligible sheet_faces:

The **wall_faces**, **bend_faces** and **detained_faces** will be subsequently evaluated to verify if they are eligible to be changed. So far, these **sheet_faces** have been classified according to

their adjacency. Now, if eligible, their classification is changed so that they appear in the appropriate sequence. The sequence that **bend_faces**, **connect_faces** or **wall_faces** eventually appear in needs to comply with two rules: **bend_faces** appear between **wall_faces**; and **connect_faces** appear between **bend_faces**. All the **detained_faces** will eventually be changed as either **connect_faces**, **bend_faces**, or **wall_faces**.

First, the **bend_faces** that are adjacent to other **bend_faces** are checked to identify those that are adjacent to only one other **bend_face**. If the adjacent **bend_face** is adjacent to more than one **bend_face**, it is changed to a **connect_face**. Since a **connect_face** actually ‘connects’ **bend_faces**, it is designated so, as illustrated by the **connect_faces** shown in green in Figure 3.9 (j). The change of the eligible **bend_faces** to **connect_faces** is continued until there are no **bend_faces** adjacent to another **bend_face**.

The **detained_faces** are then evaluated and the ones that are surrounded by preceding **bend_faces** are changed to **connect_faces**. The **connect_faces** are basically the **detained_faces** that are surrounded by **bend_faces** or the **bend_faces** that are adjacent to more than one **bend_face**. If there are any remaining **detained_faces**, the ones that are adjacent to preceding **connect_faces** are changed to **bend_faces** and the ones adjacent to preceding **bend_faces** are changed to **wall_faces**.

Finally, the **wall_faces** that are adjacent to other **wall_faces** are checked to identify the ones that are adjacent to only one other **wall_face**. If the adjacent **wall_face** is adjacent to more than one **wall_face**, it is changed to a **bend_face**. The change of the eligible **wall_faces** is continued until no **wall_faces** are adjacent to another **wall_face**. Figure 3.9 (j) illustrates the end result of the change of eligible **bend_faces** and **detained_faces** to **connect_faces**, and change of eligible **wall_faces** to **bend_faces**.

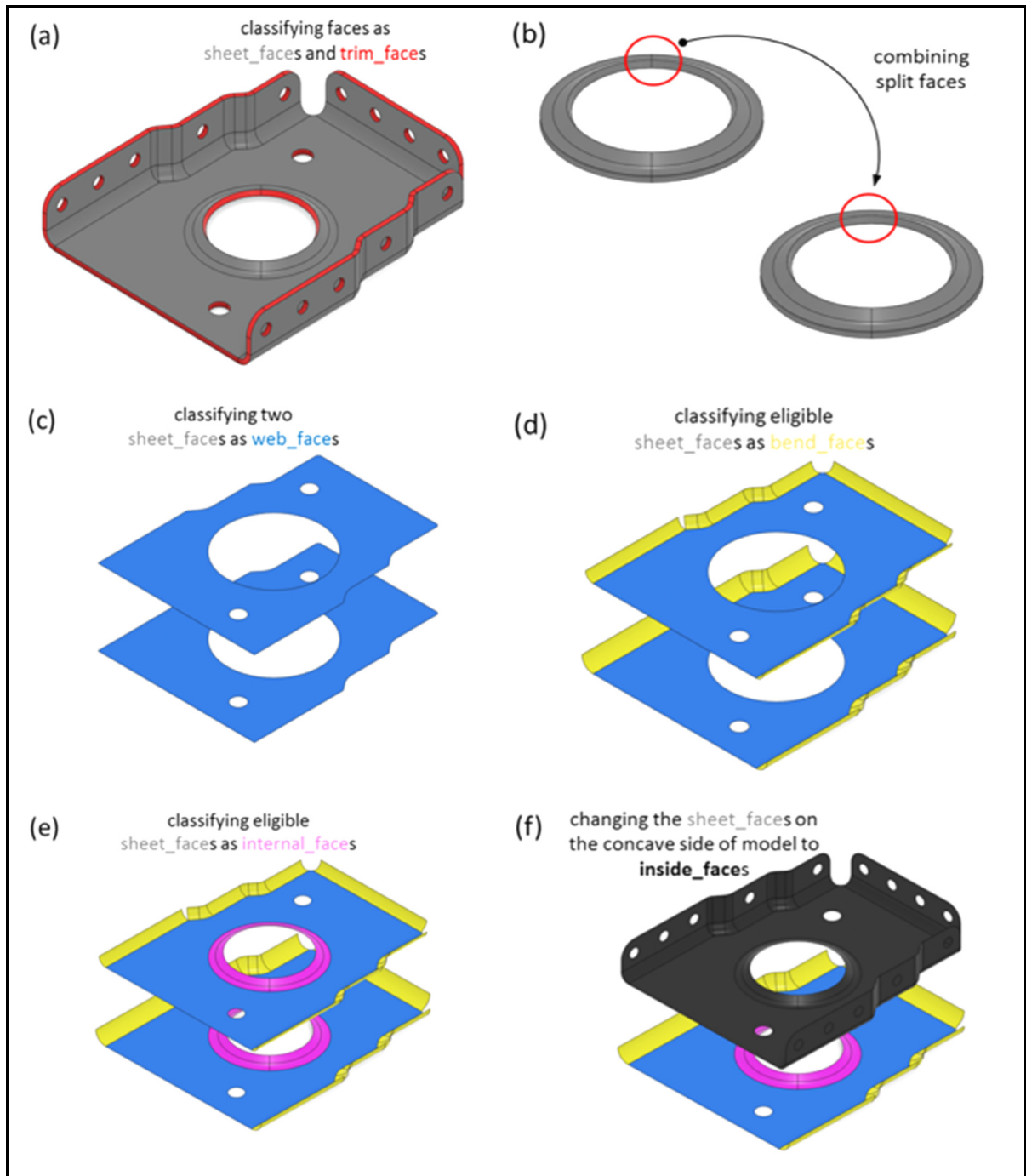


Figure 3.8 Illustration of the process of classifying faces (a, b, c, d, e, f)

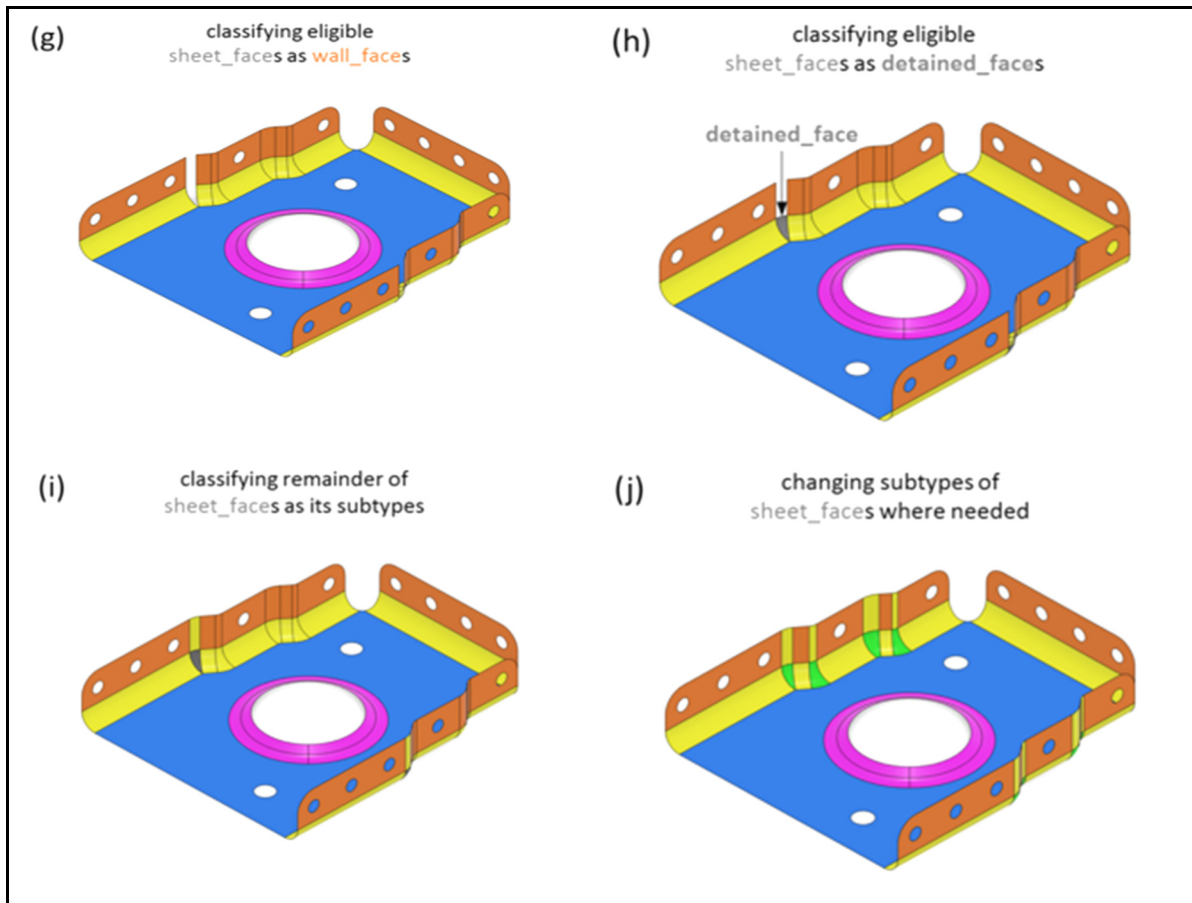


Figure 3.9 Illustration of the process of classifying faces (g, h, i, j)

Substep 1.2: Creating `joggle_face_sets`

Now that **faces** are classified, eligible ones are grouped to create **joggle_face_sets**. Indeed, the presence of certain sets of two **connect_faces** is an indicator of the presence of a joggle feature on a web or flange. Observing different scenarios of joggles on webs or flanges reveals that it takes **connect_faces**, **bend_faces** and a **wall_face** to materialize a joggle. Therefore, it is proposed to create **joggle_face_sets** to be used in describing joggles. A **joggle_face_set** consists of two **connect_faces**, three **bend_faces** and one **wall_face**.

To create a **joggle_face_set**, a **bend_face** that is adjacent to only one **connect_face** is identified first, as indicated in Figure 3.10 (a). The **connect_face** adjacent to this **bend_face** is evaluated to verify if one of its adjacent **bend_faces** are adjacent to another **connect_face**. If

yes, the two identified **connect_faces** are included in the **joggle_face_set** as well as the **bend_face** between them. This **bend_face** is evaluated to identify its adjacent **wall_face** that is not adjacent to any of the aforementioned **connect_faces**. The **wall_face** is also included in the **joggle_face_set** in addition to all of its adjacent **bend_faces**. The **wall_face** and the **bend_faces** are indicated in Figure 3.10 (b).

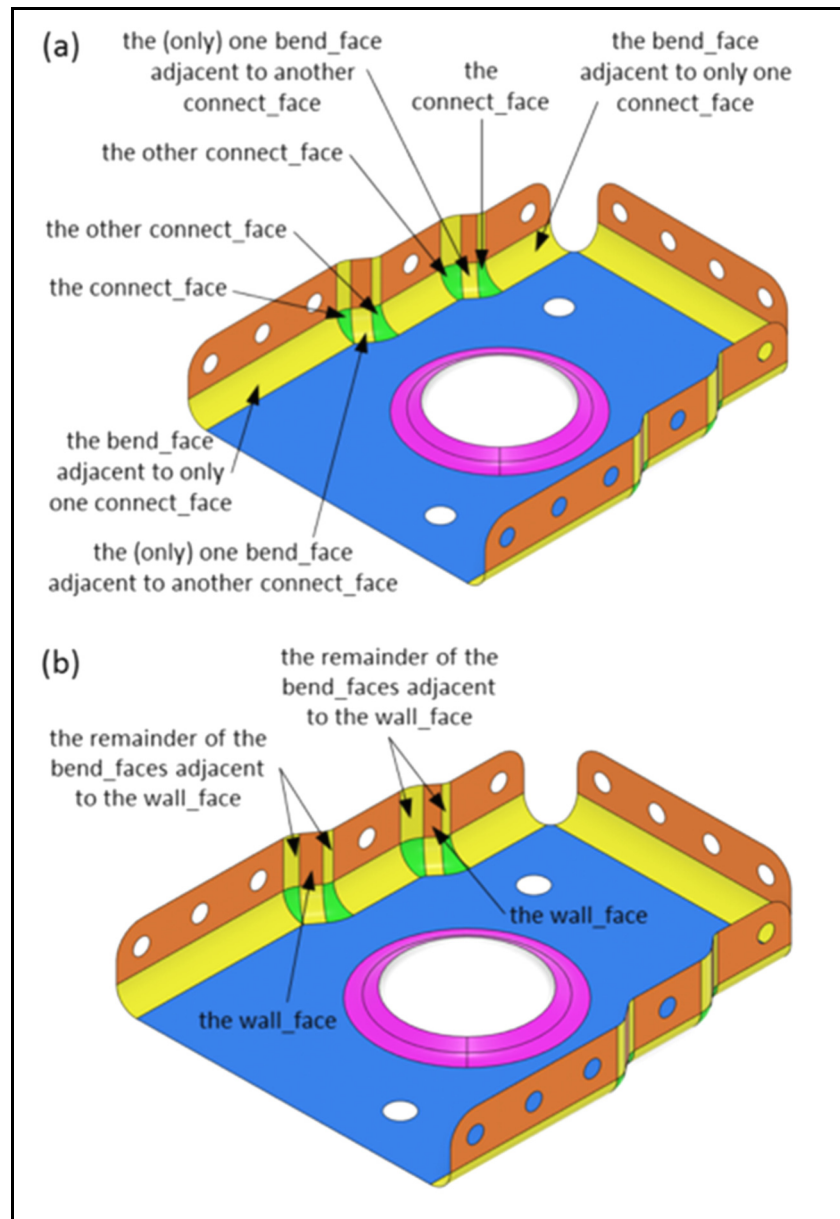


Figure 3.10 Illustration of the faces included in creating **joggle_face_sets**

Substep 1.3: Classifying **face_bounds** and **edges**

Now that **faces** are classified and that **joggle_face_sets** are created, we classify **face_bounds** and **edges**.

Classifying face_bounds:

Similar to what was done with **faces**, the **face_bounds** are now classified into novel subtypes. The **face_outer_bound** is the only subtype of the **face_bound** in STEP B-rep models (*Industrial automation systems and integration -- Product data representation and exchange - Part 42: Integrated generic resource: Geometric and topological representation - STEP 10303-42*, 2003). In our work, **face_bounds** are classified as **face_outer_bound**, **bead_bound**, **internal_bound** or **hole_bound**, as illustrated in Figure 3.11. If a **face_bound** is not a **face_outer_bound**, it is evaluated to verify if it surrounds seven **internal_faces**. In such a case, the **face_bound** is classified as a **bead_bound**. If a **face_bound** is neither a **face_outer_bound** nor a **bead_bound**, and it consists of **edges** associated to distinguishable **curves** (*Industrial automation systems and integration -- Product data representation and exchange -- Part 42: Integrated generic resource: Geometric and topological representation - STEP 10303-42*, 2003), it is classified as an **internal_bound**. If a **face_bound** consists of **edges** associated to undistinguishable **curves**, it is classified as a **hole_bound**. Distinguishable **curves** are the ones that are not defined by identical underlying geometric equations, whilst undistinguishable **curves** are the ones that are defined by the same equations. A **hole_bound** is indeed a special type of **internal_bound** that carries the additional semantics of defining an interior boundary related to a hole or a lightening hole. Thus, a **hole_bound** is comprised of **edges** that are associated to undistinguishable **curves**.

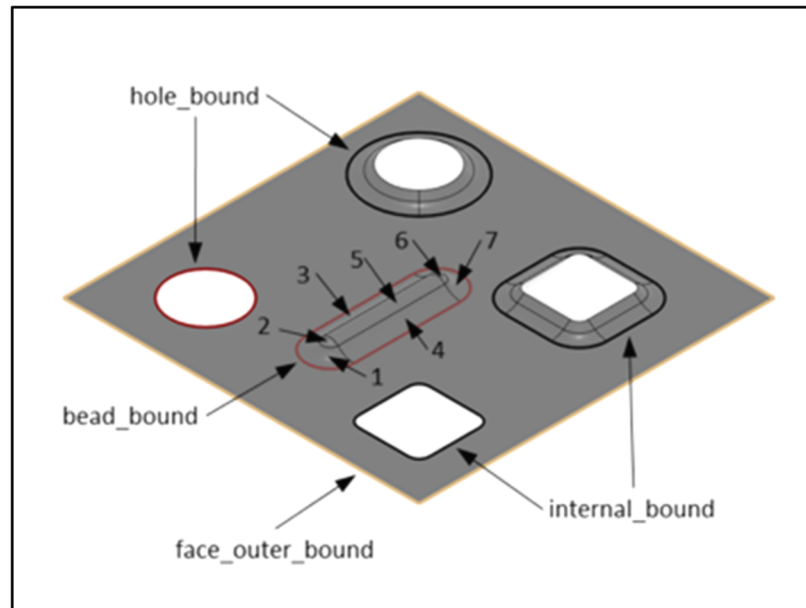


Figure 3.11 A fabricated example to illustrate subtypes of **face_bounds**

Classifying edges:

The **edges** are classified as novel subtypes based on the types of faces they are associated to and their geometry. They are classified as **trim_lin_edges**, **trim_nonlin_edges**, **untrim_lin_edges** or **untrim_nonlin_edges**, as illustrated in Figure 3.12. A **trim_lin_edge** is an edge that is shared between any subtype of **sheet_faces** and a **trim_face**, and is associated to a **line**. A **trim_nonlin_edge** is an edge that is shared between any subtype of **sheet_faces** and a **trim_face**, and is associated to any subtype of **curve** other than a **line**. An **untrim_lin_edge** is an edge that is not shared with any **trim_face** and is associated to a **line**. An **untrim_nonlin_edge** is an edge that is not shared with any **trim_face** and is associated to any subtype of **curve** other than a **line**. The classification of the edges is pivotal to recognizing features.

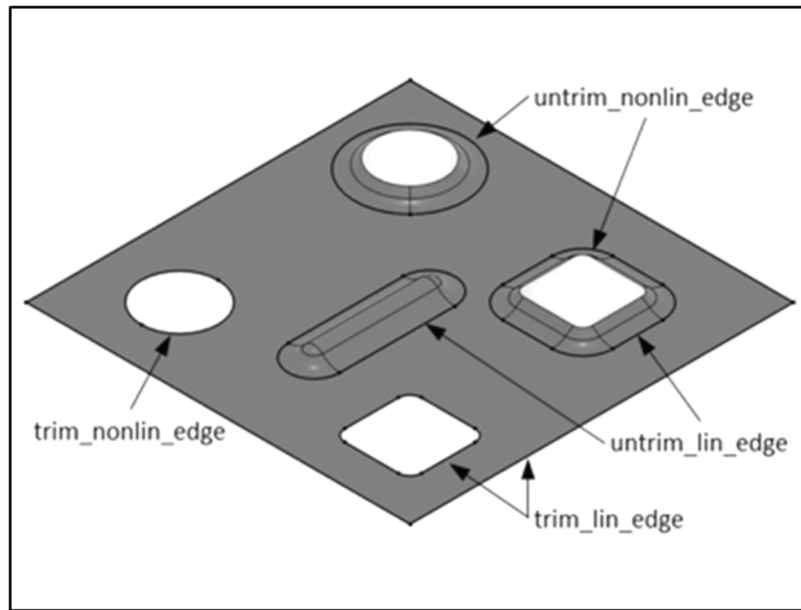


Figure 3.12 A fabricated example to illustrate subtypes of **edges**

3.5.2 Recognizing features

In previous section, a method for classifying and grouping **faces**, **face_bounds** and **edges** was proposed, as well as a method for grouping eligible faces to create **joggle_face_sets**. In this section, the method to use these entities for recognizing and constructing features and for extracting meaningful design information is proposed. Here, we explain the correlation of the geometric, topological and feature entities. ASM feature recognition involves identifying the topologic entities that comply with the conditions required for recognizing the corresponding feature. The parameters of features are calculated from the geometric and topological elements and the feature relationships are formed by the topological elements according to the features' descriptions.

Before describing the steps of feature recognition, it should be noted that some of the ASM features could have either parent or child relationships with other features, while some features could only have either a parent relationship or a child relationship with other features. For instance, a flange is always child of its parent feature and is itself the parent feature of its child

features. On the other hand, features such as holes, lightening holes, cutouts, lightening cutouts, beads, corner and bend reliefs could only be the children of their parent feature and could never be the parent of any other feature.

The web is a feature that exists in all ASM parts, and all other features of the ASM part are directly or indirectly its child features. The web does not have a parent feature, and thus it should be recognized in the first step. This step is also where it should be determined if the web is deformed by joggles and if so, that the deformed web is also recognized (section 4.2.1). Next, the existing flanges on the web, the joggles and the twin joggles on those flanges and the deformed flanges are recognized and fully characterized (section 4.2.2). In section 4.2.3., it is explained how the preceding flanges, deformed flanges and joggles are evaluated to recognize any existing bend reliefs, stringer cutouts or corners. Finally, in section 4.2.4., all the holes, lightening holes, cutouts, lightening cutouts and beads on the web and on the deformed web as well as the holes, cutouts and corners of preceding flanges and deformed flanges are recognized. Although a flange could possibly have child lightening holes, lightening cutouts and beads, such a case was not found in the industrial samples of this study and it is not considered in this work.

Substep 2.1: Recognizing webs, their child joggles and twin joggles, and deformed webs

The web is proposed to be recognized first. To recognize the web, identifying the **web_face** is sufficient, as it represents the geometry of the web. The feature relationships between the web and its child features are formed by the **bead_bounds** and **internal_bounds** and the **hole_bounds** of the **web_face**, and the edges that are shared between the **face_outer_bound** of the **web_face** and the adjacent **bend_faces** of the child features. The only parameter of the web is its supporting plane, which is the plane that the **web_face** occurs in.

After recognizing the web, to avoid mistakenly recognizing a joggle on the web as a flange, the presence of a joggle on the web is verified first. If the **web_face** shares an edge with a **bend_face** of a **joggle_face_set** but does not share an edge with any of the **connect_faces** of the **joggle_face_set**, the web is deformed by a joggle. If any of the **connect_faces** of the

joggle_face_set share an edge with the **web_face**, the corresponding joggle is not pertinent to the web; it does not deform the web but rather deforms a child flange of the web. Figure 3.13 (a) and (b) illustrate a fabricated example in which the presence of a shared edge between the **web_face** and the **connect_faces** of a **joggle_face_set** distinguishes the joggle on a flange from the joggle on the web. Thus, to recognize a joggle on the web, the **web_face** is checked to verify whether it shares an edge with a **bend_face**, but not with any of the **connect_faces** of the **joggle_face_set**. The **joggle_face_set**, including its **bend_faces**, **connect_faces** and **wall_face**, form the geometry of the joggle. The feature relationships of a joggle-on-the-web to its parent web and to its child deformed web are formed by the shared edge between each corresponding **bend_face** of the **joggle_face_set** and the **web_face** of the web and **wall_face** related to the deformed web. Those shared edges are colored red in Figure 3.13 (b).

The parameters of joggles, including joggle vectors indicating depth and runout directions, the radius of the bends, the depth and the runout length, are extracted from the geometry of the joggle, as shown in Figure 3.13 (c). It should be noted that, in the specific example in Figure 3.13 (c), the bend radii differ by 3 mm (the thickness of the part), so that in fact both bends on the actual part are defined as equal to the minimum bend radius value (6 mm) for the considered material and part thickness. In practice, these parameters, except depth vector, are obtained from a manufacturing standard and a standard number could represent them.

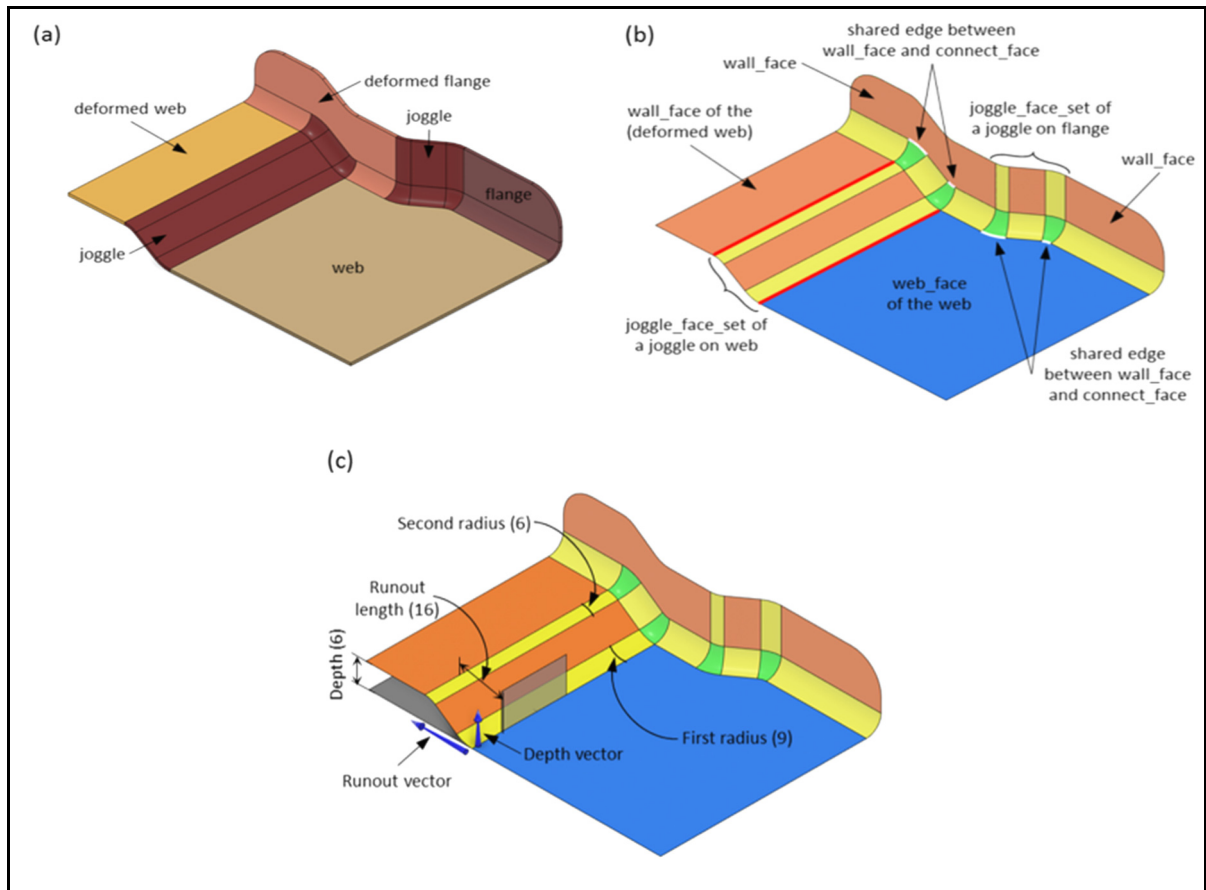


Figure 3.13 A fabricated example of how the presence of a shared edge between the **web_face** and the **connect_faces** of a **joggle_face_set** distinguishes a joggle on a flange from a joggle on the web (a) and (b); and the parameters of joggles (c)

Every joggle has one child feature: the deformed web or the deformed flange, if the joggle is on a flange. To recognize a deformed web, the **wall_face** that shares an edge with a **bend_face** of the preceding **joggle_face_set** but does not share an edge with any of the **connect_faces** of the **joggle_face_set** is identified, as shown in Figure 3.13 (b). The **wall_face** forms the geometry of the deformed web. The feature relationship between a deformed web and its parent joggle is formed by the shared edge between the **wall_face** and the **bend_face** of a **joggle_face_set**. Its feature relationship with its child features is formed by the **bead_bounds** and **internal_bounds** and **hole_bounds** of its **wall_face** and the edges that are shared between the **face_outer_bound** of the **wall_face** and the adjacent **bend_faces** of the child features. The parameters of a deformed web are its supporting plane, which is the plane that the **wall_face** occurs in, if we look at it from a geometric perspective. From a more semantic perspective, the

deformed web is characterized by its distance to the web, which is equal to the depth of its parent joggle.

If the deformed web is itself deformed by a joggle or has a child joggle, the parent and child joggles of the deformed web are evaluated to recognize twin joggles, as illustrated by Figure 3.14 (a). Merging two joggles into one twin joggle means that the parent and child features of the two original joggles must also be merged. The deformed webs are evaluated to identify the one to be merged into the web. The geometry of the newly merged features is comprised of the combined geometry of both original features. For example, when the deformed web is being merged into the web, its **wall_face** is added to the geometry of the web. Figure 3.14 shows an example where two joggles (in Figure 3.14 (b)) are merged into a twin joggle and a deformed web merged into the web. Figure 3.14 (c) indicates how the **wall_face** of the deformed web that merges into the web is consequently associated to the web, or how the two **joggle_face_sets** associated to two individual joggles are associated to the one twin joggle.

The feature relationship of a twin joggle to its parent web is formed by the two shared edges between the **bend_faces** of a **joggle_face_set** and the **web_face** and **wall_face** of the web, as colored red in Figure 3.14 (c). The feature relationship of a twin joggle to its child deformed web is formed by the two shared edges between the **bend_faces** of the **joggle_face_sets** and the **wall_face** of the deformed web, as colored blue in Figure 3.14 (c). The parameters of the twin joggle are the same as those of a joggle, except that the runout vector is not included. In the industrial samples reviewed in this work, webs were only deformed by one joggle, two joggles on their two ends, or a twin joggle.

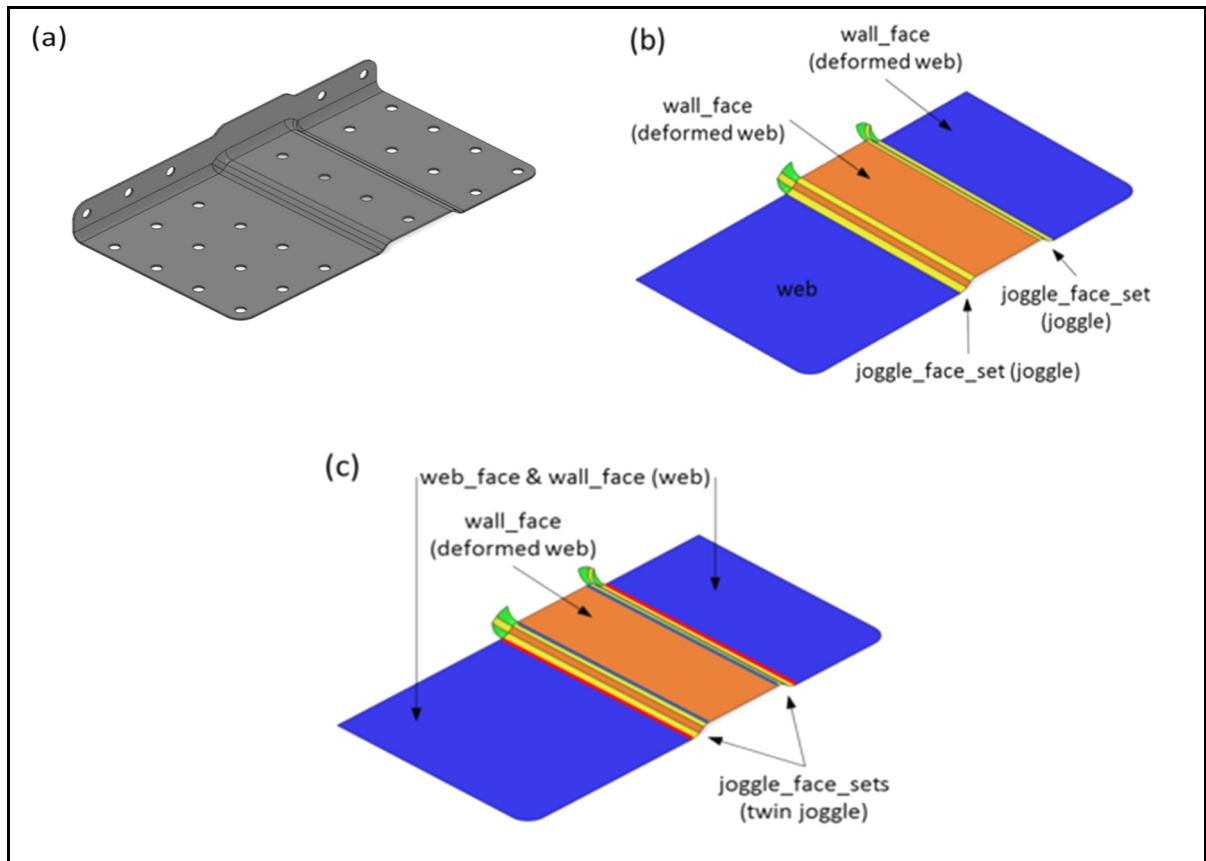


Figure 3.14 An example of two joggles merged into a twin joggle and a deformed web merged into the web

Substep 2.2: Recognizing flanges, their child joggles and twin joggles, and deformed flanges

Once the web and its child joggles, twin joggles and deformed webs are recognized, the flanges and their potential child joggles, twin joggles and deformed flanges are recognized in a next step. The joggles on the flanges are proposed to be recognized before recognizing the flanges. To recognize the joggles on the child flanges of the web, the faces included in the web geometry are checked to verify if they share edges with a **bend_face** or with any of the **connect_faces** of any **joggle_face_set**. The **joggle_face_set**, including its **bend_faces**, **connect_faces** and **wall_face**, forms the joggle geometry. The parameters of a joggle on a flange are extracted from its geometry, as described for a joggle on the web. The feature relationships between the joggle and its parent flange and its child deformed flange are formed

by the shared edge between each corresponding **bend_face** of the **joggle_face_set**, and the **wall_face** of the flange and the deformed flange. However, the feature relationships and parameters of a joggle-on-the-flange is dependent on determining its parent and child features, which are the flange and the deformed flange.

To recognize a flange and a deformed flange, the **bend_faces** of the **joggle_face_set** are checked to identify the two **wall_faces** that share an edge with any of them but that does not share an edge with any of the **connect_faces** of the **joggle_face_set**, as indicated in Figure 3.15. Each of these **wall_faces** share an edge with a **bend_face** that is not included in the **joggle_face_set** but does share an edge with the web geometry (**web_face**), as indicated in Figure 3.15. Each of the **wall_faces** and its adjacent **bend_face**, which is not included in the **joggle_face_set**, form a temporary feature entity, a temp-flange, until the flange and deformed flange are distinguished. The definition of the joggle, on a flange, provides the basis for distinguishing a flange from a deformed flange; since a joggle on a flange always creates a recess.

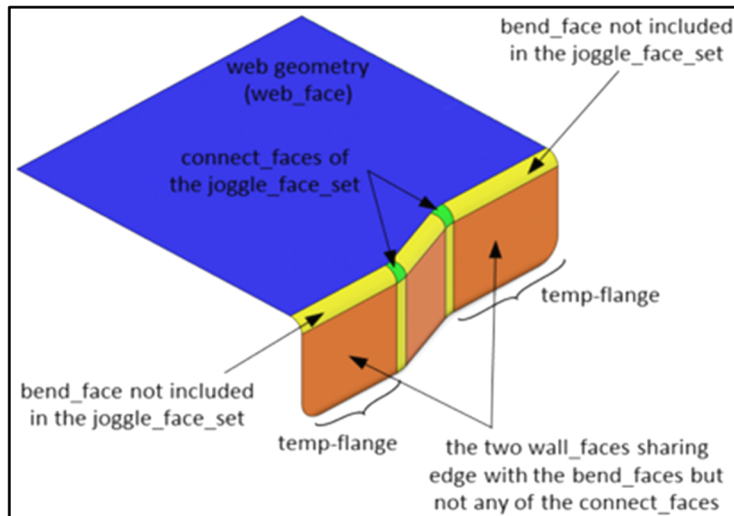


Figure 3.15 Illustration of the geometry of temp-flange

If the cross product of the normal vector of a **wall_face** of the temp-flange with the normal vector of the **wall_face** of the joggle is in the same direction as the normal vector of the

web_face, the order in which the parent feature of the joggle, the joggle and the child feature of the joggle share edges with the **web_face** is counterclockwise. If they are in the opposite direction, the order is clockwise. Figure 3.16 (a) and (b) give two examples of how the order of these features can be determined. A temp-flange is converted to a deformed flange if it is a child feature of any joggle, or it is converted to a flange if it is only the parent feature to a joggle and is not itself a child feature of another joggle.

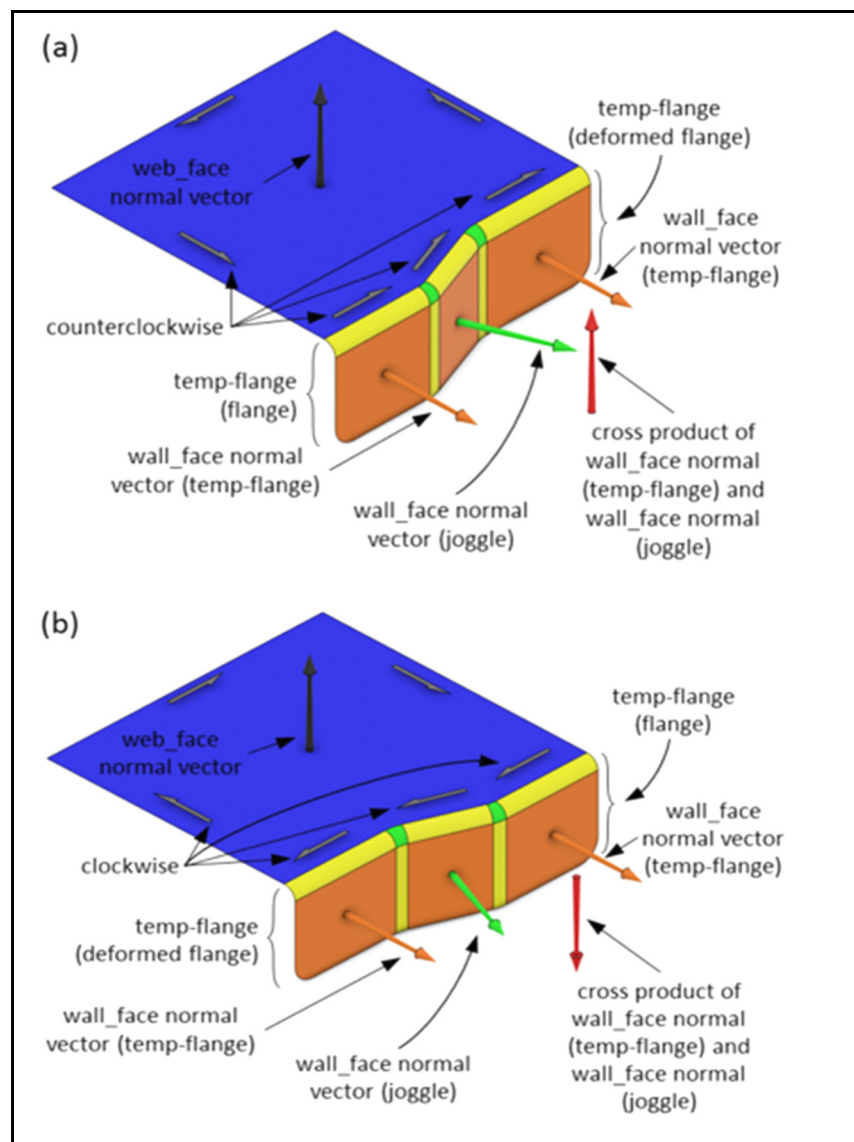


Figure 3.16 Two examples of determining the order in which the parent feature of the joggle, the joggle and the child feature of the joggle occur, to distinguish flange and deformed flange

If a deformed flange is the child feature to two joggles, the parent joggles are evaluated for recognizing twin joggles. Similar to joggles on the web, if the joggles have identical parameters, but their runout vectors are in opposite directions, they are merged into one feature, the twin joggle. Merging two joggles into a twin joggle means that the two parent flanges of the two original joggles must be merged into one parent flange. The geometry of the newly merged features is composed of the combined geometry of both original features. For example, one of the flanges is removed and its geometry (**bend_face** and **wall_face**) are added to the geometry of the other flange. The feature relationship between a twin joggle and its parent flange is formed by the shared edges between the **bend_faces** of a **joggle_face_set** and the **wall_faces** of the parent flange. The feature relationship of a twin joggle to its child deformed flange is formed by the shared edges between the **bend_faces** of the **joggle_face_sets** and the **wall_face** of the deformed flange. The parameters of a twin joggle on a flange are the same as those of twin joggles on the web and are extracted from its geometry.

When all the flanges that are deformed by joggles are recognized, the other flanges are searched for. To recognize them, the **web_face** and **wall_faces** that are associated with the web and the flanges are checked to identify their adjacent **bend_faces** that is not included in the geometry of the previously-recognized features. Each **bend_face** that is found is checked to identify its adjacent **wall_face** that shares an edge with it and is also not included in the geometry of the previously recognized features. The **bend_face** and its adjacent **wall_face** form the geometry of a flange. In case the **bend_face** only shares an edge with a **wall_face**, that is part of the geometry of a previously-recognized flange, and a **connect_face**, that is not included in a **joggle_face_set**, the flanges are merged into one combined flange. The **bend_face** and the **connect_face** as well as the **bend_faces** and **wall_faces** of the adjacent flanges form the geometry of the flange. Figure 3.17 (a) shows an example for this case.

Once all flanges and deformed flanges are recognized, flanges are further characterized. The feature relationship between a flange and its parent feature is formed by the shared edge between the **bend_face** of the flange and the **web_face** or **wall_face** of the parent feature. The

feature relationship between the flange and its child features is formed by the **bead_bounds** and the **internal_bounds** as well as the **hole_bounds** of the **wall_face** or the edges that are shared between the **face_outer_bound** of the **wall_face** and the adjacent **bend_faces** of the child features.

The parameters of flanges, including their supporting geometry, bend radius, length and type are extracted from the geometry of the flange, as shown in Figure 3.17 (b). The planar and curved flanges are distinguished by evaluating their supporting geometry to verify if it is planar or non-planar. The assembly and stiffening flanges are distinguished by checking their **wall_faces** to verify if it has **hole_bound**. The immediate and return flanges are distinguished by checking their **bend_faces** to verify if they are adjacent to another **wall_face** or the **web_face**. The single and combined flanges are validated by checking their geometry for the presence of a **connect_face**. The perpendicular, open and closed flanges are distinguished by checking the angle between the normal vector of their **wall_face** and the supporting geometry of their parent feature.

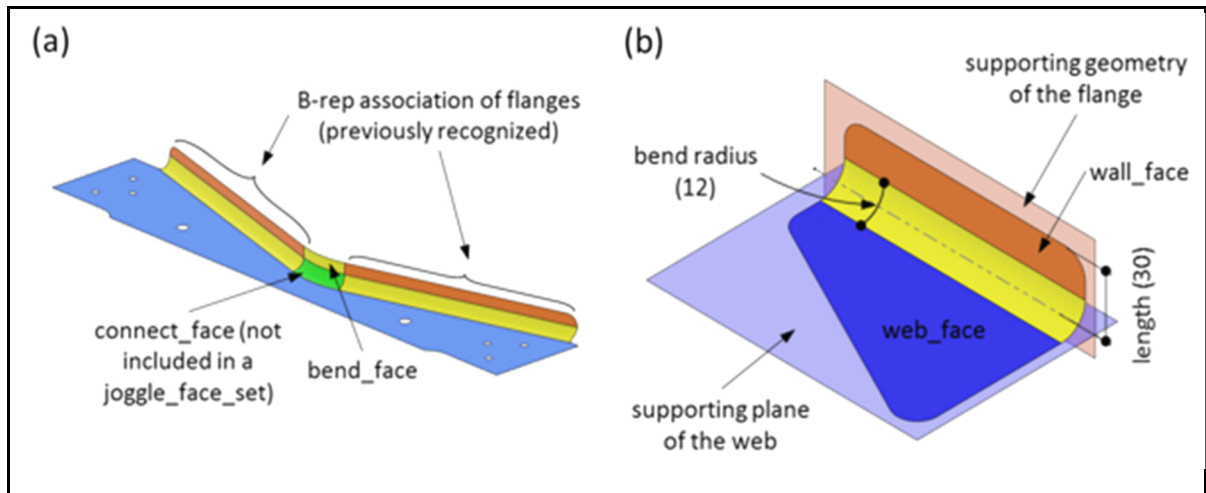


Figure 3.17 Illustration of recognizing and characterizing (a) a combined flange and (b) the parameters of a flange

Now that flanges are characterized, deformed flanges are further characterized next. The feature relationship between a deformed flange and its parent feature is formed by the shared

edge between its **wall_face** and the **bend_face** of the **joggle_face_set** of its parent joggle. Its feature relationships with its child features is formed by the **hole_bounds** of its **wall_face** and the edge that is shared between the **face_outer_bound** of the **wall_face** and the adjacent **bend_face** of the child feature. The parameters of the deformed flange are its supporting geometry, which is the plane or surface that the **wall_face** occurs in, and its distance to its related flange, which is equal to the depth of its parent joggle.

Substep 2.3: Recognizing stringer cutouts, bend reliefs and corners

After all the flanges and deformed flanges have been recognized and characterized, they are evaluated to identify some of their child features: stringer cutouts, bend reliefs and corners. To recognize the stringer cutouts, the deformed flanges that have identical supporting geometry, or the flanges that have identical supporting geometry, are identified, as indicated in Figure 3.18 (a) and (b). These flanges and deformed flanges are then evaluated to verify if there are **trim_nonlin_edges** and **trim_lin_edges** between them. Figure 3.18 (a) and (b) show two examples of stringer cutouts in which the **trim_nonlin_edges** and **trim_lin_edges** are indicated. If that is the case then the two flanges are merged into one and the edges form the geometry of a stringer cutout. When a stringer cutout is between two deformed flanges, they are merged into one deformed flange and their corresponding joggles and parent flanges are merged into a twin joggle and one flange. The feature relationship of the stringer cutout to its parent web is formed by the **face_outer_bound** of the **web_face**. The parameter of the stringer cutout is its profile, which is formed by its edges.

To recognize a bend relief, the deformed flanges and flanges are evaluated to verify if their supporting geometry is not identical and they are apart by one **trim_nonlin_edge**, or a series of **trim_nonlin_edge** and **trim_lin_edges**, in the **face_outer_bound** of the **web_face** or the **wall_face** of their parent web. If the edges are G1 continuous with the **trim_nonlin_edges** of the **bend_faces** corresponding to the flanges and/or deformed flanges, it forms the geometry of a bend relief. Figure 3.18 (c) illustrates a fabricated example of a bend relief in which the **trim_nonlin_edges** are indicated. The feature relationship of a bend relief to its parent web or flange is formed by the **face_outer_bound** of the **web_face** or the **wall_face** that the

trim_nonlin_edge is a part of. The parameter of the bend relief is its radius, calculated from the **trim_nonlin_edge**.

To recognize corners, the **trim_nonlin_edges** included in the **face_outer_bound** of the **web_face** or **wall_faces** are evaluated to identify the ones that are G1 continuous to at least one **trim_lin_edge**. Figure 3.18 (d) illustrates examples of corners in which the **trim_nonlin_edges** and the **trim_lin_edges** are indicated. The **trim_nonlin_edge** forms the geometry of the corner, and the **face_outer_bound** of the **web_face** or of a **wall_face** form the feature relationship of the corner with its parent web or flange. The parameter of the corner is its radius which is calculated from the **trim_nonlin_edge**.

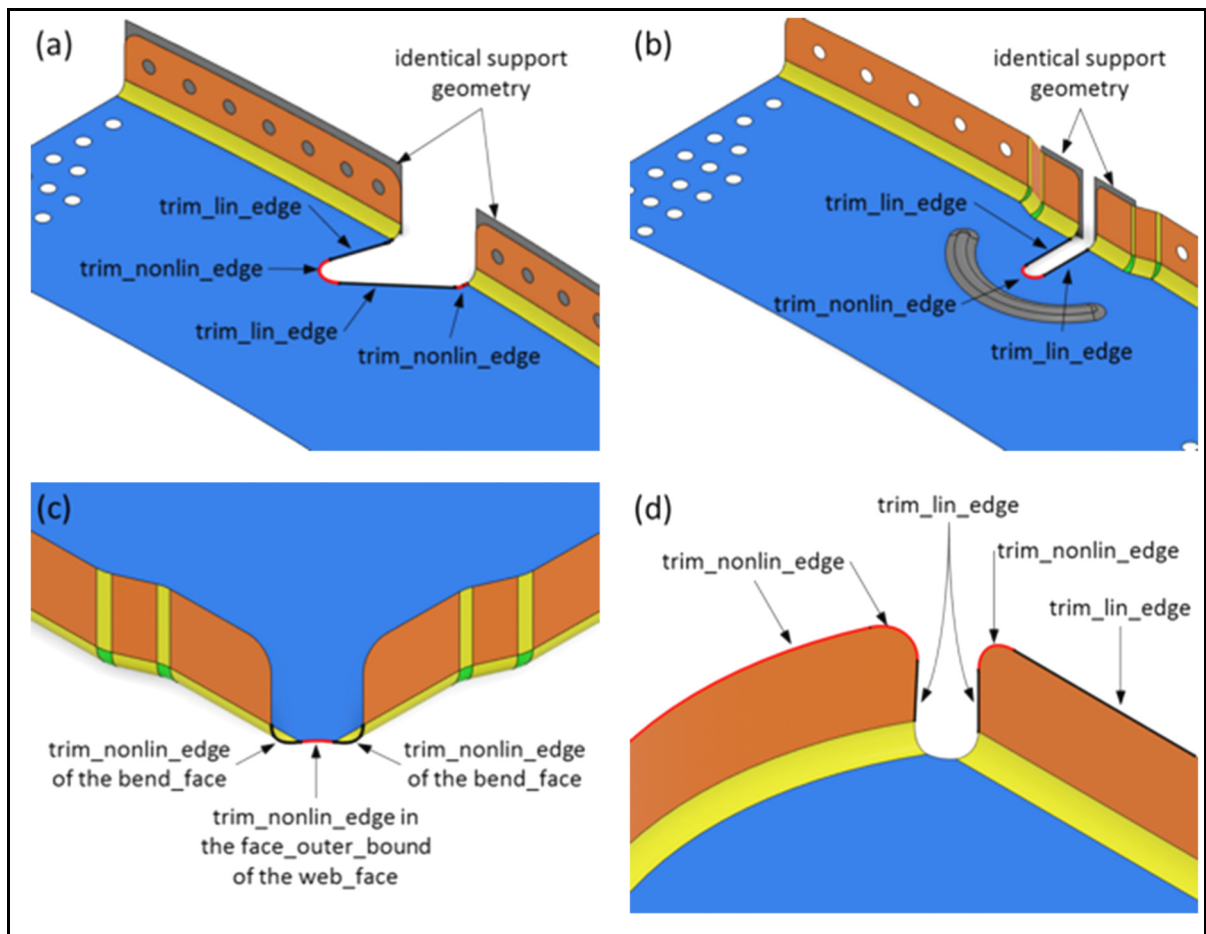


Figure 3.18 Analyzing edges to recognize stringer cutouts (a) and (b), bend reliefs (c) and corners (d)

Substep 2.4: Recognizing holes, lightening holes, cutouts, lightening cutouts and beads

In this last substep, all the holes, lightening holes, cutouts, lightening cutouts and beads on the web and the deformed web as well as the holes, cutouts and corners on the flanges and deformed flanges are recognized. To recognize these features, all of the **wall_faces** and **web_faces** are checked to identify their **non-face_outer_bounds**.

Finding a **hole_bound** on a **wall_face** or **web_face** that consists of **trim_nonlin_edges** verifies presence of a hole on the web or the feature that the **wall_face** is associated with. The **hole_bound** forms the geometry of the hole, and the **wall_face** or **web_face** form its feature relationship with its parent feature. The parameters of the hole are its location and its diameter; the latter is calculated from the edges included in the **hole_bound**. In practice, the holes diameter corresponds to a manufacturing standard and a standard note could represent. The location of the hole is represented by a point at the center of the **hole_bound**.

On the other hand, finding a **hole_bound** on a **wall_face** or **web_face** that consists of **untrim_nonlin_edges** verifies presence of a lightening hole. The **hole_bound** as well as the **internal_faces** that are surrounded by the **hole_bound** form the geometry of the lightening hole. The feature relationship of a lightening hole is the same as that of a hole. The parameters of a lightening hole include its location, diameter, bend radius and height. Such a combination of values would typically correspond to a manufacturing standard. These parameters are all calculated from the **internal_faces** in the geometry. Figure 3.19 illustrates the parameters of holes and lightening holes.

Cutouts and lightening cutouts are recognized in a procedure that is similar to how holes and lightening holes are recognized. Finding an **internal_bound** on a **wall_face** or a **web_face** that consists of **trim_nonlin_edges** and **trim_lin_edges** verifies the presence of a cutout on the web or the feature that the **wall_face** is associated with. On the other hand, if the **internal_bound** consists of **untrim_nonlin_edges** and **untrim_lin_edges**, it verifies the presence of a lightening cutout. The geometry of a cutout is formed by the **internal bound** and

the geometry of a lightening cutout is formed by the **internal_bound** as well as the **internal_faces** that are surrounded by it. The **wall_face** or **web_face** that the **internal_bound** is on forms the feature relationships of cutouts or lightening cutouts with their parent feature. The parameters of cutouts and lightening cutouts are their profile, which is formed by the **internal_bound**, bend radius and the height, which are calculated from the **internal_faces** included in its geometry. In practice, bend radius and height are designed according to a manufacturing standard (similar to lightening holes manufacturing standards) and a standard number could represent them. Figure 3.19 illustrates the geometrical parameters of a cutout and of a lightening cutout, but no references to manufacturing standards for confidentiality reasons.

To recognize a bead, the **web_face** or the **wall_face** associated to the web are evaluated to verify the presence of a **bead_bound**. The geometry of a bead is formed by the **bead_bound** and the **internal_faces** that are surrounded by it. The **wall_face** or **web_face** that the **bead_bound** is on forms the feature relationship of the bead with its parent feature. The parameters of a bead are its profile, which is formed by the **bead_bound**, its type (straight or curved) and its first and second radius (designed according to a manufacturing standard). An example is shown Figure 3.19.

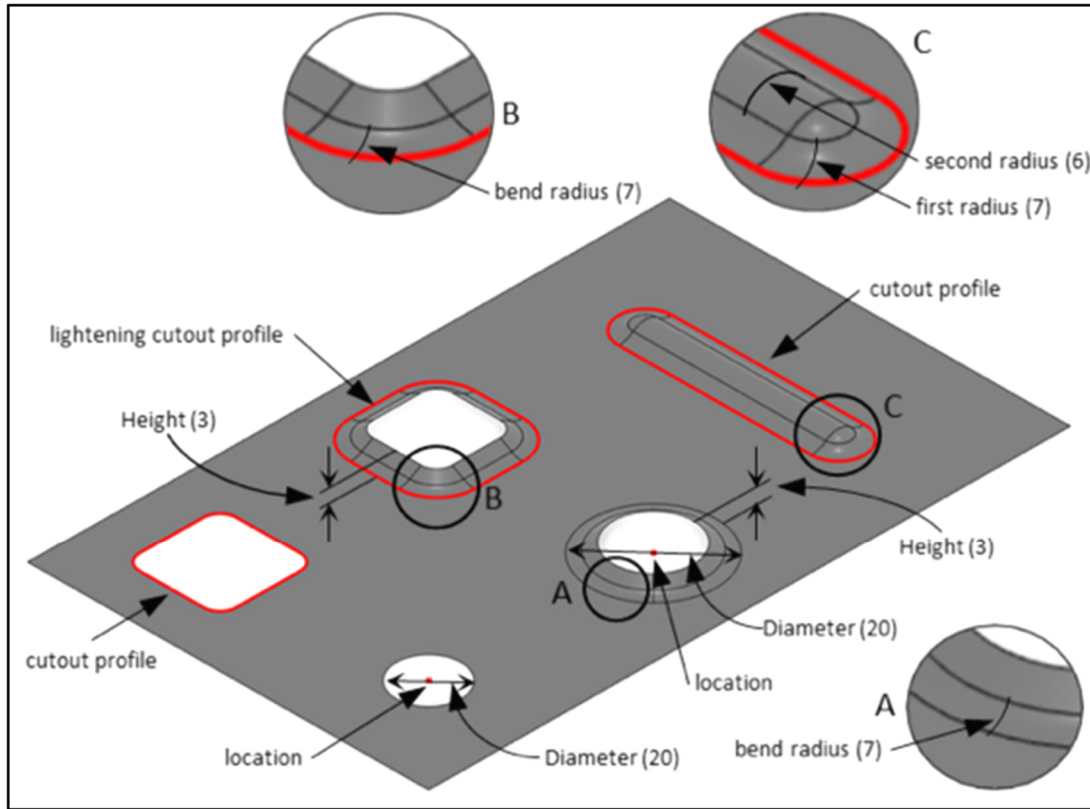


Figure 3.19 Parameters of a hole, a lightening hole, a cutout, a lightening cutout and a bead

3.6 An example

In this section, a real part is used to illustrate the proposed feature recognition process. The part is illustrated in Figure 3.20 (a), and the progression of its feature recognition process is shown in Figures 3.20 (b) to (d), 3.21 and 3.22. The 3D B-rep model of the part, illustrated in Figure 3.20 (b), is imported. The first major step of classifying and grouping the elements of 3D B-rep model is shown in Figure 3.20 (c) and (d), and Figure 3.21 (e), (f) and (g). First, its faces are classified as **sheet_faces** and **trim_faces**, shown in Figure 3.20 (c). Then, the **sheet_faces** (on the outside of the part) are classified as their appropriate subtypes, shown in Figure 3.20 (d). It should be noted that to remain concise the process of classifying **sheet_faces** to their subtypes is not illustrated in detail. Next, the faces are evaluated to identify the potentially existing **joggle_face_sets**, indicated by red circles in Figure 3.21 (e). The **face_bounds** and **edges** of all **sheet_face** subtypes are also classified as their subtypes. As an

example, the subtypes of **face_bounds** and **edges** of a **wall_face** are illustrated to be classified to their subtypes in Figure 3.21 (f) and (g).

The second major step of recognizing features is shown from Figure 3.21 (h), (i) and (j) and 3.22. First the web is recognized, shown in Figure 3.21 (h). Then, the joggles are recognized along with the temp-flanges, illustrated in Figure 3.21 (i). The temp-flanges are changed to flanges and deformed flanges, illustrated in Figure 3.21 (j). Next, the rest of the flanges (those without joggles) are recognized, shown in Figure 3.22 (k). Once all flanges and deformed flanges are recognized, their child bend reliefs and corners are also recognized. As an example, the bend reliefs and corners of two flanges are pointed out in Figure 3.22 (k). Eventually the holes are recognized, shown in Figure 3.22 (l).

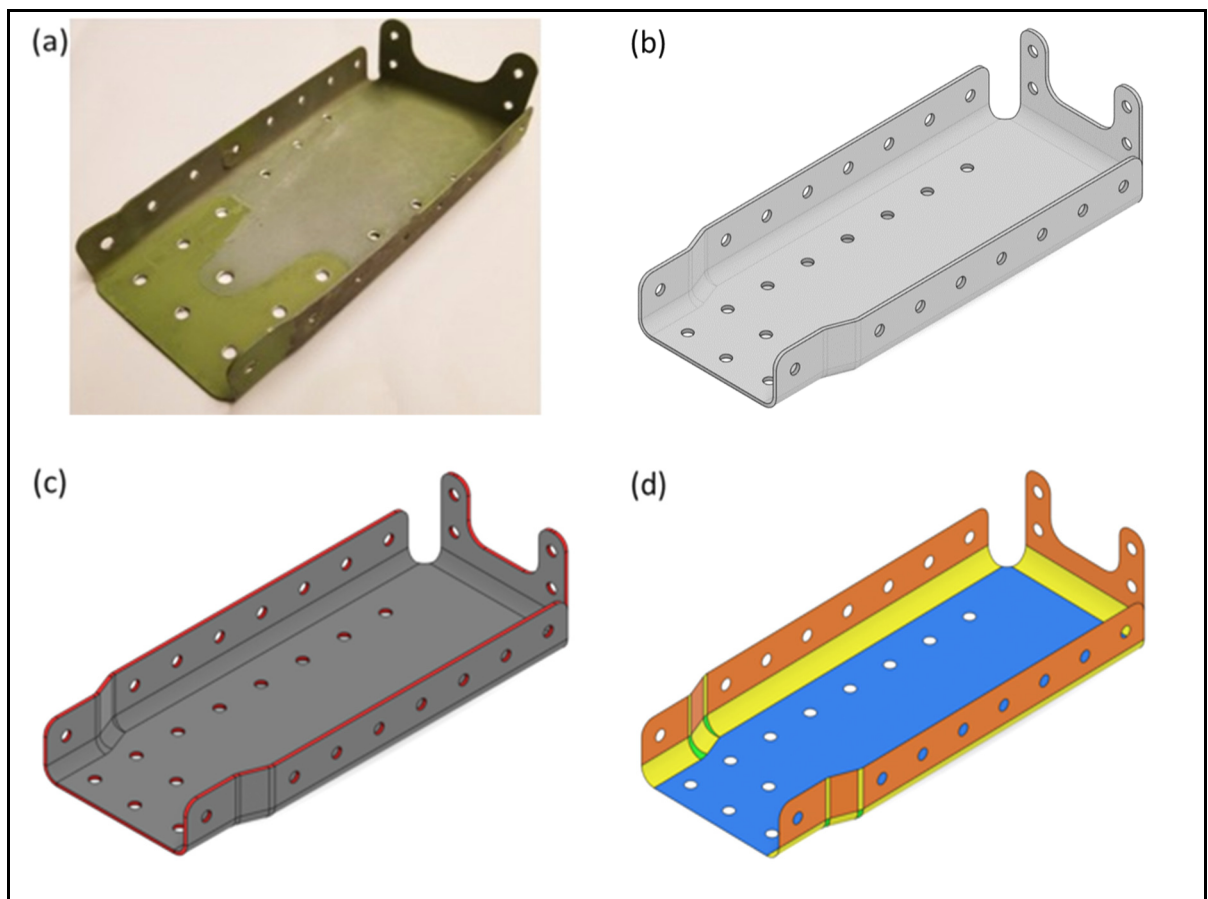


Figure 3.20 Illustration of feature recognition process for a real part model (a, b, c, d)

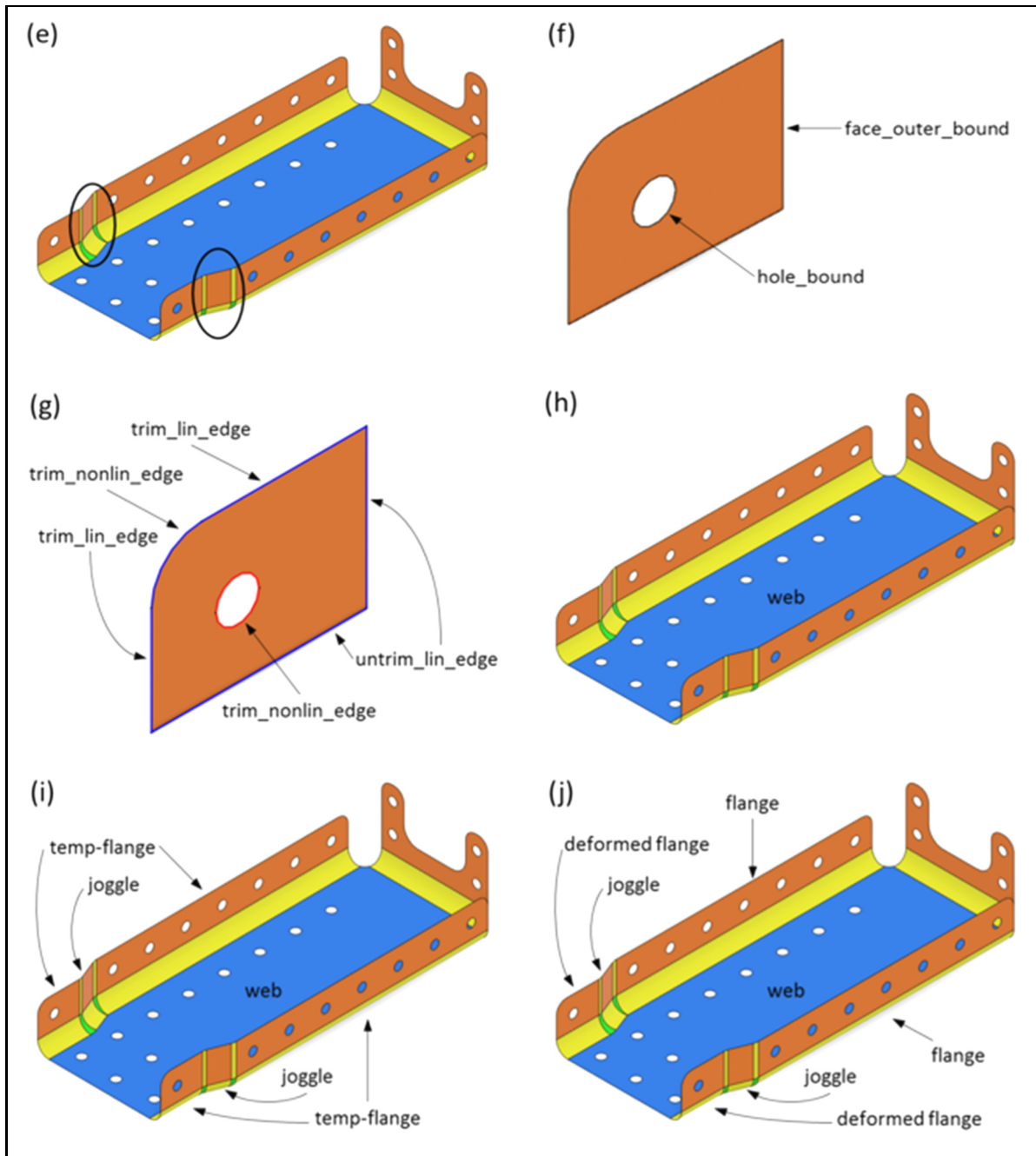


Figure 3.21 Illustration of feature recognition process for a real part model (e, f, g, h, i, j)

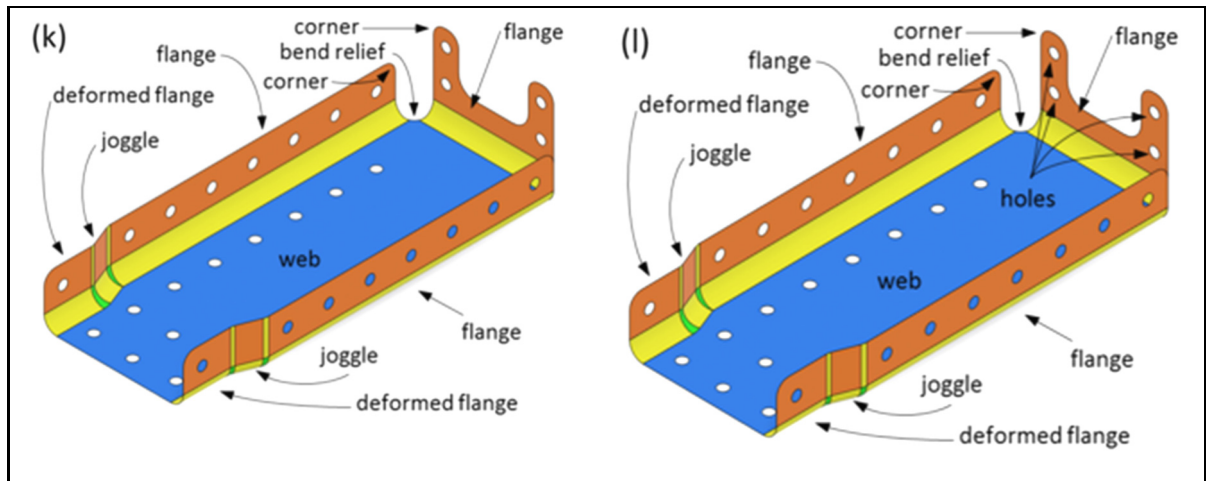


Figure 3.22 Illustration of feature recognition process for a real part model (k, l)

3.7 Conclusion

The main objective of this work is to propose a feature recognition method for structural aerospace sheet metal (ASM) part models, which has not been addressed by previous works. The proposed method is designed to cover all of the features observed from studying real-world structural ASM parts, models and designs, rather than generic sheet metal parts.

The nomenclature of the features and the terminology used in this work are based on the terminology used in the aerospace industry, rather than the terminology utilized in previous academic works. For example, a lightening hole and a lightening cutout are called a collar and an internal flange (Kannan & Shunmugam, 2009b), or an extruded hole (Sunil & Pande, 2008). To the best of the authors' knowledge, features like web, joggle, twin joggle, stringer cutout, corner and bend relief, that are used in the design of structural ASM parts, have not been included in the literature, contributing to the novelty and practicality of this paper. On the other hand, features like hem and curl (Kannan & Shunmugam, 2009a), lance and louver (Gupta & Gurumoorthy, 2013) are not observed in structural ASM parts and are therefore excluded from the scope of this paper.

A description of ASM features was provided in this work. The taxonomy of the features of structural ASM parts was presented, based on a detailed study of aerospace design guides and

168 actual parts. The studied sample parts were produced by brake-forming or hydro-forming. Skin panels were omitted since they belong to another class of parts. Based on the design guides and the studied samples, two assumptions were made:

1. A web is assumed to be the planar portion of an ASM part with the highest surface area.
2. All the bends are assumed to have constant bend radii.



The proposed automated feature recognition method consists of two major steps: 1) classifying and grouping the elements of 3D B-rep model, and 2) recognizing aerospace sheet metal features. Through step 1, relevant topological elements of B-rep models are categorized to their subtypes so as to enable step 2.

The two above-mentioned assumptions are satisfied by a great majority of structural ASM parts. Indeed, only 2% of the studied parts did not meet these premises. However, devising an automated feature recognition method able to include these 2% would be a serious challenge to be embarked upon by future work.

The proposed feature recognition process describes the features of a part's model through their geometry, feature relationships and parameters. From there, it becomes possible to describe any ASM part by its features, so as to increase the semantics level of the dialog with the CAD user. End users are thus envisioned to interact with the feature ASM models like they interact with native CAD models and their operations. Hence, combining the feature structure with the B-rep information could help build fully modifiable feature models. Such feature models could then easily be adapted for new designs. Our next research goal, however, is to compare similar structural ASM parts and express their differences in terms of meaningful features.

CHAPTER 4

FEATURE-BASED MODEL DIFFERENCE IDENTIFICATION FOR AEROSPACE SHEET METAL PARTS

Syedmorteza Ghaffarishahri¹  and Louis Rivest¹ 

¹ Department of Systems Engineering, École de Technologie Supérieure,
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3

Paper published in *Computer-Aided Design and Applications*, June 2021

4.1 Abstract

A semantic model difference identification (MDI) solution can be defined as an MDI solution that identifies and represents the differences between two compared models in terms of meaningful engineering information. Semantic representation of the differences between 3D CAD models is especially challenging due to the variety of modeling solutions used across industries, the non-uniqueness of modeling sequences and the use of low-level information, e.g. B-rep STEP files, in engineering communications. This work proposes an MDI method that represents the differences between 3D CAD models based on engineering semantics through features. Brake- and hydro-formed aerospace sheet metal parts are used as the application domain in which to propose and illustrate the proposed method. This method consists mainly of a pose registration stage and a difference identification stage. The pose registration method considers that all the features of a part serve specific functions, some of which are fundamental to the part's essential functionality, and that they are intertwined with the design intent of the part, which is particularly true for aerospace sheet metal parts. This provides the opportunity to semantically register feature-based 3D CAD models according to the unique purpose of the features in this specific domain of application. Difference identification is approached by primarily identifying and segregating the commonality between the compared 3D CAD models and then identifying the differences. The differences between 3D CAD models are classified as added, removed or differed features. Differed features are those features that are of the same type, but whose definition varies. As an outcome, the

proposed method describes a way to fully pose-register 3D CAD models and identify their differences semantically based solely on their features, and, by extension, their design intent.

4.2 Introduction

Information reuse is one of the most prominent strategies to improve efficiency throughout a product's life cycle, and it is applicable to all activities related to product development (Hicks, Culley, Allen, & Mullineux, 2002). Information reuse can basically be divided into retrieving useful information and identifying the useful part of retrieved information (J. Liu, Liu, Ni, & Zhou, 2018).

In a scenario in which a process planner benefits from reusing the process plans of parts similar to the current project's part, a 3D shape search solution enables finding the similar models. In addition, a Model Difference Identification (MDI) solution would identify and measure the differences between the models to facilitate reuse of their process plans according to their differences. In another case scenario, a model of a part is distributed, for example between design partners, and modified unilaterally. The MDI solution would identify and measure the difference between the versions of the model and help evaluate and approve the modifications. In these scenarios, it is important that the differences are identified and represented at an appropriate semantic level to support rationalizing their impact on their related downstream engineering tasks and to facilitate user interactions.

The above-mentioned scenarios reveal the importance of MDI in effective information reuse and engineering work. The ability of an MDI method to represent the differences in an engineering-wise informative way is key to facilitate information reuse. Although MDI solutions and methods exist, representing the differences in a semantic way has not been addressed. For example, where the difference between two models is the displacement of a hole, available solutions identify the difference as the addition of material at the original place of the hole and removal of material from the new place of the hole (Brière-Côté et al., 2013).

Semantic representation of the differences between part models is a difficult task due to the variety of modeling solutions used across industries, the non-uniqueness of modeling sequences and the use of low-level information, e.g. STEP files, in engineering communications. Various modeling solutions and different versions of each of those modeling solutions are used for the same application domain in different industries, although standardization of the data exchange of 3D CAD models has already been proposed (J. Kim, Pratt, Iyer, & Sriram, 2008).

Because of the non-uniqueness of modeling operations sequences, even if the same modeling solution is used to model the same part, models of the same part could be identified as being different. Even if the data structure differences between various modeling solutions and their versions, as well as the non-uniqueness of modeling operations sequences were not hindrances to conducting MDI on native models, identifying the differences between models based on modeling operations would not necessarily express engineering semantics.

Considering these drawbacks of an MDI solution based on native CAD models or B-rep models, it seems inevitable to take measures to elevate the level of information that is readily available from CAD models. For this purpose, automated feature recognition has been researched comprehensively to elevate the level of information of CAD models (Y. Shi et al., 2020). Feature models contain features that represent high-level engineering semantics. An MDI method based on feature models created by an automated feature recognition solution would avoid any non-uniqueness of the native CAD models resulting from solution variances and from modelling sequence variances.

This work proposes an MDI method that represents the differences based on engineering semantics through features. The relevant previous works are reviewed in the following section. Next, the proposed method is detailed in steps that describe how to perform pose registration and difference identification. CAD models of real-world parts are used to illustrate the

proposed method and its steps. Different aspects of difference identification are elaborated in order to paint a clear picture of the problem and the solution. Lastly, the method is illustrated by comparing real-world examples and identifying their differences.

4.3 Related literature and positioning of proposal

The method proposed in this study involves pose registration and feature-based 3D model comparison, as well as their implementation for aerospace sheet metal parts. Therefore, the following short subsections review the pertinent fundamental literature.

4.3.1 Pose registration

Comparing 3D CAD models requires either matching the size and orientation of the models (Vranic & Saupe, 2004; Vranic, Saupe, & Richter, 2001; Yang, Lin, & Zhang, 2007) or utilizing size- and orientation-insensitive comparison methods. Granted that using a size- and orientation-insensitive comparison method is incongruous to a comparison method that aims at proposing a semantic MDI, we focus on presenting literature relevant to the former approach.

The process of matching the orientation of the models is also referred to as “pose registration” or “pose estimation” (Vranic & Saupe, 2004). Pose registration processes generally include the translation of 3D CAD models so that their center of mass shifts to the coordinate origin and then using Principal Component Analysis (PCA) to determine their canonical coordinate system axes (Fukunaga, 1990; Vranic & Saupe, 2004; Vranic et al., 2001). The Iterative Closest Point (ICP) algorithm is another method, introduced by Besl and McKay, to address the geometry alignment problems of point clouds to a reference CAD model, applicable to inspection (Besl & McKay, 1992). For similar problems, Pottmann et al. provide another method which is based on instantaneous kinematics and on the geometry of the squared distance function of surfaces (Pottmann, Leopoldseder, & Hofer, 2004).

Tarbox et al. proposed a pose registration method of octree 3D CAD models in which a gross registration estimates a pose transformation and then a fine registration refines the relative pose of the compared CAD models vis-à-vis the objective function (Tarbox, Gottschlich, & Gerhardt, 1993). While our proposal does not use the method proposed by Tarbox et al. (Tarbox et al., 1993), it does borrow that method's division of registration into stages. Eventually, all of the aforementioned pose registration methods normalize the orientation of the 3D CAD models regardless of their feature structure. In their survey paper, Yang et al. reports pose registration as a step before feature extraction in a typical framework of content-based 3D CAD model retrieval (Yang et al., 2007). In contrast, the pose registration method of our proposal is considered as a step after feature extraction.

4.3.2 Feature-based comparison

The literature in the area of feature-based comparison for the purpose of difference identification is quite limited. In one of the few works, Smit et al. presented a way to describe the difference between two models in terms of features (Smit & Bronsvort, 2007). They concluded that the problem is resolvable only if the features between models could be mapped. This conclusion is very significant and will be discussed in the method proposed here.

In contrast, the literature in the area of feature-based comparison for the purpose of similarity assessment (Elinson, Nau, & Regli, 1997; Hong, Lee, Kim, Chu, & Lee, 2005; McWherter, Peabody, Shokoufandeh, & Regli, 2001) or similar model retrieval is quite rich. Similarity and difference are two sides of the same coin; however, there is a delicate point to be highlighted. Similarity (representation of commonality of two objects) is commutative, while difference (representation of objects subtraction from each other) is not commutative. This means that the similarity of a to b is equal to the similarity of b to a, but the difference of a from b is not equal to the difference of b from a. In addition, identifying the similarities or differences of two CAD models could be prioritized differently according to the scenario. For example, similarity

assessment would be a priority in part family formation. On the other hand, in remeshing 3D CAD models, finding differences is more important.

Originally, Cicirello and Regli proposed a new method for implementing solid model comparison by machining features for model retrieval (V. Cicirello & Regli, 2001; V. A. Cicirello & Regli, 2002). The machining features were used to represent the solid models in Undirected Model Dependence Graphs. Excluding the directionality from their graph representation resulted in the elimination of feature order, also known as the precedence constraint. Meanwhile, Li et al. proposed a method for reusing 3D CAD models through Knowledge-Driven Dependency Graph Partitioning, in which the precedence constraint is preserved (M. Li et al., 2010). It is worth noting that the order in which features are created in the native 3D CAD model was used to extract feature precedence, rather than a hierarchical notion. Even though the interdependencies (relationships) of the features are completely conveyed through the proposed method, being dependent upon CAD operations to extract features and their interdependencies limits their proposal's application. Chu et al. proposed the integration of form-feature adjacency graphs and topology graphs to improve similar model retrieval accuracy (Chu & Hsu, 2006). They suggest that if such measures fail to discriminate the best match for the queried model, D2 shape distributions would be used to rank the results of the search based on form features and topology.

4.3.3 Feature definition and description

Our previous work proposed a definition of a feature as: a portion of a geometry model that is significant in at least one of the phases of the product's lifecycle and can be described by its attributes (Ghaffarishahri & Rivest, 2019). The attributes of a feature include its geometry, its relationship to other features and its parameters. For example, the geometry of a hole is its faces, and its relation to its parent feature is represented by the edges connecting it to its parent feature. The parameters of features are from relatively lower- to higher-levels of abstraction. Inspired by the work of Brunetti and Grimm (Brunetti & Grimm, 2005), who proposed a model

of representation layers for feature-based shape data, we propose that feature parameters be categorized as dimensions, dimensional constraints, geometric constraints and representative elements. The representative elements of a feature are of the lowest level of abstraction, as they are basically geometric or topological elements. Dimensions and geometric constraints are of relatively higher-level of abstraction, while dimensional constraints are of the highest-level of abstraction. Figure 4.1 (a) displays a graphic way to organize the information of a feature and its attributes with the help of illustrations (Figure 4.1 (b) to (d)) of an aerospace sheet metal part.

Given that the parameters of a model are considered significantly relevant to the engineering knowledge of a part (Brunetti & Grimm, 2005), their detailed explanation helps in understanding their application in this work. Dimensions constrain the geometry or the relative positions of shape elements, and dimensional constraints constrain dimensions through equations or inequations. For example, the diameter of a hole or the radius of a corner constrain their shape, and a dimensional constraint relates these values according to design rules. Geometric constraints relate the geometries of shape elements. For example, parallelism relates the geometry of two planes to each other. The representative topological or geometric elements are parts of feature definitions that are used to calculate their geometry. For example, the plane or surface that a flange rests on is a geometric element that is used to calculate the geometry of that flange.

A feature's relationships to other features are its semantic links to them. We propose that the relationships of a feature to other features are formed by a set of topological elements connecting their geometry together. These relationships reveal the feature structure of geometry models. Such feature structures, which are dependent on the topological elements, are immune to the non-uniqueness of modeling operations' sequences.

Parent-child relationships have been used in previous works to rationalize relationships between features in sheet metal parts (Z. Liu et al., 2004; Sunil & Pande, 2008). Parent-child

relationships are limited to refer exclusively to the relationship between an item of the hierarchy with another one at one level higher or lower. Because the parent-child relationships between features are not sufficient to describe the multi-level hierarchy of features in 3D CAD models, we introduce some helpful nomenclature. Here, a feature that depends on another feature, directly or indirectly, is its *subordinate feature*, and the feature it is dependent upon, directly or indirectly, is its *superior feature*. A feature could be subordinate to its superior feature and at the same time superior to its subordinate feature. An *immediate* subordinate or superior feature is a feature at one level lower or higher in the hierarchy, respectively. An extended subordinate or superior feature is a feature at more than one level lower or higher in the hierarchy. Hierarch is a feature of highest rank, which does not have a superior feature in the feature structure. Two features are peers if they are at the same rank in the hierarchy and from the same branch (their superior features to the hierarch are in the same order).

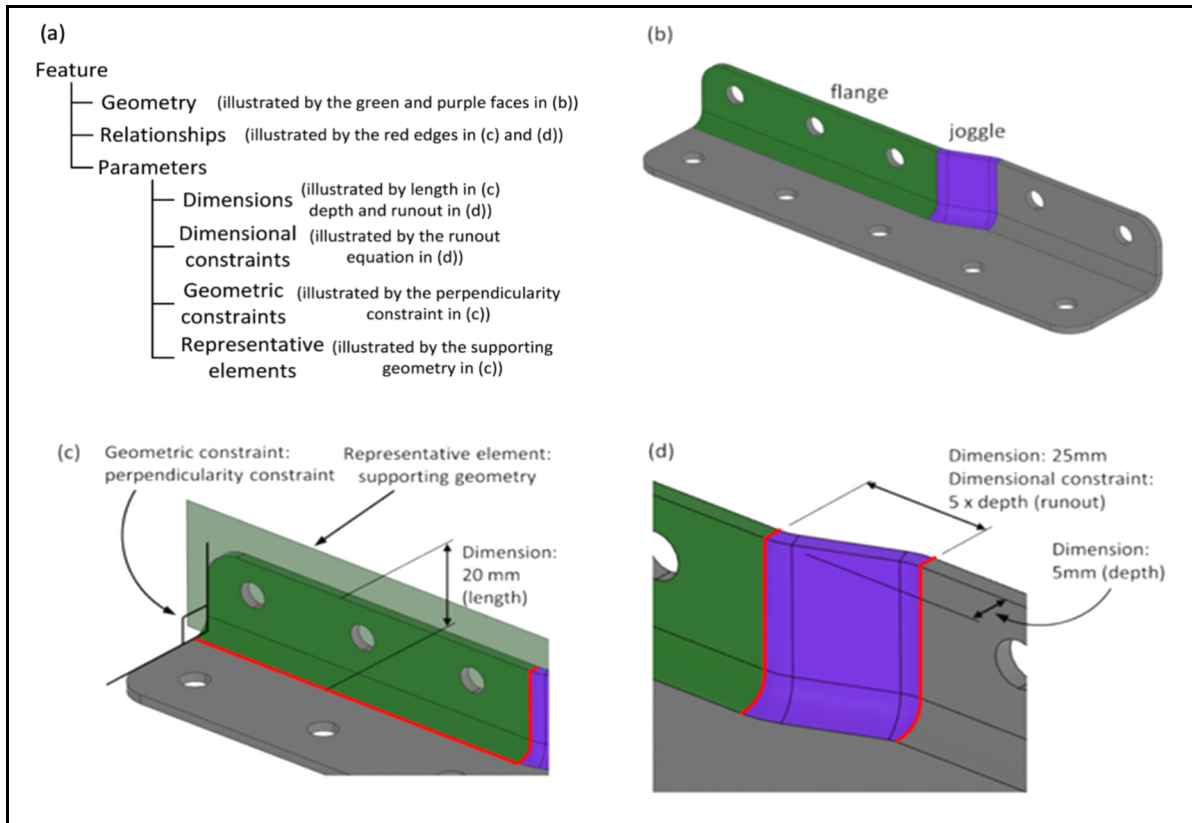


Figure 4.1 A representation of feature information and the illustration of its attributes

4.3.4 Aerospace sheet metal sample parts

Contemporary feature-based modeling solutions are specialized for specific applications, such as aerospace sheet metal part design. Contrarily, the MDI solutions and methods are not nearly as specialized (Brière-Côté et al., 2012a, 2013); however, they could take advantage of being approached with specific applications in mind. Given that in a previous paper we proposed an automated feature recognition method for Aerospace Sheet Metal (ASM) parts, we used the same domain of application to propose and illustrate our MDI method in this work.

The features in ASM parts included in this study were observed by studying the design guidelines and 168 diverse structural sheet metal parts of aircraft structure. A structural system is comprised of a thin-skinned shell which is stiffened by longitudinal stringers supported by transverse frames to form a semi-monocoque structure (Niu, 1999). The parts that we studied for this paper were all produced by brake-forming or hydro-forming, and thus skin panels and stringers were omitted. Brake-formed and hydro-formed parts may include frames, bulkheads, passenger and cargo floor structures (Niu, 1999) and cockpit components.

The generic features of the ASM parts were listed as web, trim features (cutout, hole, stringer cutout, bend relief and corner) and deformation feature (lightening cutout, lightening hole, flange, lip, joggle, twin joggle, deformed flange, deformed web and bead) in our previous paper (Ghaffarishahri & Rivest, 2019). The previous taxonomy of features of the ASM parts was based on the manufacturing viewpoint. In this work, however, the focus is on functionality and design intent, and so holes are divided into attachment holes and tooling holes, while flanges are divided into stiffening flanges and attachment flanges. Figure 4.2 illustrates examples of all these features in actual ASM parts.

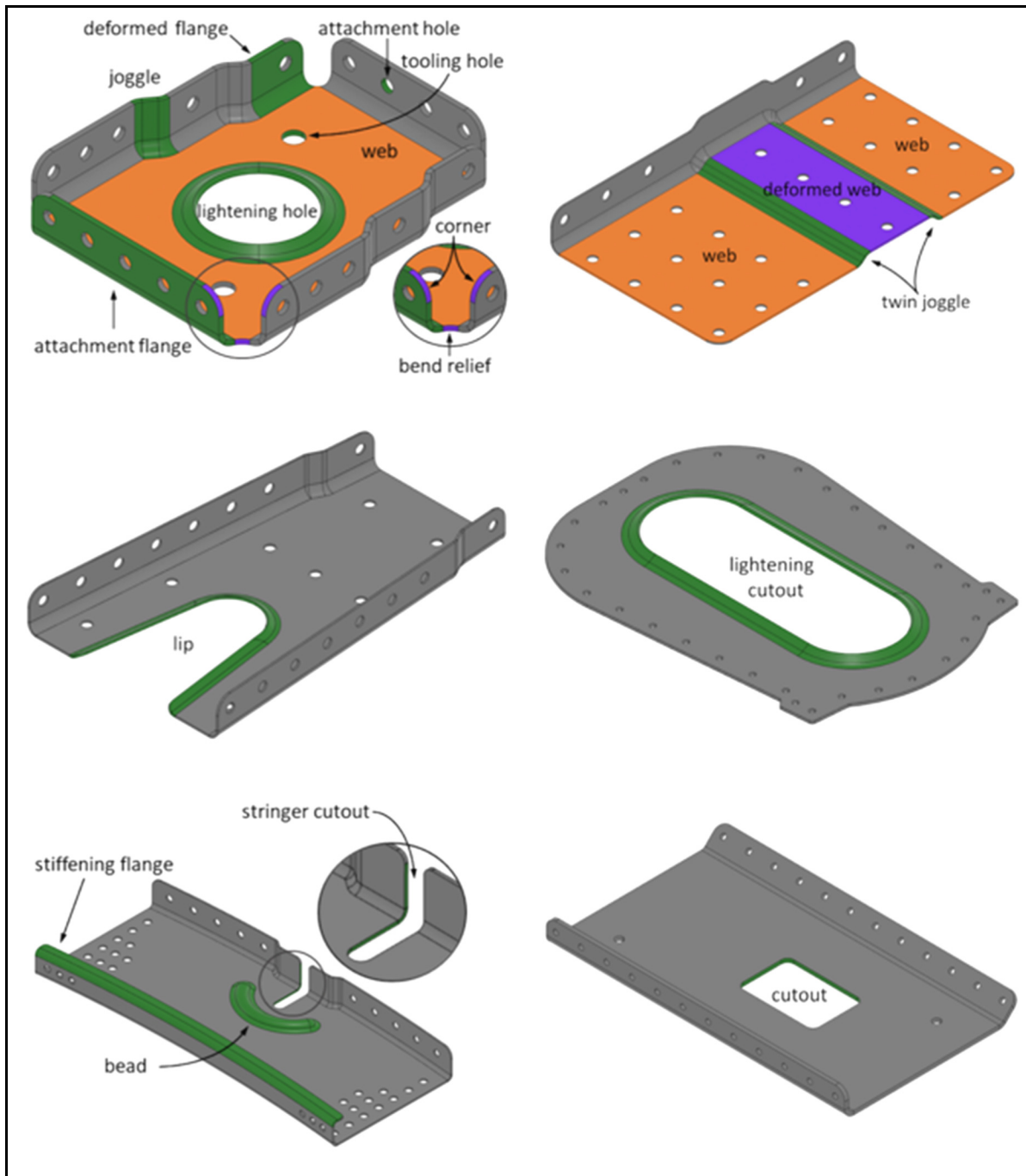


Figure 4.2 Examples of features in real-world ASM parts
From Ghaffarishahri et Rivest (2019)

Attachment and stiffening flanges can be subcategorized according to their class: planar or curved, immediate or return, single or conjoint, and closed, perpendicular or open depending on the angle between a flange and its superior feature. The examples illustrated in Figure 4.3 show some variations of attachment flanges and stiffening flanges that occur in real-world part designs.

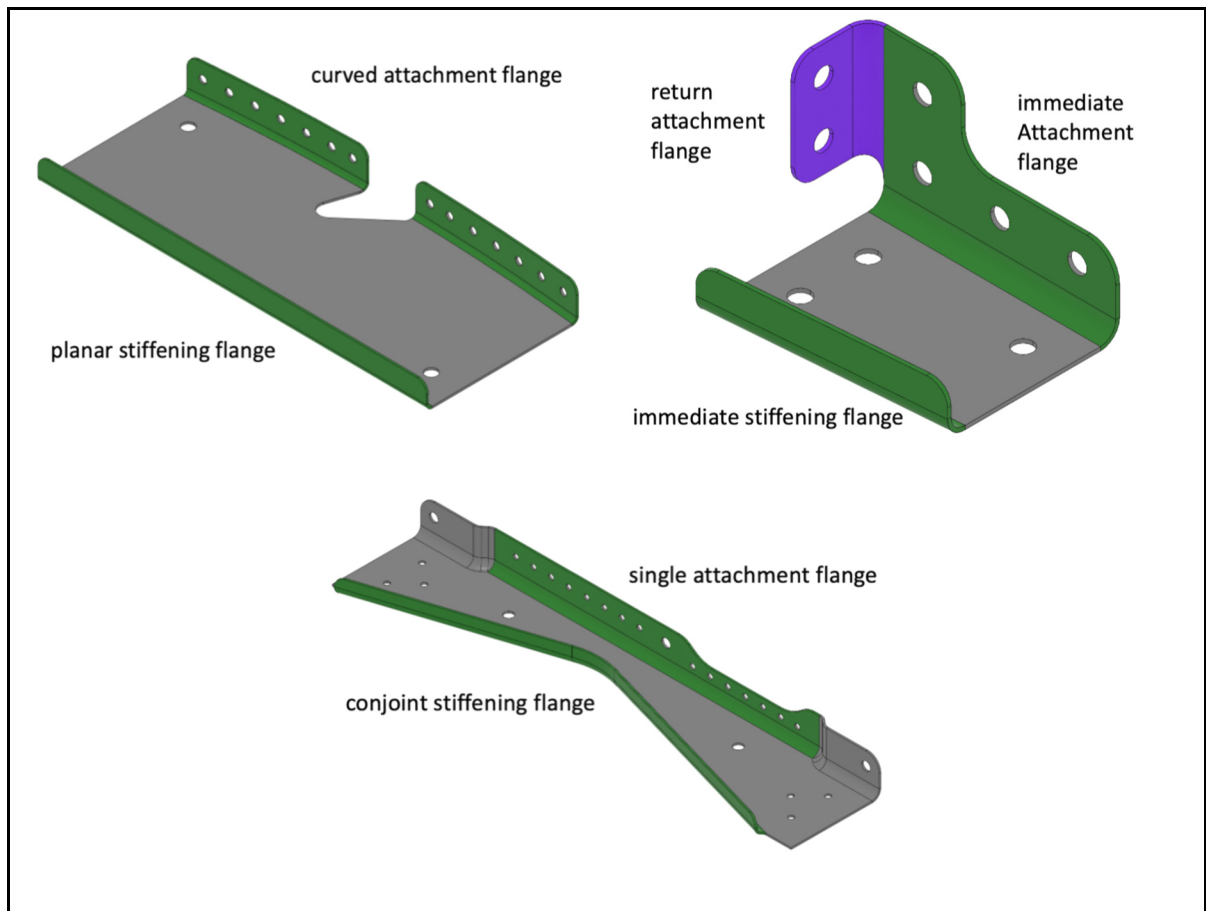


Figure 4.3 Variations* of attachment flanges
and stiffening flanges

* Variations that occur in real-world part designs

In this work, the taxonomy is adapted to meet the needs of semantic MDI. The ASM features are categorized into *base feature*, *contact features* and *refinement features*. In many domains of feature-based 3D CAD modeling, there is a base feature that the rest of the features are built

upon or are modifying. Here, the web is the base feature in brake- and hydro-formed ASM parts. The contact features are the features, other than the web, that are involved in either parts' interfacing or in facilitating the interfacing, so as to provide sound contact between interacting parts. These include the attachment flange, joggle, twin joggle, deformed flange, deformed web and attachment hole. Contact features have specific functions like attachment (fulfilled by attachment flanges, deformed flanges, deformed webs or attachment holes) and adjustment (fulfilled by joggles or twin joggles).

The refinement features are non-contact features that are created for refinement purposes such as weight saving, or for allowing pass-through, stiffening, or manufacturing purposes, or a combination of these purposes. Refinement features include corner, bend relief, lightening hole, lightening cutout, cutout, stringer cutout, lip, bead, stiffening flange and tooling hole. It is worth noting that using attachment holes on the web with a diameter smaller than the diameter of a standard tooling hole for tooling purposes, if required, is a common practice. Figure 4.4 illustrates the feature taxonomy of the studied ASM parts in terms of their functions and purposes.

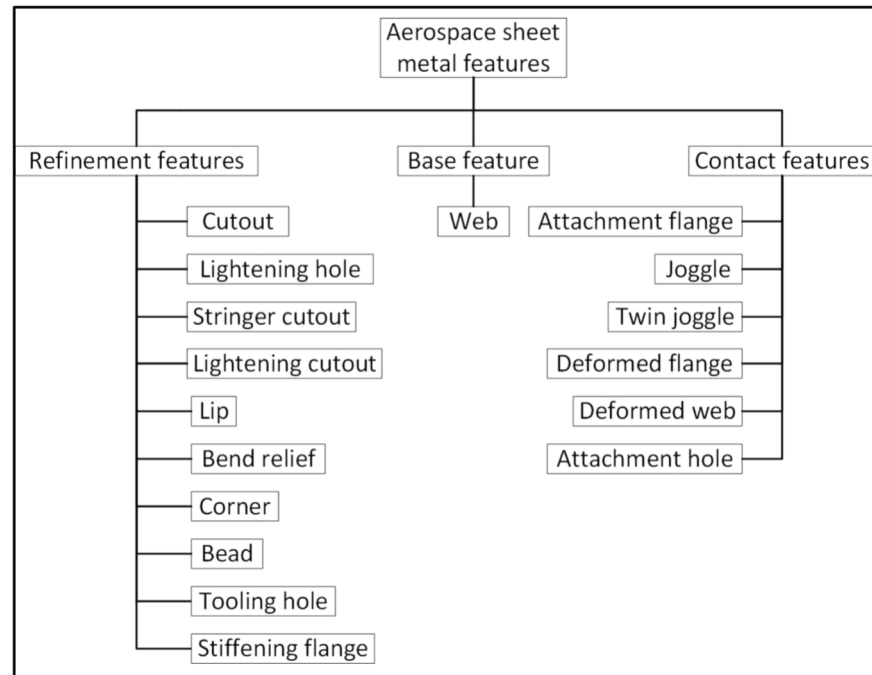


Figure 4.4 The taxonomy of feature types in the studied ASM parts in terms of their functions and purposes

To clarify the representation of ASM features, the web, an attachment flange, a corner, and a lightening hole of an actual ASM part are illustrated in detail in Figure 4.5. Figure 4.5 (a) shows a 3D CAD model of the part and its features, which are colored. The geometry of the features is formed by the B-rep elements (their faces), some of which are shown separately in Figure 4.5 (b). The relationships between each feature and its subordinate or superior feature is formed by the edges shared between their faces, indicated in Figure 4.5 (b) with arrows pointing to the superior feature. In Figure 4.5 (c), the lower-level parameters of these features are illustrated and their higher-level parameters are indicated.

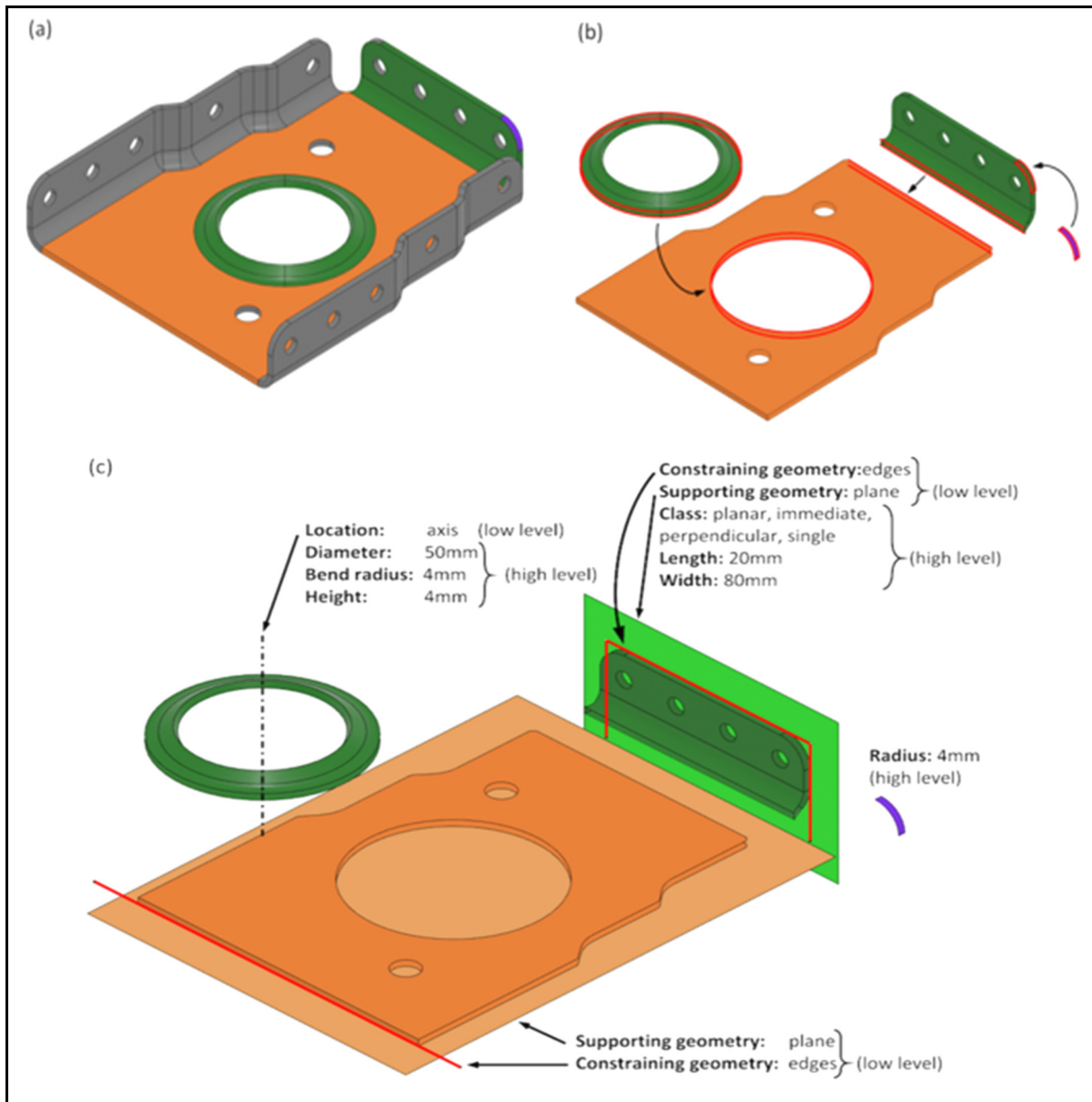


Figure 4.5 Representation of the web, an attachment flange, a lightening hole and a corner of an actual ASM part according to the feature definition

4.4 Proposed method

The proposed feature-based model difference identification method is divided into two stages: pose registration and difference identification. The difference identification stage can in turn be divided into two main steps: 1) *commonality segregation* and 2) *difference identification*

and *difference characterization*. Figure 4.6 summarizes the proposed method; some steps taken in the example illustrated in section 4.4.3 are color coded to match the colors of the pertinent shape elements in Figures 4.13 through 4.16.

Before detailing the pose registration and difference identification methods, it is important to explain that these methods are based on an explicit representation of the parts in terms of their features; feature models. These feature models could be created in a feature recognition preprocessing stage (Ghaffarishahri & Rivest, 2020), which is excluded from the scope of this paper.

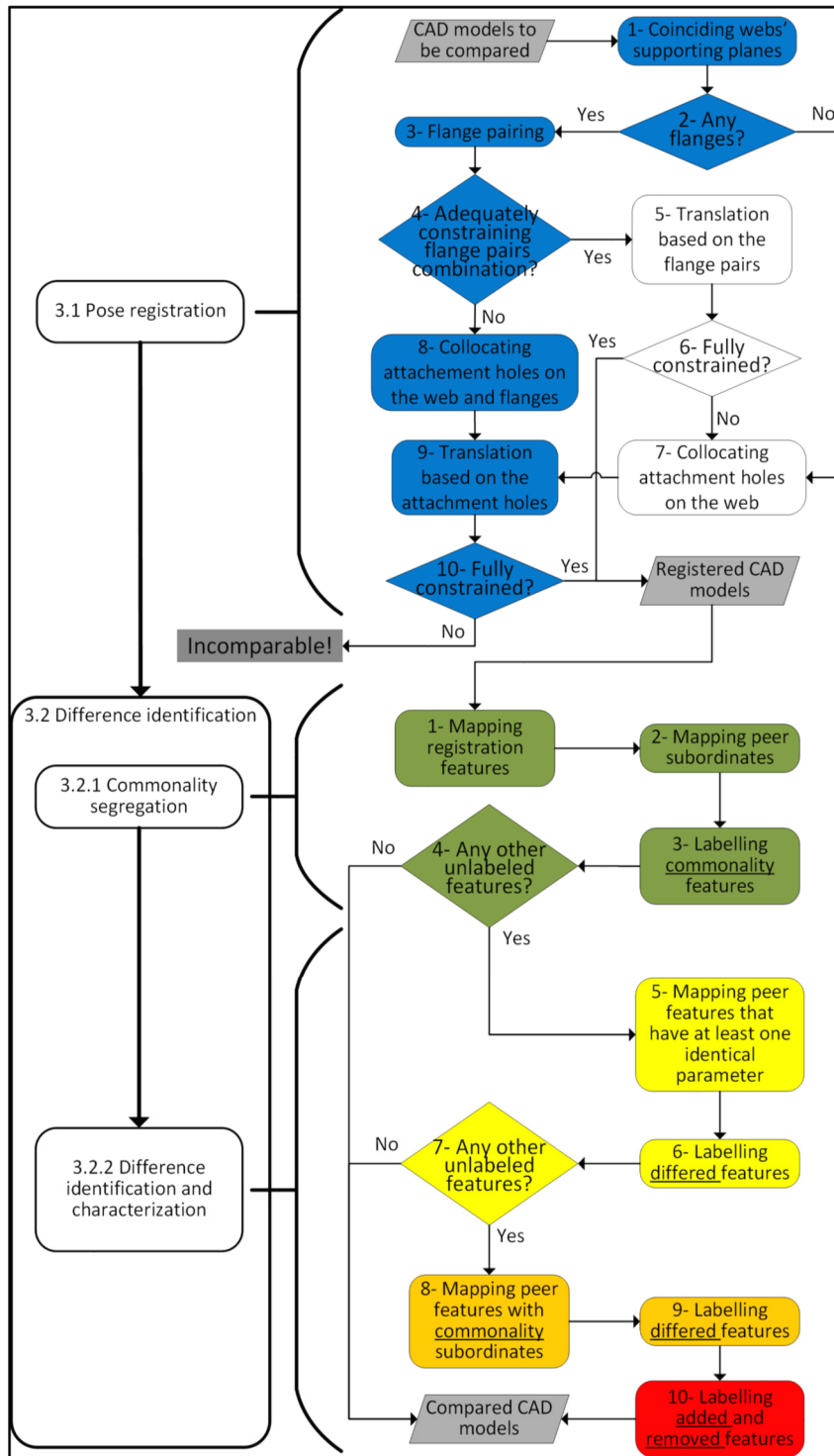


Figure 4.6 Overview of the proposed pose registration and difference identification stages

4.4.1 Pose registration

One of the gaps that this work aims to address is registering the compared 3D CAD models semantically, here called *semantic registration*. We assume that all the features that compose a part are purposeful. This provides the opportunity to semantically register feature-based 3D CAD models according to the unique purpose of the features in our specific domains of application. Linking design intent, parts' functionality, and features have been the subject of other studies, e.g., for robust CAD modeling (Cheng & Ma, 2017; Roy & Bharadwaj, 2002). In ASM parts, functionality and features are intertwined into design practices. With a focus on the design practices of ASM parts, the 168 sample parts mentioned above were investigated, and interfacing and attachment were found to be the indispensable functionality of all ASM parts (as explained in section 2.4). Therefore, the features vital to interfacing and attachment are considered of semantic significance in order to propose a method for semantic registration. For example, the frames, bulkheads, passengers and cargo floor structures are formed by assembling ASM parts together, which makes features with interfacing and attachment functionality fundamental to part design.

The web is the base feature; the rest of the features of parts are formed directly or indirectly based upon it. As a result, the semantic registration of two ASM parts must involve pairing their webs. The notion of using base features to semantically pair feature-based 3D CAD models of parts is applicable to different domains of application, e.g. machined parts, non-aerospace sheet metal parts, composite parts, etc.

In addition to using the base feature, attachment flanges (essential to the interfacing of ASM parts) and/or attachment holes (necessary for attachments) should also be used in semantic registration. Attachment flanges are given precedence to attachment holes in their use for semantic registration. Nevertheless, in the absence of attachment flanges, or when they are not

constraining enough to register the two 3D CAD models, attachment holes are taken into account.

According to the above-mentioned premises, we propose the following steps, which are illustrated by the example in Figure 4.7 (a):

1. *Coinciding webs' supporting plane*: Translate the target model so that the supporting plane of its web coincides with that of the reference model, as shown in Figure 4.7 (b).
2. *Checking for the presence of any attachment flanges*: Check if the two compared models have attachment flanges; if not, skip to step 7.
3. *Flange pairing*: Pair the attachment flanges of the two compared models according to their classes, as shown in scenario A and scenario B in Figure 4.7 (c). An attachment flange in each of the models could be paired simultaneously with one or more of the attachment flanges of the other model if their classes agree, as in the example here.
4. *Checking for the presence of an adequately constraining combination of flange pairs*: Check if there could be a single combination of attachment flange pairs that enables the superposing of all or of a plurality of the paired attachment flanges simultaneously so that their supporting geometries coincide. It is possible that there are more than one attachment flange pairs that enable the superposing of all or of an equal plurality of the paired attachment flanges simultaneously (illustrated by Figure 4.7 (c) and (d)). If there is not a single combination of the attachment flange pairs that could enable superposing all or a plurality of them, skip to step 8.
5. *Translation based on flange pairs*: Calculate the translation to superpose the selected attachment flange pairs and translate the target model according to the calculated translation, as shown in Figure 4.8 (e).
6. *Checking if the parts are fully constrained*: Evaluate if the models are fully constrained to each other (so they have no relative degree of freedom). If the models are not fully constrained to each other, continue with the next steps, otherwise, the models are semantically pose registered. Being fully constrained could be evaluated by verifying that

- the mapped attachment flanges between the models are greater than one and are not only parallel planar attachment flanges or concentric curved attachment flanges.
7. *Collocating attachment holes on the webs:* Identify the largest set of attachment holes on the webs that could collocate without violating the previous attachment flange superposition and skip to step 9. It should be noted that this step is used in cases where there is no flange on a part, or when flange pairing did not suffice to fully register the models.
 8. *Collocating attachment holes on the webs and flanges:* Identify any of the combinations of attachment flange pairs that enables the largest set of attachment holes on the webs to collocate. If required, identify the combinations of attachment flange pairs that also enables collocation of the largest set of attachment holes on the attachment flanges themselves. Figure 4.8 (f) and (g) illustrate how collocating attachment holes on the web and the attachment flanges themselves could be determinant in selecting the attachment flanges pair combination.
 9. *Translation based on attachment holes:* Calculate the translation to collocate the set of attachment holes and translate the target model according to the calculated translation.
 10. *Checking if the parts are fully constrained:* Evaluate if the models are fully constrained to each other (so they have no relative degree of freedom). If the models are not fully constrained to each other, they are not viable to be registered based on their web, attachment holes and attachment flanges (the features that are fundamental to the functionality of aerospace sheet metal parts), so they are considered incomparable due to a lack of adequate functional similarities.

The proposed semantic registration steps only include translation and rotation transformations and is therefore rigid. Before proceeding to the difference identification stage of this work, it is relevant to discuss the comparability of the models. Although an MDI method per se does not need to include verifying an adequate commonality condition, such condition is a prerequisite that needs to be stipulated. The studied ASM parts are designed for the purpose of structuring the fuselage and cockpit to be adequately strong and to provide a frame to which

to assemble other systems. The web, attachment holes and attachment flanges are the main features to enable the parts' functionality. While what constitutes adequate commonality could be perceived subjectively, here we suggest that the presence of enough of the features fundamental to the parts' basic functionality is essential to consider two parts comparable. Identifying the differences between two parts is meaningful so long as the parts have adequate commonality. Comparing two parts without enough functional commonality would allow futile difference identification results. Interestingly, if the adequate commonality condition is based on their constituent features, and the constituent features are considered uniquely purposeful, the comparison scenarios must be restricted to parts whose design purposes are adequately similar.

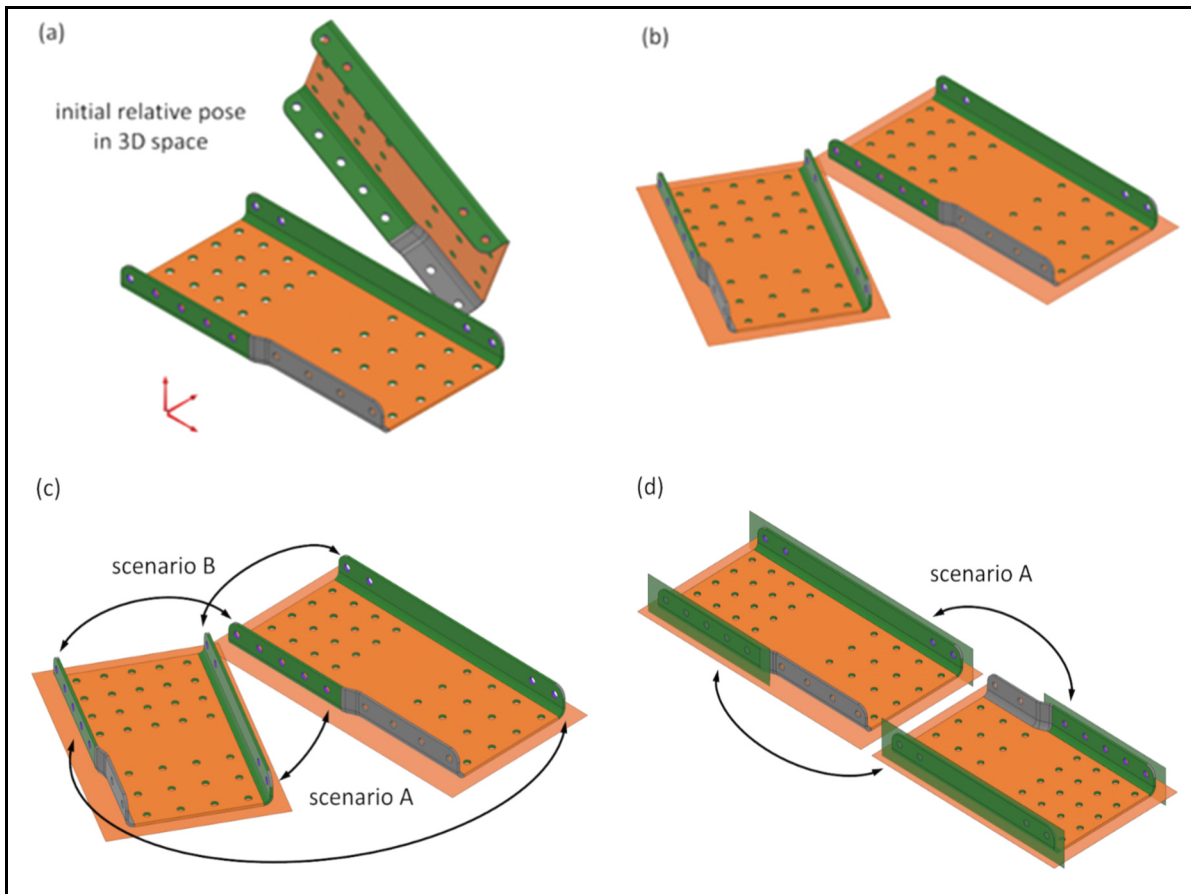


Figure 4.7 Illustration of the proposed semantic registration method through an example (a, b, c, d)

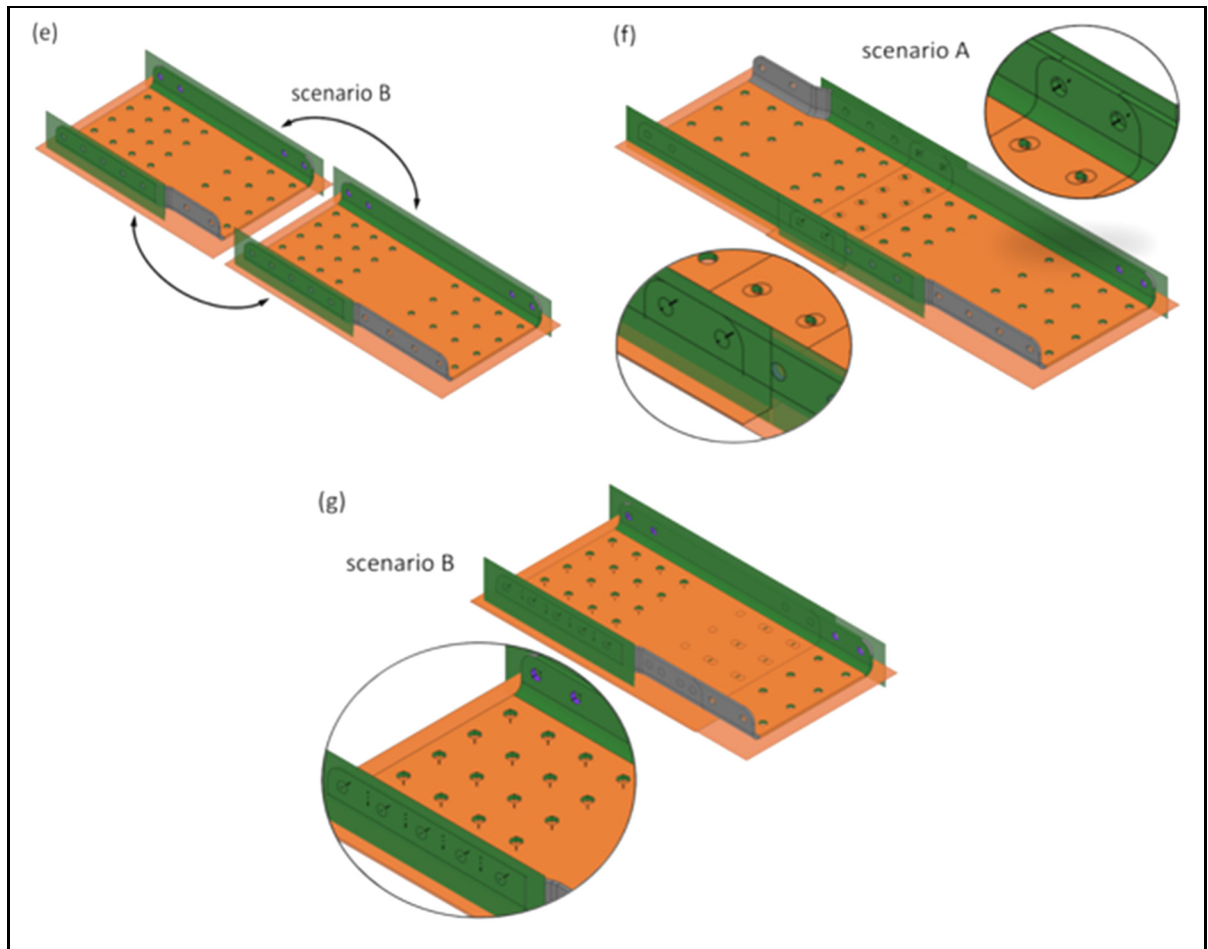


Figure 4.8 Illustration of the proposed semantic registration method through an example (e, f, g)

4.4.2 Difference identification

In a very insightful paper, Viswanathan et al. asked a question that is pivotal to this work: “How can the dimensions and positions of geometric features in the two given components be compared?” (Viswanathan, Chowdhury, & Siddique, 2008). Although their work proposed a measure for calculating a commonality value based on dimensions and position, their pair-wise comparison viewpoint matches perfectly with the scope of this work. They also proposed using an automated feature recognition method rather than extracting features from modelling

operations in the native 3D CAD models. In response to the question asked by Viswanathan et al., the parameters of features, which resemble the role of dimensions and positions of geometric features, are involved in the difference identification method proposed here. A feature's high-level parameters include all of its related dimensions, and its low-level parameters constrain its position.

In order to identify differences, comparison is required. However, comparison between any two objects yields both their commonalities and their differences. Excluding the commonalities of two objects from their comparison results effectively identifies their differences. Thus, we propose to begin the difference identification process by a number of initial steps to identify and exclude the commonalities between the feature models (commonality segregation), and then to follow with difference characterization steps to finalize difference identification. While the bottom half of Figure 4.6 summarizes the steps of the proposed difference identification stage, Figures 4.9 through 4.12 illustrate these steps in an example.

Commonality segregation

The feature structure of the 3D CAD models – formed by their features, their relationships, and their features' parameters – are used to map features with identical parameters between the compared models (primary mapping). These mapped features form the commonality between the feature models. The commonality segregation steps identify the identical features between the compared models to discard them from the difference identification process.

Figure 4.9 (a) and (b) displays two fabricated examples of ASM parts with their feature structures. The AHs, AFs, BR, Cs, TJ and DF nodes represent the attachment holes, attachment flanges, bend relief, corners, twin joggle and deformed flange, respectively. Different symbols are used in this figure to distinguish between different hierarchy levels. The following four steps are proposed to complete the commonality segregation:

1. *Mapping registration features:* Map together the webs, attachment flanges and attachment holes that were involved in semantic registration provided that they have identical parameters. Figure 4.10 (c) and (d) illustrates such mapped features through the blue colored features. Because the compared models can be fully constrained only by coinciding their web's supporting plane and superposing their attachment flanges, they can be registered using only these features. The web and the attachment flanges (AF1 and AF2) are thus mapped at this step.
2. *Mapping peer subordinates:* Search for and map together every two peer (immediate or extended) subordinate features of the same type (e.g. AH9 from the reference model and AH10 from the target model) of the already mapped features, provided that they have identical parameters. Figure 4.10 (e) and (f) illustrates such mapped features via the green colored features. Note that features like the deformed flanges (DF) and the attachment holes on them (AH12 from the reference model and AH13 from the target model) are peer extended subordinate features of the same type that are located on the already mapped attachment flanges (AF2).
3. *Labeling Commonalities:* Label the mapped features (at steps 1 and 2), e.g. commonality, to distinguish them from the unmapped features.
4. *Checking for the presence of any remaining unlabeled features:* Check if there are any unlabeled features and take them to the next stage. If all features are already labeled, the models' comparison is over, and their difference is an empty set of features.

Difference identification and characterization

Before continuing to the difference identification and characterization stage, we need to elaborate on the possible differences between compared 3D CAD models in terms of their features. Difference in terms of features can be identified as added, removed or differed features. Differed features are features that are partially similar or partially different. A prime challenge in difference identification is distinguishing differed features from added and removed features. If no differed feature is identified, the differences between the compared 3D

CAD models are inevitably identified as removed features from the reference model or as added features to the target model.

To identify all the differed features, we propose a secondary mapping process that is not restricted to features with identical parameters. The following steps, except for step 10, are proposed to complete the secondary mapping:

5. *Mapping peer features with at least one identical parameter:* Search for and map together every two peer features of the same type that are unlabeled, provided that they have at least one identical parameter. If there are multiple mapping possibilities, prioritize mapping together the features with the greatest number of identical parameters. Figure 4.11 (g) and (h) indicates such mapped features in yellow.
6. *Labeling differed features:* Label these mapped features as differed, to distinguish them from the rest of the previously mapped features and the still unmapped features.
7. *Checking for the presence of any other unlabeled features:* Check if there are any unlabeled features and take them to the next stage. If all features are already labeled, this means that the models' comparison is over, and that their difference does not include any added or removed features.

At this point, the remaining unlabeled features are either added or removed features – depending on belonging to a reference or a target part – or differed features. Indeed, subordinate features of an unlabeled feature may already be mapped and labeled, which would contradict with the superior feature to be either an added or a removed feature. For example, all the parameters of a joggle could vary between two parts, without impacting any parameters of its extended subordinate attachment hole (diameter and location of a hole with respect to the part's coordinate system). In all cases, the unlabeled superior features of a pair of already mapped features which are labeled as commonality must be considered as differed. To identify such differed features and to distinguish them from added or removed features, the following steps are proposed:

8. *Mapping peer features with commonality subordinates*: Search for and map together every two unlabeled peer features of the same type, provided that they have an immediate or extended subordinate feature labeled as a commonality.
9. *Labeling differed features*: Label these mapped features as differed. The orange-colored features (the DFs) in Figure 4.11 (i) and (j) indicates such differed features. The deformed flanges' subordinate attachment holes have identical parameters and thus show that the DFs, despite not having any identical parameters, serve similar design intent and functionality. Therefore, the DFs are differed features, as opposed to added or removed features.
10. *Labeling added and removed features*: Identify all the remaining unlabeled (still unmapped) features and label them as added if they belong to the target model and removed if they belong to the reference model. Figure 4.12 (k) and (l) shows such mapped features in red.

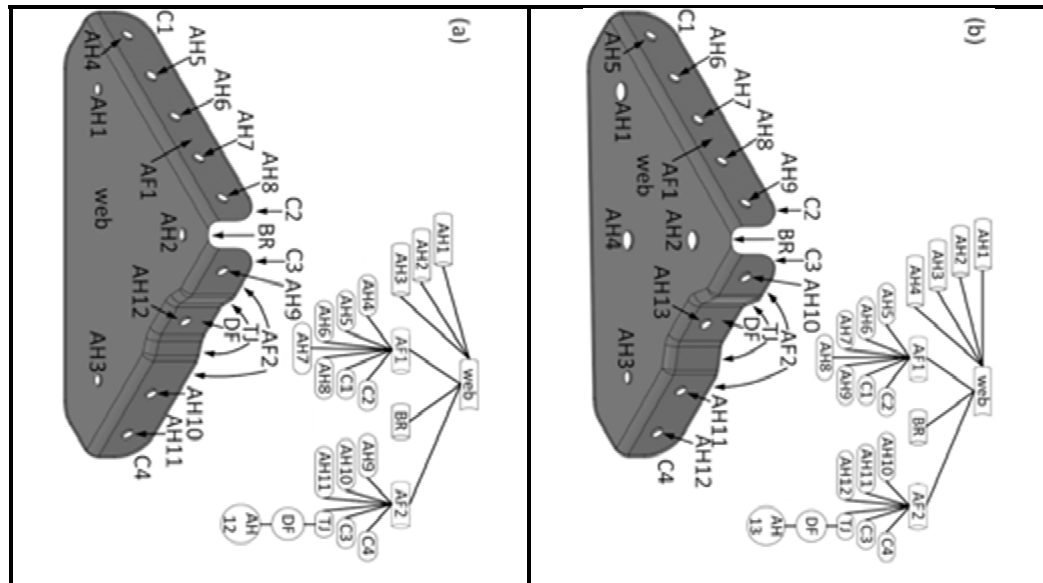


Figure 4.9 The reference (a) and target (b) models for the proposed difference identification method

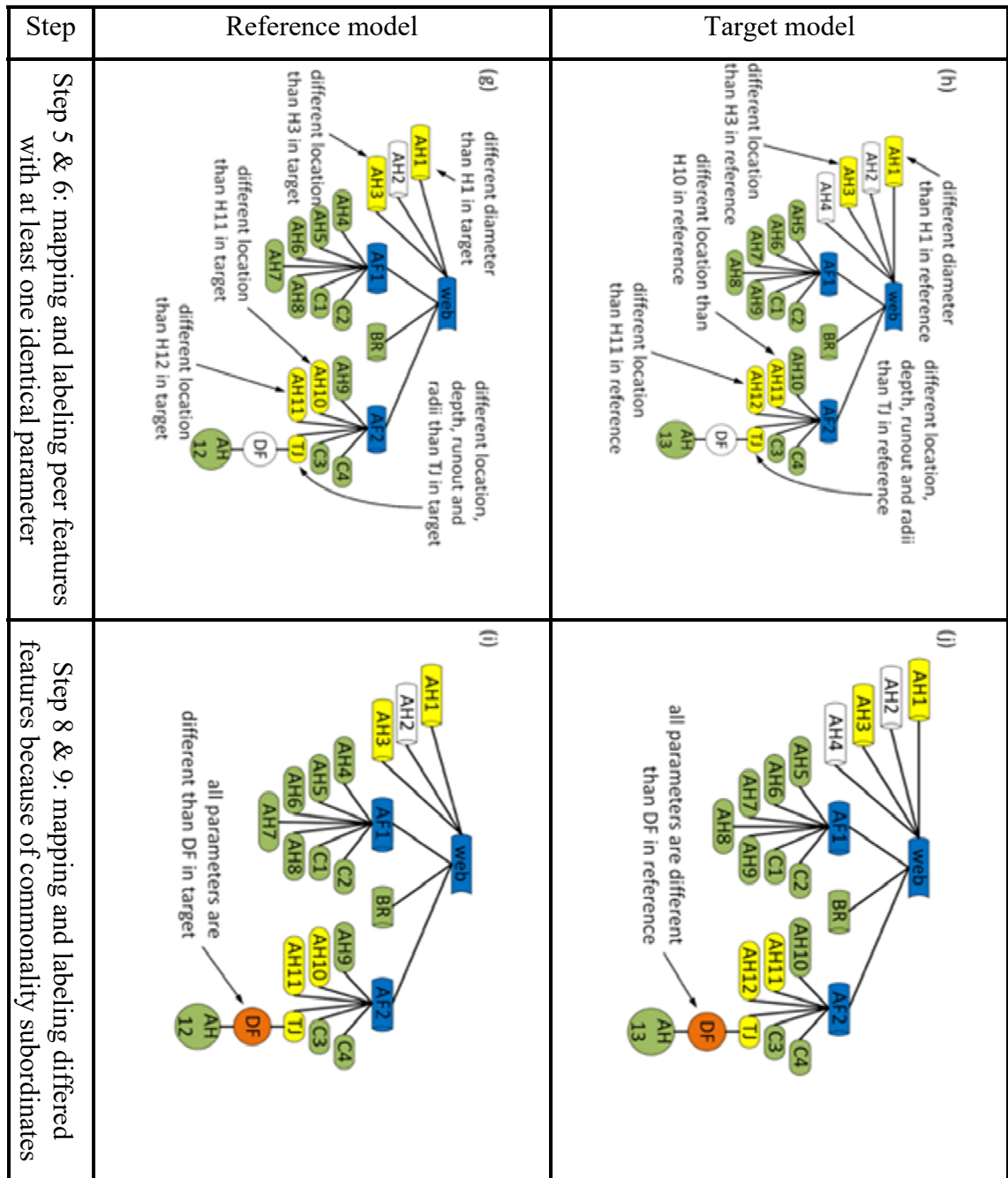


Figure 4.11 Illustrations of key steps of the proposed difference identification method (Steps 5, 6, 8 and 9)

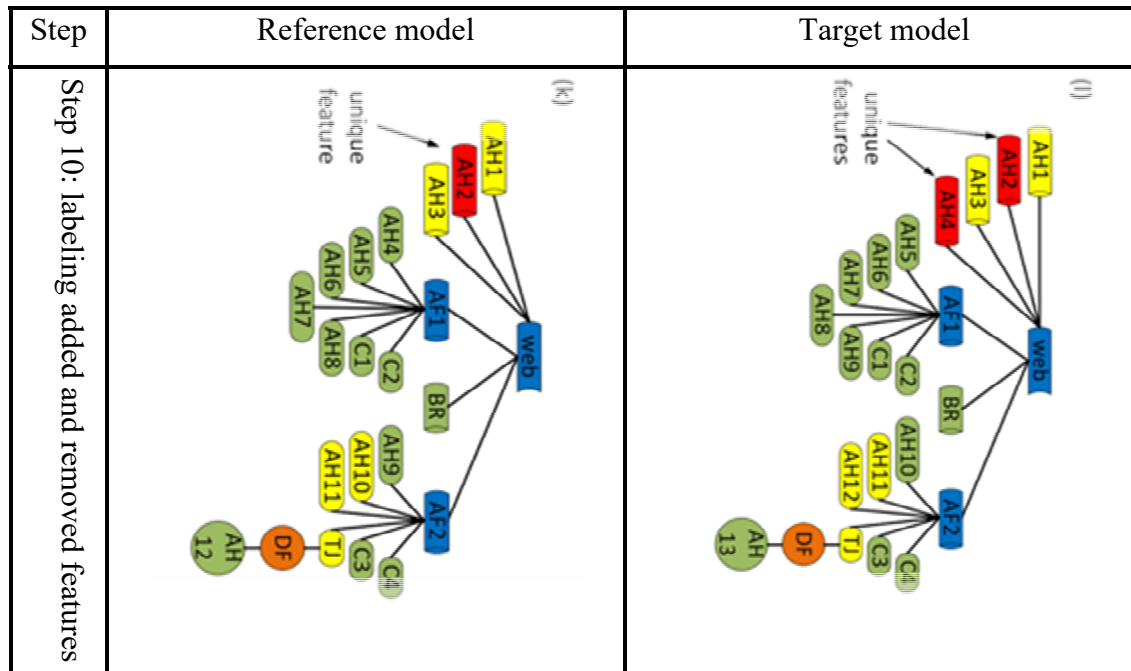


Figure 4.12 Illustrations of a key step of the proposed difference identification method (Step 10)

4.4.3 Illustration of the proposed method

Here, we will explain the execution of the proposed method using an example from the samples that were selected from a pool of real parts of a Bombardier DHC-8-102. B-rep models of the sample part and a new version of it were created to recognize their features according to the method explained in our previous work (Ghaffarishahri & Rivest, 2019). The B-rep model of the new version of the part is translated to a random position in 3D space. Figures 4.13 through 4.16 illustrate the models and the significant steps in identifying their differences.

Figure 4.13 (a) to (c) illustrates the part, two versions of its 3D CAD model and their features' structures. The four corners that are identical between the models and so are removed from the features' structures to simplify the figures are indicated in Figure 4.13 (a). In order to register the two 3D CAD models, they are first translated so that their webs' supporting geometries

coincide. The parts have attachment flanges that can be used towards semantic registration; however, because the attachment flange pair is not fully constraining, collocating the attachment holes on the attachment flanges is utilized to complete semantic registration. It should be noted that the holes on the web are tooling holes and are not involved in the semantic registration process. Finally, the attachment flange pairs and the collocated attachment holes are used to calculate the translation of the target 3D CAD model. The steps of the semantic registration method taken to semantically register the 3D CAD models are being colored blue in Figure 4.6. Figure 4.8 (f) and (g) show the representative elements (the webs' and attachment flanges' supporting geometry in blue and the attachment holes' axes indicated) of the features that were involved in the semantic registration of the target and reference 3D CAD models.

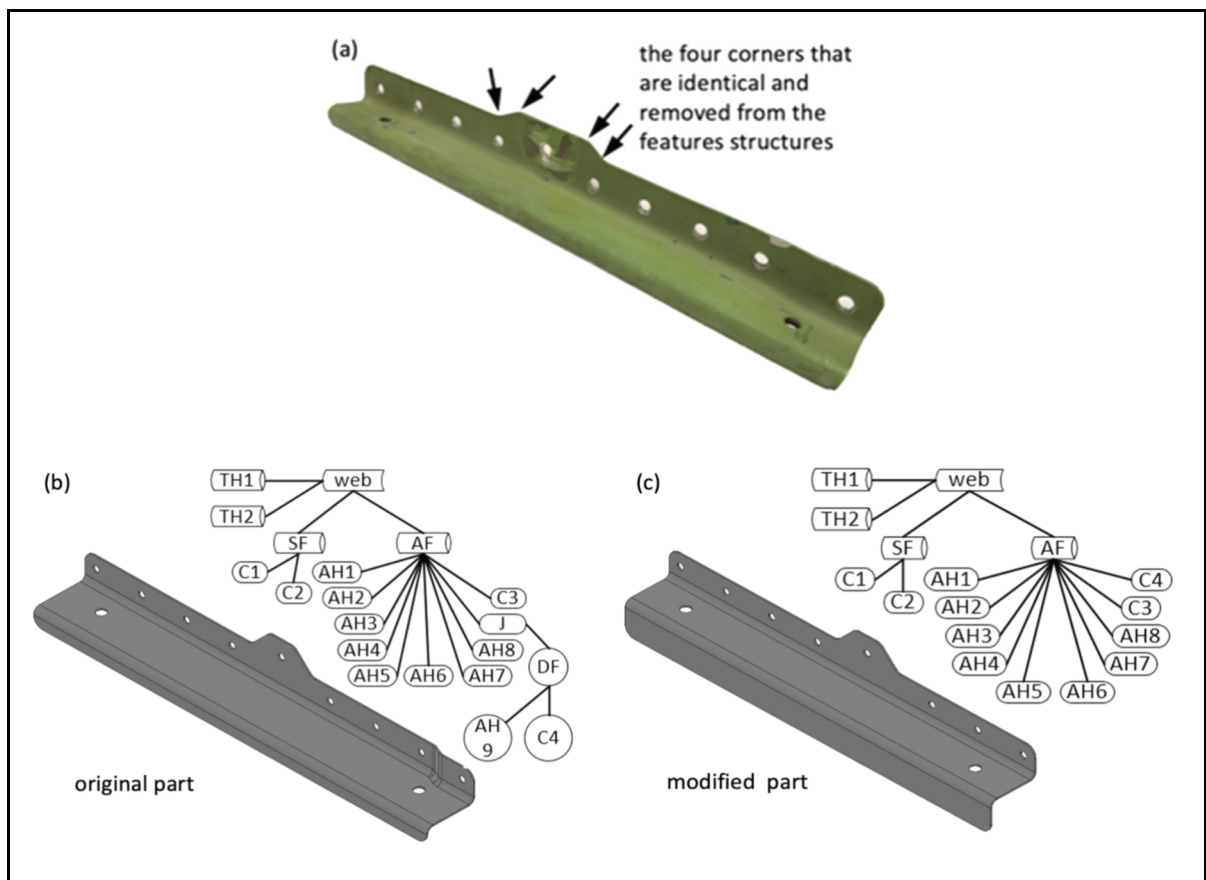


Figure 4.13 Semantic registration of two versions of a 3D CAD model of an ASM part

The steps to identify the commonalities and differences between the compared 3D CAD models are color coded in Figure 4.6 to match the color of the features mapped and labeled in each step. In order to identify and segregate the commonality between the 3D CAD models, the web, attachment flanges and the attachment holes that were used in the semantic registration are evaluated, and those with identical parameters are mapped. Figure 4.14 (a) and (b) shows these features in green and indicate them with arrows. Every other peer subordinate feature of the same type that have identical parameters are identified and mapped (indicated in Figure 4.14 (c) and (d)). The features that have already been mapped are labeled as (shown in Figure 4.15 (e) and (f)), and, since not all of the features of the models have been exhausted, it is now time to identify the differences between the 3D CAD models.

Hence, the peer features of the same type with at least one identical parameter are identified, mapped and labeled as differed. Figure 4.15 (g) and (h) shows these features (the immediate-attachment flange, the attachment holes on the flanges and the tooling hole on the web) in yellow and indicate them with arrows. The attachment flanges are both planar, immediate flanges, and have identical lengths but different widths. The tooling hole and the attachment holes have identical diameters but different locations. The pertinent steps are colored yellow in Figure 4.6.

The peer features of the same type that have subordinate features with commonality labels are also identified, mapped and labeled as differed. Figure 4.15 (i) and (j) shows the stiffening flanges (in orange) of both models that have their subordinate commonality corners (indicated in Figure 4.14 (c) and (d)).

The remaining unlabeled features of the reference 3D CAD model are unique features and therefore are labeled as removed. The remaining of the unlabeled features of the target 3D CAD model are also unique features and therefore are labeled as added. The removed and added features of the 3D CAD models are colored red and indicated in Figure 4.16 (k) and (l).

Finally, the differences between the compared models are as shown by the annotations in Figure 4.16 (m) and (n). The width of a flange is the dimension from side to side, and the length is the dimension from the bottom to edge of the flange.

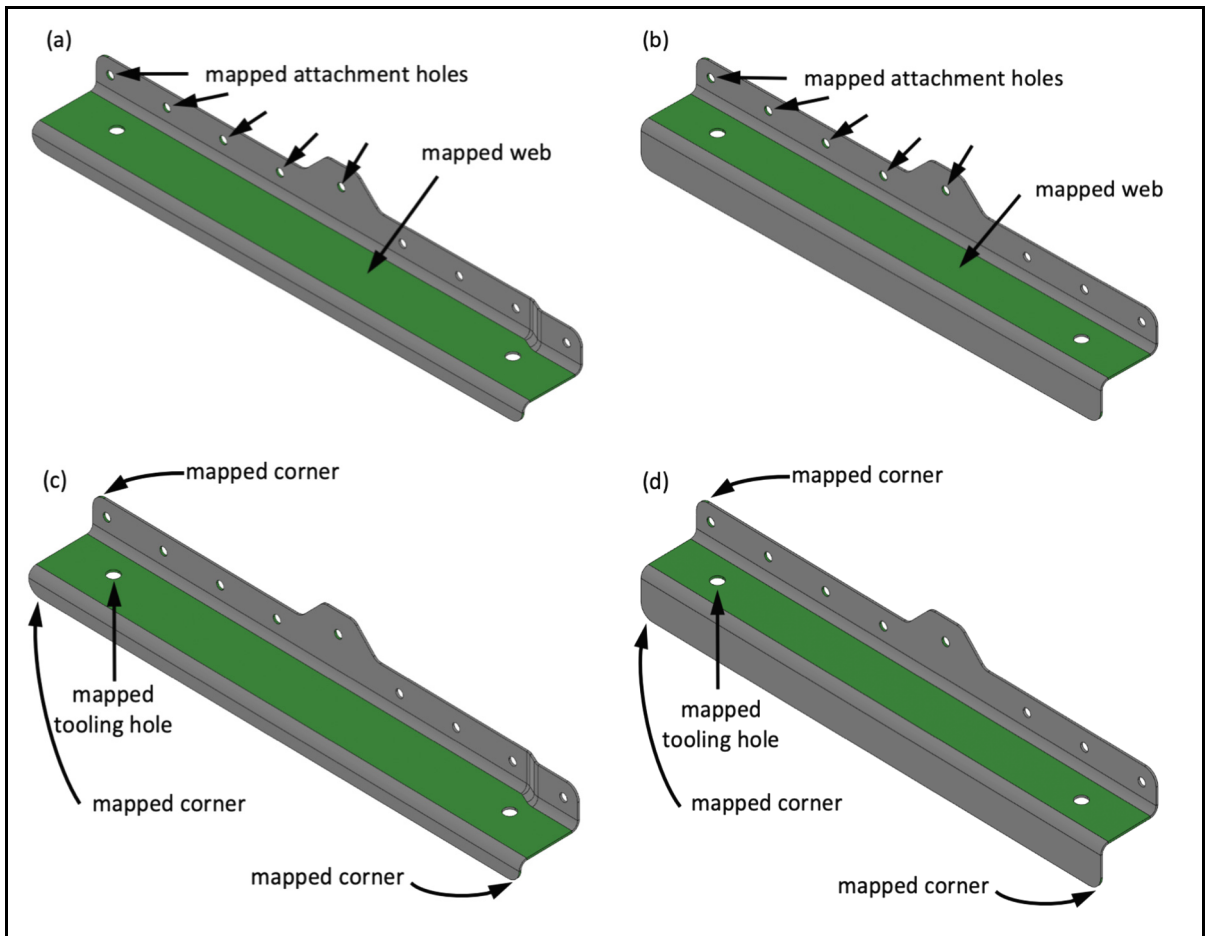


Figure 4.14 Difference identification of two versions of a 3D CAD model of an ASM part (a, b, c, d)

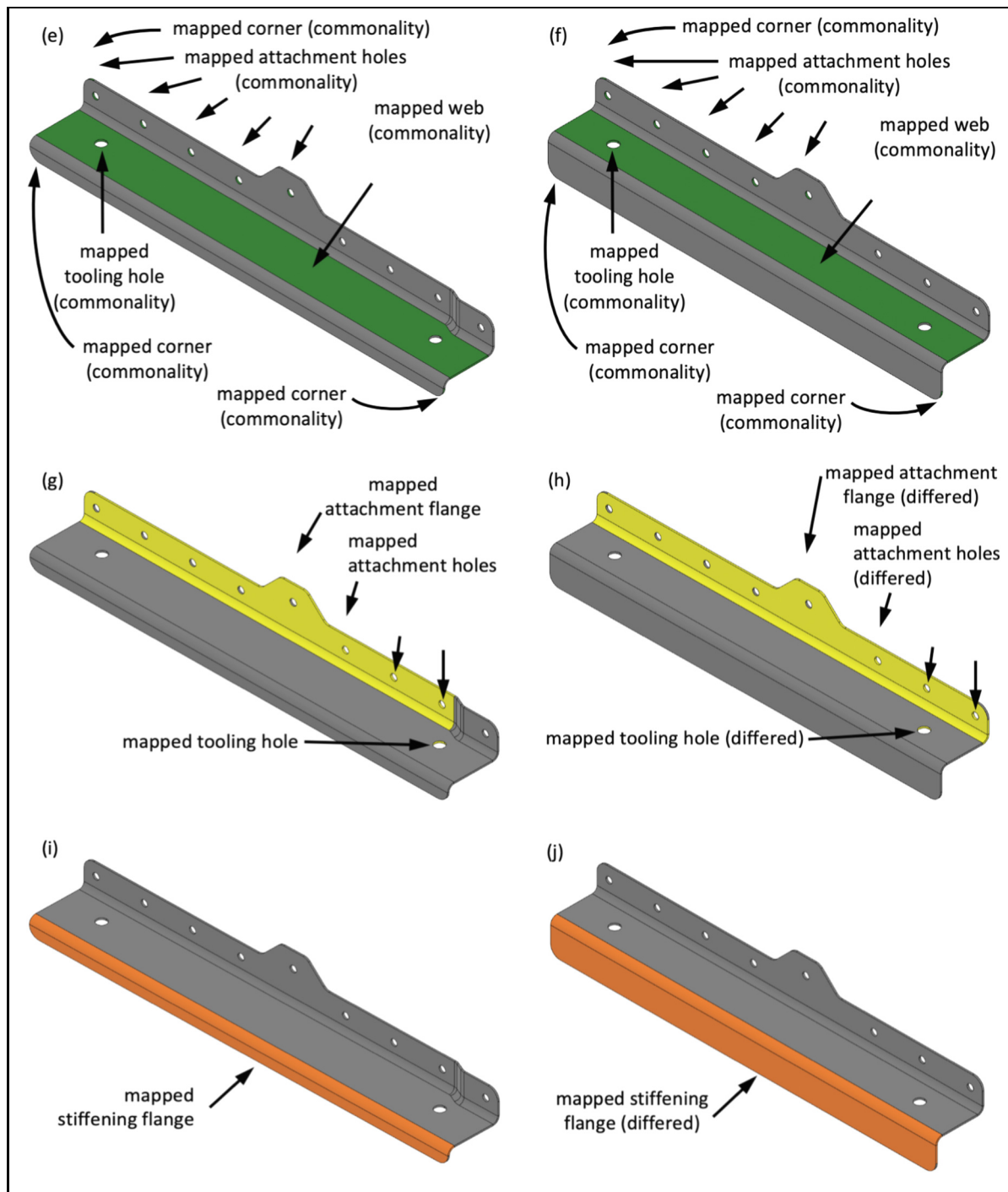


Figure 4.15 Difference identification of two versions of a 3D CAD model of an ASM part (e, f, g, h, i, j)

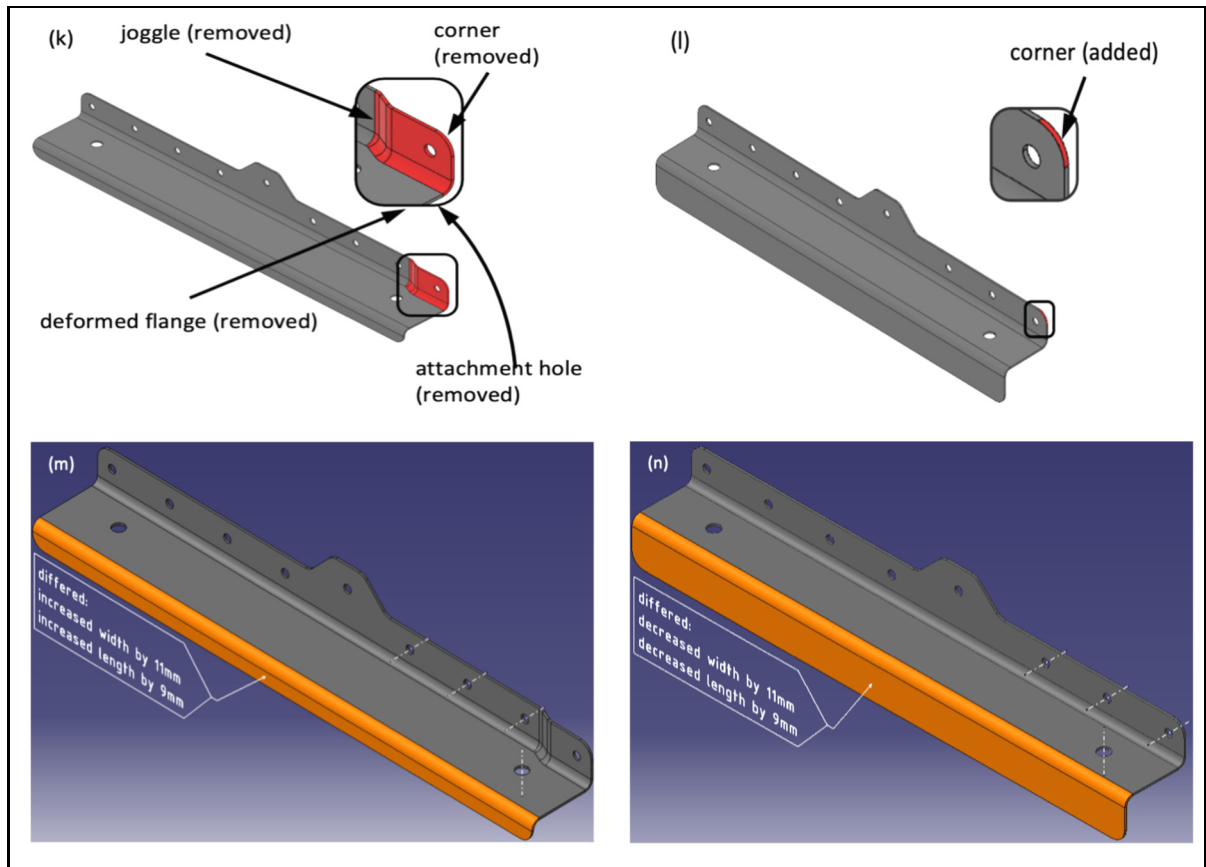


Figure 4.16 Difference identification of two versions of a 3D CAD model of an ASM part (k, l, m, n)

4.5 Discussion

The importance of MDI solutions for effective and efficient information reuse workflows has drawn researchers' attention to propose methods to compare 3D models in various ways. The MDI problem have usually been addressed in a primary pose registrations stage and a secondary difference identification stage. None of the previously proposed pose registration methods take design intent, functions or design features into account. Moreover, none of the previously proposed difference identification methods, although feature-based, take design intent or features' functions and purpose into account, rather the major emphasis has been put on feature-graphs comparison. In this work, the significance of design intent and features' functions and purpose in both pose registration and difference identification have been

recognized and put to good use in the proposed method. In fact, the underlying assumptions in this work are that the compared models represent parts with similar-enough functions and that the functions are served by the same features. The steps are based on a set of principles that are explicitly expressed in the following:

1. Any type of parts has a base feature, e.g. the web, that must be involved and prioritized in pose registration;
2. Any type of parts has a number of absolutely essential features, e.g. attachment flanges and attachment holes, that are needed to design any useful and functional part and must be involved in pose registration;
3. Identifying commonalities between compared parts based on mapping features in the features structure of each part and verifying identical parameters between the mapped features must be prioritized to perform difference identification; and
4. Difference identification must be able to differentiate partially different features, e.g. differed features, from added and removed features.

Based on the first two principles the smallest set of features, i.e. the web, attachment flange and attachment hole, was selected to propose the pose registrations method. These principles are applicable to any MDI solution regardless of the domain of application for which the compared parts are designed. One main strength of this work is the simplicity of its premises and principles, which makes them applicable to a wide range of part types.

Here, ASM parts were used to show the proposed method, because their feature structure and feature-function links are industrially established. In some way this is a limitation of the presented work since ASM design use a limited number of features. However, future works could focus on implementing and adjusting the proposed method to compare parts of domains of application with different feature structures like machined parts.

4.6 Conclusion

This work presented an innovative semantic model difference identification method. It compares 3D CAD models represented as feature structures and identifies their differences based on engineering semantics by evaluating the models' features. The proposed method includes two major stages: semantic pose registration and difference identification.


Semantic pose registration provides the opportunity to register the compared models based on their feature structure and their parameters, both of which convey meaningful engineering information and design intent. Semantic registration takes advantage of design intent, and hence of parts' functionality to meaningfully register the models. It also decouples registration from the feature recognition process. Involving parts' functionality in pose registration rather than mere parts' geometry conveys significant value to the proposed method, indicating successful pose registration as the index of comparability of the models. Here, the semantic registration was based on webs, immediate-attachment flanges and the attachment holes on them. The web is the only feature whose purpose as the base feature mandates its existence in all ASM parts.

In order to identify differences, the proposed method relies firstly on eliminating commonalities between the models. Excluding the commonalities of two objects from their comparison results effectively identifies their differences. Thus, we proposed to begin the difference identification process by means of a number of initial steps to identify and exclude the commonality between the feature models (commonality segregation), and to then identify and characterize the differences. Since it is based on the models' features, the model difference identification method proposed here increases the semantic level of the comparison, which would not be possible by comparisons based on simple geometric elements. Both the semantic registration and the difference identification methods proposed here are essentially novel for their integration of the features' design purposes in the comparison process. No similar

methods have been proposed. The concept of segregating commonalities in order to group differences has not yet been explored, and this work shows its useful application.

CHAPTER 5

A PROTOTYPE OF AN AUTOMATED FEATURE RECOGNITION ALGORITHM FOR AEROSPACE SHEET METAL PARTS

Syedmorteza Ghaffarishahri¹  and Louis Rivest¹ 

¹ Department of Systems Engineering, École de Technologie Supérieure,
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3

Paper published in *Computer-Aided Design and Applications*, August 2022

5.1 Abstract

Automated feature recognition (AFR) makes it possible to abstract semantic information from neutral CAD models. In an earlier work, we proposed an AFR method for aerospace sheet metal (ASM) parts. In this new work, that method's implementation as an AFR prototype is outlined and the differences between the prototype and the original proposal are pointed out. Then, streamlined data structures are described and explained. They are used to organize the B-rep elements extracted from the ASM parts' STEP models, classify and enhance them, and structure the features recognized from the STEP models. Next, a few examples of the algorithms that are implemented in the prototype to manipulate the B-rep elements and recognize features are represented and explained. The details of the algorithms are presented in the appendices. To validate the AFR method and verify its correct implementation, a collection of 26 real-world ASM parts was used to create CAD models that were subsequently converted to STEP models. The STEP models were processed to recognize their features, and the results show perfect accuracy. A few of the output feature files are presented in detail. Our results confirm great potential for further AFR method development for rather specialized domains of application.

5.2 Introduction

Many commercial CAD solutions provide feature-based CAD tools to model generic sheet metal parts. CATIA (Dassault Systems; V5-R2016, 2016) and NX 12 for Design (Siemens PLM Software, 2017a) are two solutions that provide features that are specialized for and commonly used in aerospace sheet metal (ASM) part design, including curved flanges, joggles, and stringer cutouts, to name a few. The ASM models produced by commercial CAD solutions are not exchanged in their native format, but rather via their boundary representation (B-rep) using Standard for the Exchange of Product (STEP) model data (Pratt, 2001). However, the specialized features cannot be exchanged via STEP. An ASM feature-recognition method for STEP models—the B-rep model—could therefore significantly elevate the level of design information exchanged via STEP. This research project aims to propose an automated method for recognizing features from ASM B-rep models. Our overarching goal is to elevate the of information available for downstream applications such as 3D model difference identification (Ghaffarishahri & Rivest, 2021). This way, it becomes possible to indicate to the user, for example, how the length of a flange differs between two models.

Automated feature recognition (AFR) is an established way to abstract semantics from neutral CAD models. It dates back to the 1980s and has been investigated extensively since then (Bojan Babic et al., 2008). AFR can be approached by rule-based methods (like syntactic pattern recognition, state transition diagrams and automata, logic rules and expert systems, the graph-based approach, the convex hull volumetric decomposition approach, the cell-based volumetric decomposition approach, the hint-based approach and the hybrid approach) or artificial neural network methods (like the graph-based approach, face coding, the contour-syntactic approach and volume decomposition) (Bojan Babic et al., 2008). It seems that in the last decade AFR-related works have focused more on its application (Gupta & Gurumoorthy, 2012; Lai, Wang, Song, Hsu, & Tsai, 2018; Langerak, 2010; Y. G. Li et al., 2009; Jing Wang & Zhou, 2018; Zubair & Mansor, 2018), optimization (B. C. Kim & Mun, 2015; Šormaz & Tennety, 2010; V. Sunil, R. Agarwal, & S. Pande, 2010; Venu, Komma, & Srivastava, 2018)

and implementation (Venu et al., 2018; Q. Wang & Yu, 2014) than on proposing new methods. A rather inconspicuous application of AFR is sheet metal parts.

Although there is relatively little literature about AFR specific to sheet metal parts, there are quite a few noticeable and inspiring works (Devarajan et al., 1997; Gupta & Gurumoorthy, 2012, 2013; Jagirdar et al., 2001; Jagirdar et al., 1995; Kannan & Shunmugam, 2009a, 2009b; Z. Liu et al., 2004; Nnaji et al., 1991; Sunil & Pande, 2008; Chunjie Zhang et al., 2009). While a few sheet metal AFR methods take a comprehensive approach, such as the one proposed by Nnaji et al. (Nnaji et al., 1991), most can be categorized as being for shear features, generic deformation features or freeform sheet metal parts. Because freeform AFR methods (Gupta & Gurumoorthy, 2012; Sunil & Pande, 2008; Chunjie Zhang et al., 2009) are not relevant to the scope of this paper, the previous works on them are not reviewed in detail here. The sheet metal AFR solutions that are currently on the market, i.e., FeatureWorks (Dassault Systems, 2019) and Sheet Metal Feature Recognition Library (HCL Technologies, 2013), are implementations of AFR methods for generic sheet metal deformation features. Our proposed AFR method (Ghaffarishahri & Rivest, 2020) is unique as it was specialized for aerospace sheet metal part features.

Shear features are sheet metal part features that are created by shearing operations like blanking, notching, piercing and cutting off. AFR methods for shear features can be based on geometric and topological reasoning, like in the work of Jagirdar et al. (Jagirdar et al., 1995); profile offsetting for layout punching tool path specifications, like in the work of Devarajan et al. (Devarajan et al., 1997); or using a center-plane model to calculate the shear layout, like in the work of Kannan and Shunmugam (Kannan & Shunmugam, 2009b).

Sheet metal generic deformation features, on the other hand, result from either only deformation operations or shear and deformation operations. Hence, there is always a deformation footprint in their structures. Although the AFR methods proposed for sheet metal part deformation features are essentially conventional rule-based AFR methods, they have

evolved independently. For example, Liu et al. (Z. Liu et al., 2004) presented a fundamental study that addressed some of the main gaps in prior works, including feature intersection and array features. Feature intersection occurs when features intersect such that one is split or topologically impacted. An array feature is made up of repeated features. Kannan and Shunmugam (Kannan & Shunmugam, 2009a, 2009b) also made two significant contributions to sheet metal AFR: 1) creating the method on STEP AP 203 that led to the promotion of its applicability, and 2) proposing and using a calculation algorithm for a center-plane model. The outcome of their proposal to use a center-plane model was twofold: it proved that a) a reduction in topographical information improved computational performance, and b) said reduction did not cripple the method. Prior to Kannan and Shunmugam, Jagirdar et al. (Jagirdar et al., 2001) assumed that a sheet metal model is represented by its center-plane equivalent and accordingly proposed an AFR method. They considered “cross-bend” features (features that pass through a bend) as the only cases of feature intersection in sheet metal parts and proposed a technique to identify intersecting features. Subsequent studies omitted the problem of intersecting features entirely, making Jagirdar et al.’s work quite unique. And most recently, Gupta and Gurumoorthy (Gupta & Gurumoorthy, 2013) proposed a general solution for recognizing generic deformation features. In their paper, cylindrical, conical, spherical and toroidal faces are considered transitive entities to characterize deformation. This method is more geometrically general than the other methods set out in published works.

We proposed in (Ghaffarishahri & Rivest, 2020) an AFR method for ASM parts that was inspired by the recent focus on AFR application in the literature. Features were identified by studying design guidelines and 168 different structural ASM fuselage and cockpit parts. A structural system is comprised of a thin-skinned shell that is stiffened by longitudinal stringers that are supported by transverse frames to form a semi-monocoque structure (Niu, 1999). This structure is very efficient and has a high strength-to-weight ratio. Brake-formed and hydro-formed parts—the parts studied to propose the AFR method—are used to form frames, bulkheads, passenger and cargo floor structures (Niu, 1999), and cockpit components.

Implementation of the proposed AFR method is not detailed in the literature. Instead, only the programming language or solution, CAD platform, or kernel used in the implementation is mentioned. The C++ programming language seems to be most popular because of its strong object-oriented programming (OOP) package and the fact that it is the industry standard for developing 3D modeling solutions, which makes it possible to use it with many CAD platforms (Lai et al., 2018; Z. Liu et al., 2004) and graphics kernels (B. C. Kim & Mun, 2015; V. Sunil et al., 2010; Sunil & Pande, 2008; Chunjie Zhang et al., 2009). Table 5.1 lists the programming languages or solutions, CAD platforms, and kernels that have been used to implement AFR methods.

Table 5.1 AFR implementation means used in the literature

AFR implementation means	References
C language	(Jagirdar et al., 2001; Jagirdar et al., 1995)
C++ language and ACIS™ kernel	(B. C. Kim & Mun, 2015; V. Sunil et al., 2010)
C++ language and Rhino CAD platform (and openNURBS functions)	(Lai et al., 2018)
C++ language and based on UGNX2.0 platform	(Z. Liu et al., 2004)
C++ language and OpenGL kernel	(Sunil & Pande, 2008)
C++ language and Open CASCADE kernel	(Chunjie Zhang et al., 2009)
Java language	(Šormaz & Tennety, 2010; Venu et al., 2018; Q. Wang & Yu, 2014)
MATLAB	(Gupta & Gurumoorthy, 2012, 2013)

In this paper, we describe the implementation and prototyping of our AFR method that we previously proposed for ASM parts in (Ghaffarishahri & Rivest, 2020). Some improvements

were made to the original method, and the feature taxonomy was changed to match the one detailed in a subsequent paper proposing a semantic model difference identification method (Ghaffarishahri & Rivest, 2021). We provide details we believe will be helpful for similar future endeavors. In addition, the prototype is used to validate our AFR method by testing it with real-world samples.

5.3 Definitions and methodology

This section includes a short review of the definitions of the terms migrated from our earlier publications and illustrations and explanations of the 26 sample parts selected to validate the AFR method. These 26 parts cover all the features of all the parts studied in this research and are of various complexity levels. The main steps of the AFR method that were modified during implementation are briefly reviewed. Next, the data structure of the input STEP files, the intermediate enhanced B-rep elements manipulated to recognize features and the features are represented and explained. At the end, a few examples of the algorithms used to implement the main steps of the AFR method are explained in detail and represented in flowcharts.

5.3.1 Definitions

First, the definitions of the subtypes of the B-rep elements that were introduced in the original paper (Ghaffarishahri & Rivest, 2020) are listed. Next, the definition of the features recognized by the original method as well as the features later modified in the work proposing the MDI method (Ghaffarishahri & Rivest, 2021) are listed.

In the original method, it was detailed that faces are classified by their subtypes: `trim_face`, `sheet_face`, `bend_face`, `internal_face`, `wall_face`, `connect_face` and `detained_face` (Ghaffarishahri & Rivest, 2020).

1. The `trim_faces` are related to the faces of an ASM part that are created by trimming, and one of their dimensions is always equal to the part thickness.
2. The `sheet_faces` are related to the faces of an ASM part that are not being trimmed, and they constitute the two sides of an ASM part.

The sheet_faces are primarily classified as web_faces, bend_faces, wall_faces, detained_faces and internal_faces.

1. The planar sheet_faces with the two largest surface areas are called web_faces.
2. The sheet_faces connecting web_faces and wall_faces via their face_outer_bounds are called bend_faces.
3. The wall_faces are the sheet_faces that are connected to each other or to web_faces by bend_faces.
4. The connect_faces are sheet_faces connecting bend_faces.
5. The sheet_faces that are adjacent to web_faces and wall_faces via their non-face_outer_bounds are called internal_faces.
6. The detained_faces are intermediate sheet_faces that are reclassified as connect_faces or wall_faces in the process of classifying sheet_faces by their subtypes.

In addition, edges are classified by their novel subtypes: trim_lin_edges, trim_nonlin_edges, untrim_lin_edges and untrim_nonlin_edges.

1. A trim_lin_edge is an edge that is shared between any subtype of sheet_faces and a trim_face and is associated with a line.
2. A trim_nonlin_edge is an edge that is shared between any subtype of sheet_faces and a trim_face and is associated with any subtype of curve other than a line.
3. An untrim_lin_edge is an edge that is not shared with any trim_face and is associated with a line.
4. An untrim_nonlin_edge is an edge that is not shared with any trim_face and is associated with any subtype of curve other than a line.

The face_bounds are also classified by their subtypes: face_outer_bound, bead_bound, internal_bound and hole_bound. The face_outer_bound are obtained from B-rep models.

1. A face_bound that surrounds seven internal_faces is a bead_bound.

2. A `face_bound` that is not a `face_outer_bound` and consists of edges associated with distinguishable curves is an `internal_bound`.
3. A `face_bound` that consists of edges associated to undistinguishable curves is a `hole_bound`.

The features considered in this paper are web, cutout, hole, stringer cutout, bend relief, corner, lightening cutout, lightening hole, flange, lip, joggle, twin joggle, deformed flange, deformed web and bead.

1. A web is the planar portion of an ASM part with the largest surface area.
2. Cutouts are features formed by removing a portion of their parent feature, provided that the boundary of the parent feature is not changed.
3. Holes are circular cutouts.
4. Stringer cutouts are features formed by modifying the boundary of the parent feature and splitting flanges or the deformed flanges resulting from a twin joggle. Stringer cutouts are created to make room for a stringer to pass through.
5. Bend reliefs are cutouts created to avoid immediate adjacency between flanges and facilitate manufacturing.
6. Corners are features formed by rounding off sharp convex corners.
7. Lightening cutouts and lightening holes are features created by removing a portion of the parent feature and forming stiffening lips at the boundary of the removed portion.
8. Flanges are features materialized on the external boundary of their parent feature and are always the child of the web or another flange.
9. Lips are features that are similar in shape to combined-open-immediate-stiffening flanges.
10. Joggles and twin joggles are step-deformations that produce recesses on the web or flanges.
11. The recessed portions are either deformed webs or deformed flanges depending on the parent feature of the joggle.
12. Beads are features created by protruding a portion of the web.

The original taxonomy of the ASM part features was from a manufacturing perspective (Ghaffarishahri & Rivest, 2020). It was restructured to represent features' functionality, and a few features were also added (Ghaffarishahri & Rivest, 2021). Holes were divided into attachment holes and tooling holes, and flanges are divided into stiffening flanges and attachment flanges.

1. Tooling holes are features created to facilitate manufacturing.
2. Attachment holes are features created for attachment purposes. They are designed based on different design tables.
3. Attachment flanges are flanges that have attachment holes.
4. Stiffening flanges are flanges that do not have attachment holes.

5.3.2 3D models of sample parts

The sample ASM parts were chosen from a pool of parts belonging to frames, bulkheads, the floor and the cockpit of a Bombardier DHC-8-102. The parts were selected to satisfy the following criteria:

1. Presence of all design features of ASM parts.
2. Presence of a broad spectrum of design complexity.
3. Presence of adequately similar designs to test output consistency.

The native models were stored in STEP AP 242 files. Figures 5.1, 5.2 and 5.3 show the CAD models of the sample ASM parts.

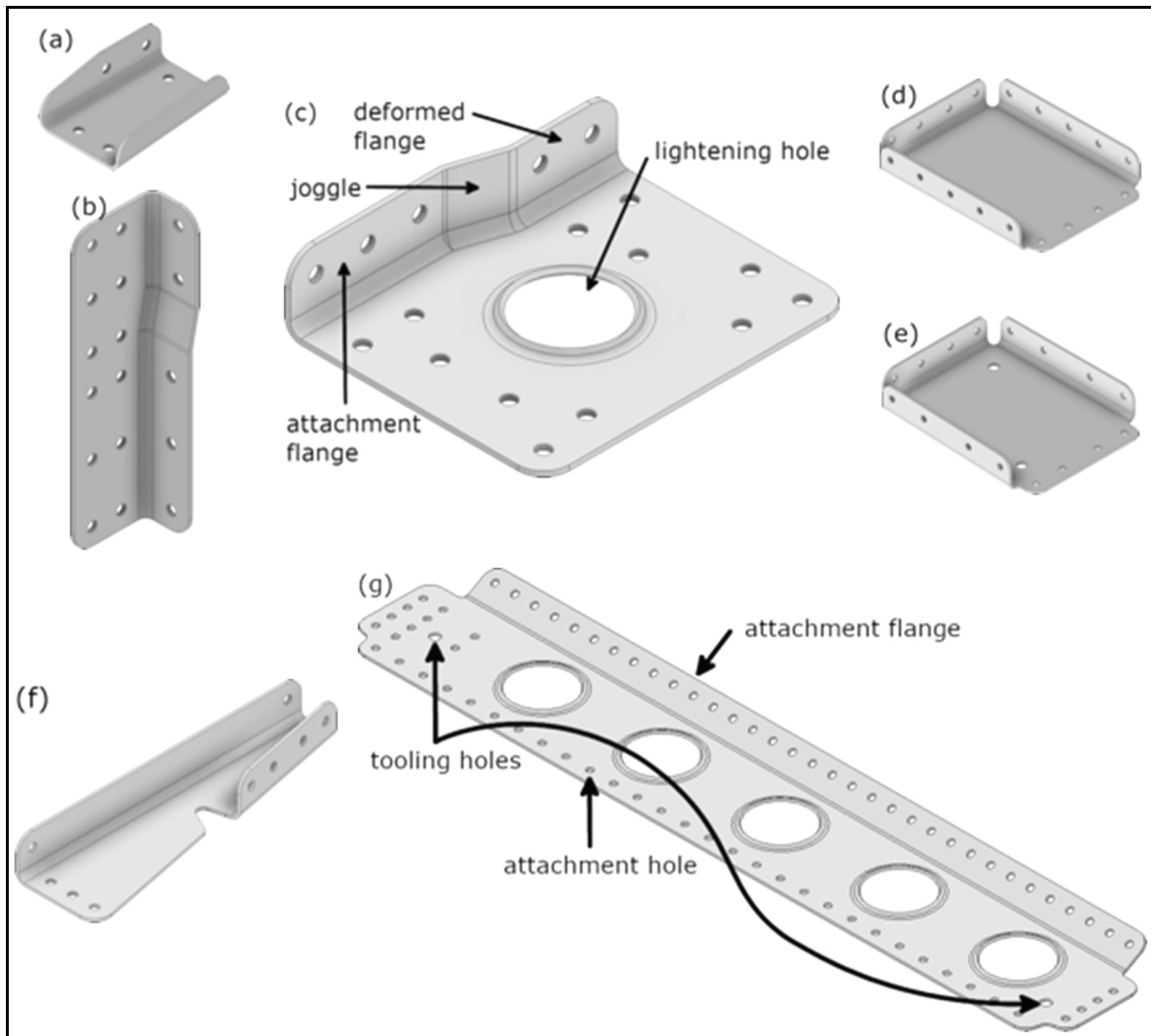


Figure 5.1 CAD models of the sample aerospace sheet metal parts (a, b, c, d, e, f, g)

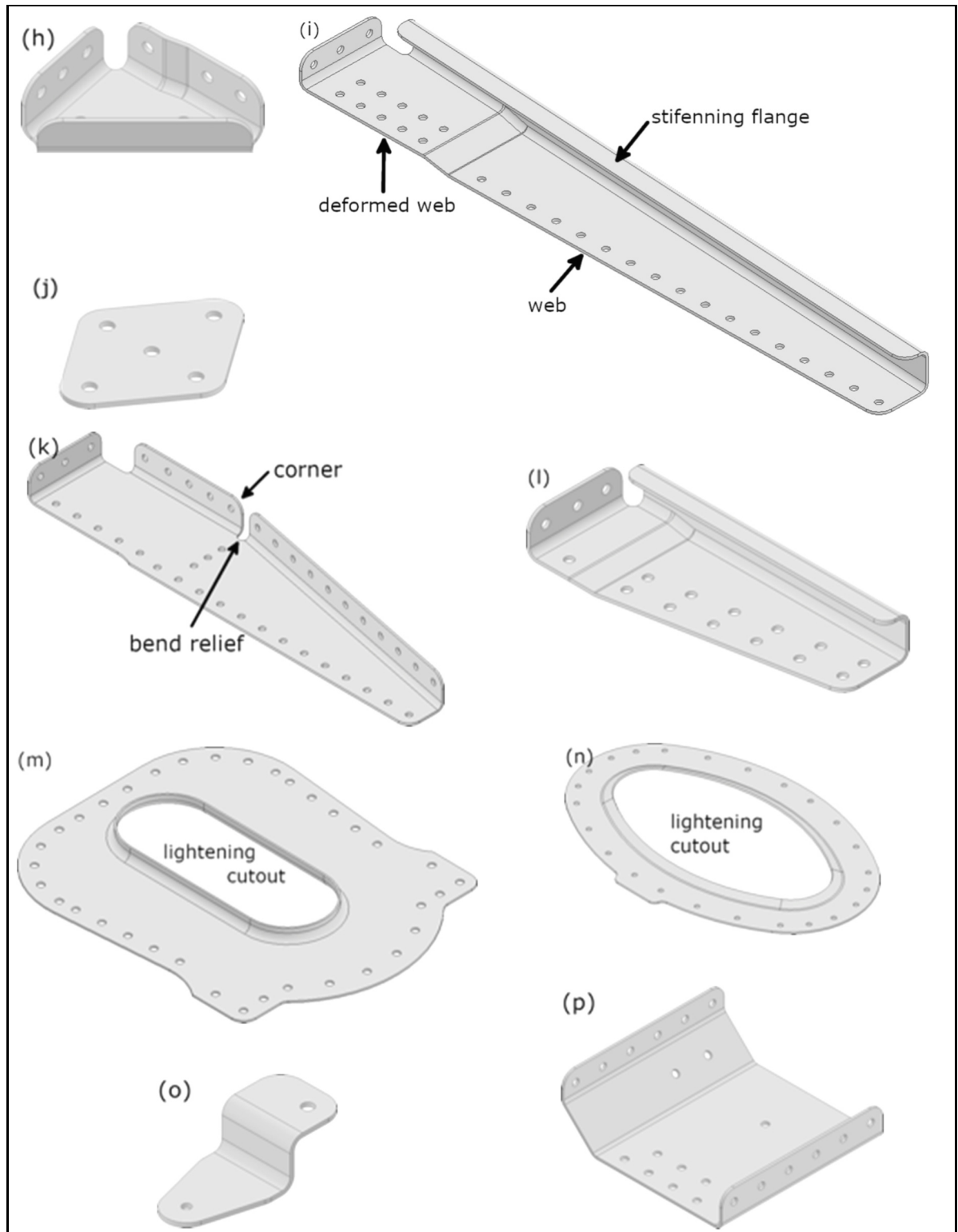


Figure 5.2 CAD models of the sample aerospace sheet metal parts (h, i, j, k, l, m, n, o, p)

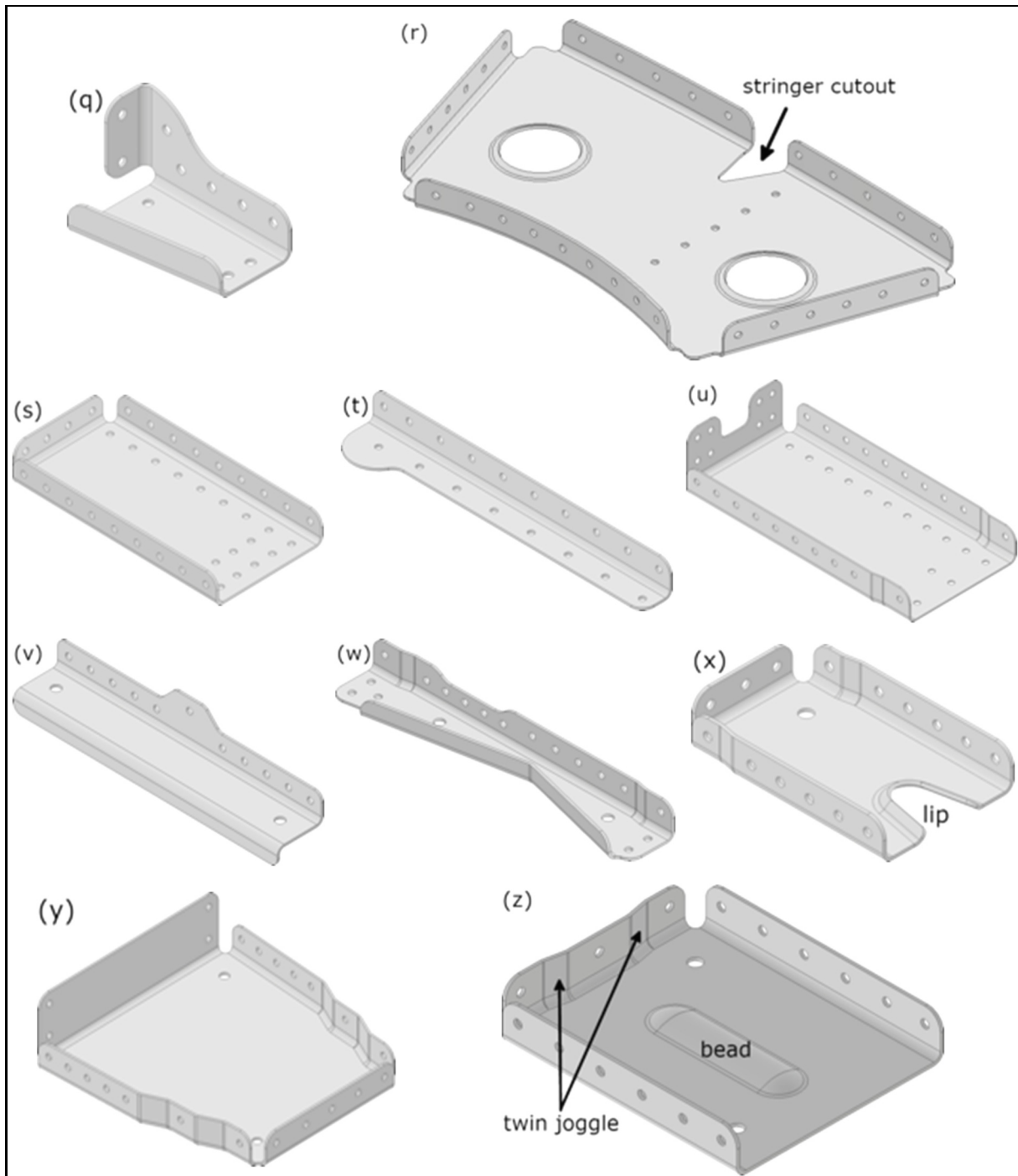


Figure 5.3 CAD models of the sample aerospace sheet metal parts (q, r, s, t, u, v, w, x, y, z)

5.3.3 Modified steps of the AFR method

The original AFR method was explained extensively in our previous paper (Ghaffarishahri & Rivest, 2020). Some modifications were made when implementing it in the prototype to either facilitate implementation or fix minor problems with the original method. Figure 5.4 highlights the major Steps 1 and 2 that were described in our previous paper and displays the steps implemented in the prototype with the modifications highlighted and denoted by the letters A to L. Also, each step whose algorithm is detailed in this paper is tagged with the number of the figure or letter of the appendix in which the algorithm can be found.

- 5.a. The B-rep elements are streamed from the STEP file and stored as C++ objects, which are themselves stored in lists.
- 5.b. To take advantage of the abstraction and encapsulation capabilities of object-oriented programming, the C++ objects storing the B-rep elements are enhanced by adding the geometric information to the topologic elements and directly connecting it to various B-rep elements via C++ pointers.
- 5.c. The `web_faces` are classified before `sheet_faces` and `trim_faces` are. This makes sense, as `web_faces` are `sheet_faces`.
- 5.d. The `bend_faces` and `internal_faces` adjacent to the `web_faces` are classified in one step.
- 5.e. Parts that do not have flanges, like those modeled in Figure 5.2 (j), (m) and (n), have all their `sheet_faces` classified by subtype (`web_face` and `internal_face`) by now and do not require step F. For those parts that have flanges and therefore have `sheet_faces` that have yet to be classified, step F is repeated until all their `sheet_faces` are classified by subtype.
- 5.f. The remaining `sheet_face` classification by subtype (`wall_faces`, `detained_faced`, `bend_faces` or `internal_faces`) continues until all `sheet_faces` are classified.
- 5.g. When classifying `face_bounds`, if the geometry of the non-lin edges is not identical, the `hole_bounds` are identified as `internal_bounds`. This change prevents incorrectly recognizing lightening cutouts like the one shown in Figure 5.2 (n) as lightening holes.

- 5.h. In the scope of this study, all joggles have faces that are adjacent to the web or a deformed web; therefore, joggles, deformed webs, flanges and deformed flanges are all recognized in one step performed after recognizing the web.
- 5.i. Although the stringer cutout recognition process is unchanged, bend reliefs are recognized between the web, deformed webs, flanges and deformed flanges, and are limited to having one non-lin edge between two of the above features.
- 5.j. A feature's ID and parent features are defined when it is recognized, and feature objects are instantiated accordingly. Its child features are also defined to complete its feature associations, and its parameters are calculated and instantiated to complete its definition.
- 5.k. Lips are features that have many similarities with flanges, so they are initially recognized as immediate-stiffening-open flanges. If any of said flanges have parameters that meet the standards defining lips, they are converted to lips.
- 5.l. In the last step, the flanges and holes are evaluated to classify them as attachment or stiffening flanges and attachment or tooling holes, respectively. Although these subtypes did not exist in the feature taxonomy of the original method, they are implemented to pave the way for the model difference identification applications intended for the output of this prototype. Attachment flanges and attachment holes carry design intentions that are pivotal for the semantic comparison of ASM parts.

At the end, the features are stored in a user-readable text file that we call a feature file. A couple of examples of feature files are presented in the results section.

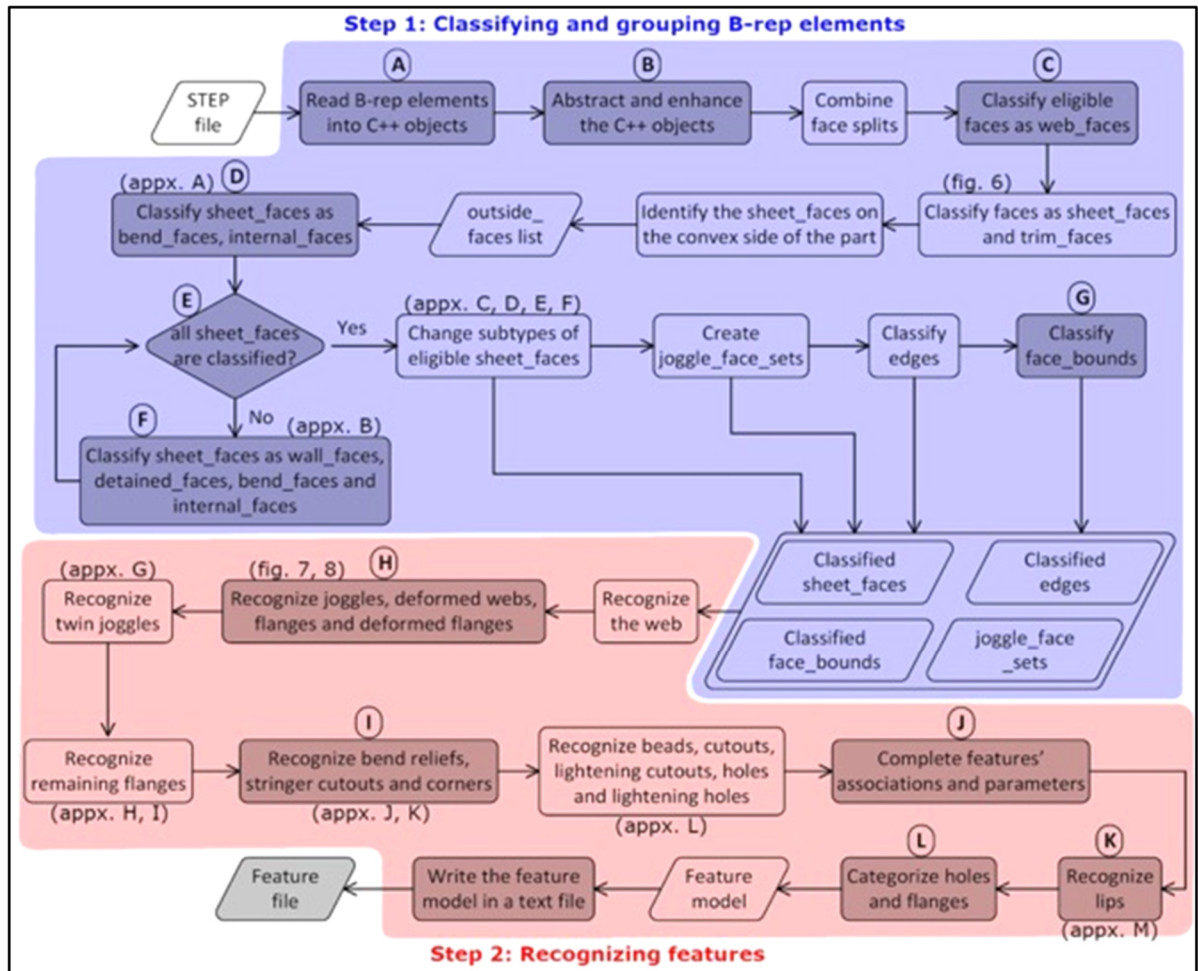


Figure 5.4 Flowchart of the modified method implemented in the prototype

5.3.4 Data structures

When the B-rep elements are read from the STEP files, they are stored in the memory as C++ objects that have identical member attributes as the B-rep elements. These C++ objects are stored in lists implemented by C++ Vectors provided by the C++ Standard Template Library (STL). A vector is a sequence container that stores elements. C++ vectors are specifically used to work with dynamic data and can expand depending on the elements they contain. That makes them different from a fixed-size array. C++ vectors can automatically manage storage. Also, they are efficient when data is added and deleted often.

Figure 5.5 shows the class diagram representing the B-rep elements of the STEP files and their attributes and associations. Implementing the method based on the data structure of the STEP files' B-rep elements causes two issues: it is not possible to take full advantage of C++ class member methods, and objects must be repeatedly searched for.

To avoid these issues, the following changes were made to the data structure of the STEP files' B-rep elements:

1. The content of CartesianPoints, VertexPoints, Directions, Vectors, Axis2Placement3Ds, EdgeCurves and LoopEdges were added as member attributes of their associated B-rep elements.
2. B-splineCurveWithKnots, Circles, Ellipses and Lines were linked to OrientedEdges by raw pointers.
3. B-splineSurfaceWithKnots, BoundedB-splineSurfaces, Planes, CylindricalSurfaces, ToroidalSurfaces, SphericalSurfaces and ConicalSurfaces were linked to AdvancedFaces by raw pointers.
4. Both FaceOuterBounds and FaceBounds were stored as FaceBounds; however, FaceOuterBounds were signified by a true Boolean member attribute.
5. Faces, Bounds and Edges classes were added to manage AdvancedFaces, FaceBounds and OrientedEdges, respectively.

Figure 5.6 shows the class diagram representing the further abstracted and enhanced data structure. This solution allows operations related to Edges, Bounds and Faces to be carried out via the classes method, which is more compliant with the abstraction and encapsulation basis of OOP. Another reason for this solution is to manage memory issues related to the use of raw pointers. When two objects (A and C) are pointing to another object (B), if object A is deleted (for example, local variable no longer in use), object B will also be deleted, and object C will have what is called a "dangling pointer" because it is pointing to a memory address (object B)

that is no longer valid. This can cause several problems such as false results and lost data. C++ makes it possible to manage the lifetime of objects and change their default behavior.

When using raw pointers, it is best practice to allocate the memory dynamically using the "new" operator, and explicitly release the memory using "delete" and delete the object when the object is no longer in use. However, in the data structure proposed in this work, AdvancedFaces, OrientedEdges and FaceBounds are shared between different objects. For example, a pointer to a FaceBound is shared between an OrientedEdge and an AdvancedFace, so it is crucial to manage the memory and decide when each object should be deleted.

Creating classes to represent the edges_list, bounds_list and faces_list makes it easier to manage objects' lifetime, since in C++, each object is created and deleted using special methods (class member functions) that are called constructors and destructors, respectively. Each class has a default constructor and destructor, so customizing these two methods makes it easy to manage the memory.

Figure 5.7 shows the class diagram representing the features and their structure as well as their parameters (like position or profile) that are implemented in C++ classes. In the proposed feature data structure, all the features are inherited from a "Feature" class. This makes it possible to use one code to define the behavior that is common to all the features. The "Feature" class is defined as an abstract class. An abstract class is a class that cannot be instantiated; it is only used as a base class in inheritance hierarchies. C++ abstract classes are defined by pure virtual functions. A pure virtual function is a virtual function that has no implementation. For example, the following feature_type function (the Feature class's member function) is a pure virtual function:

Virtual std::string feature_type() const = 0;

This virtual function returns the feature type, for example, "Flange" or "Joggle", and doesn't have an implementation in the base class, since the base class "Feature" is an abstract class.

There is no "feature" feature type, so it makes sense that the feature_type function is not implemented for the Feature class. Since, the Feature class serves as a base class from which all the other features are derived, it defines the behavior that is common to all the features.

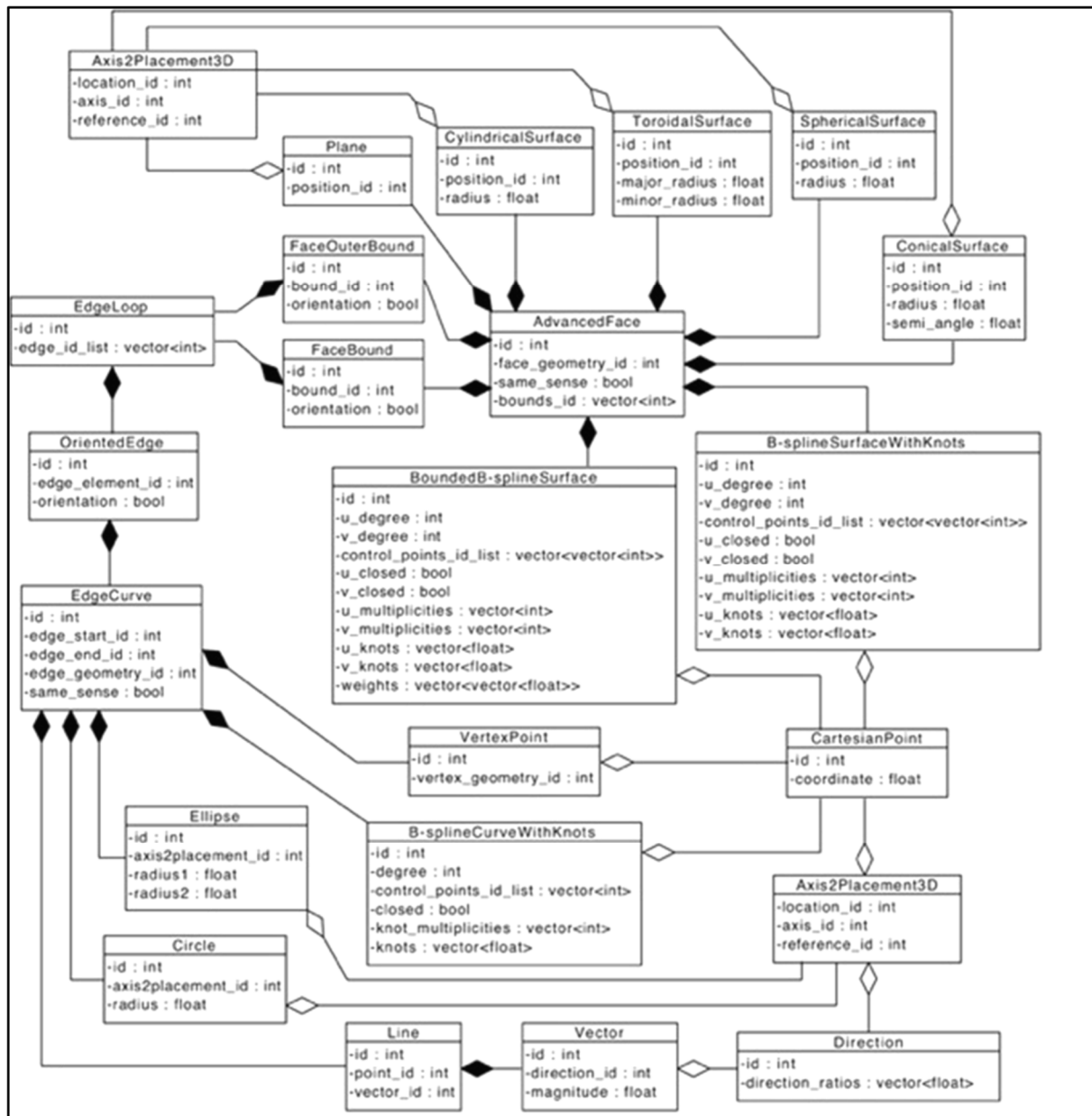


Figure 5.5 Class diagram representing the B-rep elements of the STEP files

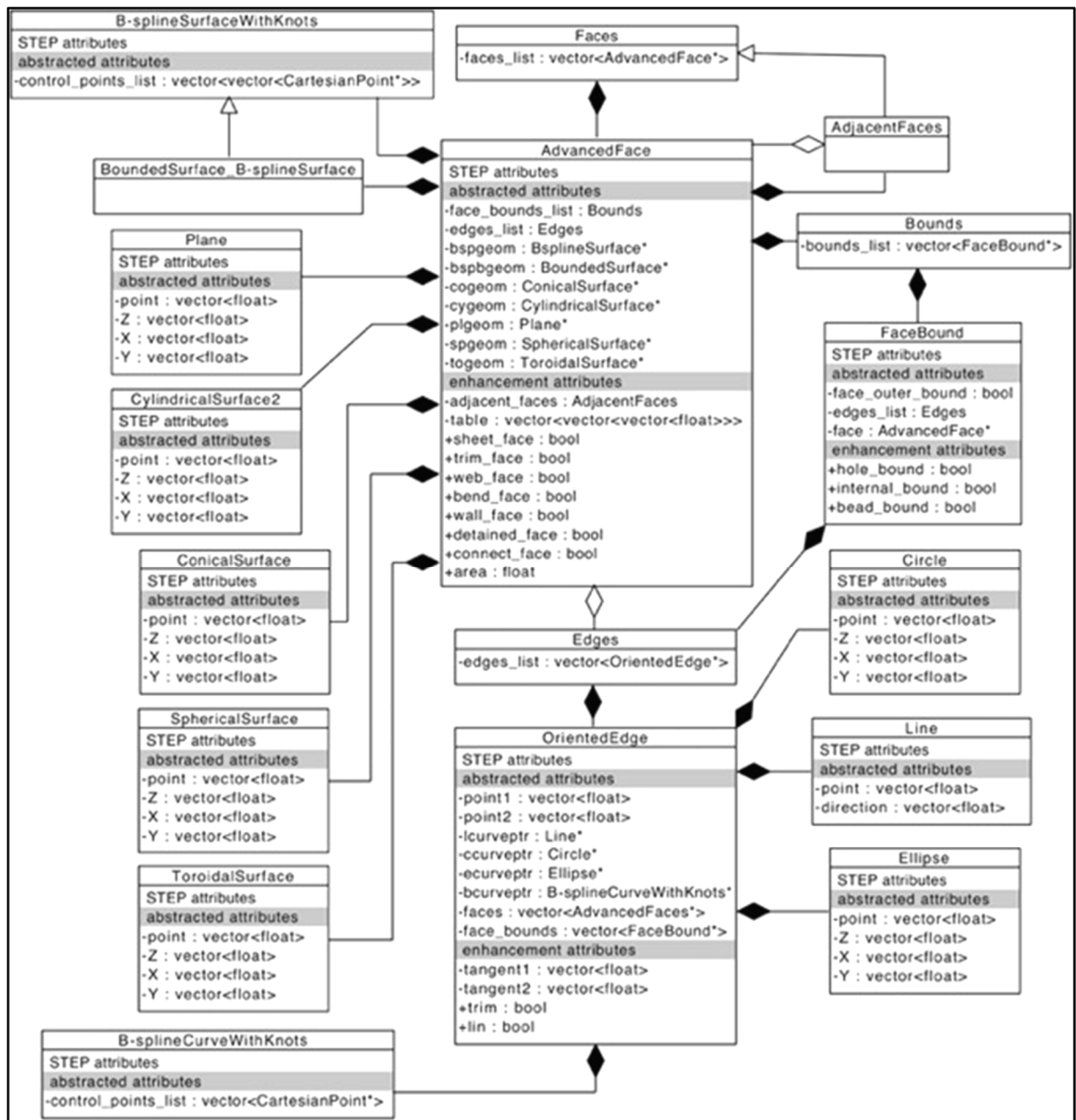


Figure 5.6 Class diagram representing the further abstracted and enhanced data structure

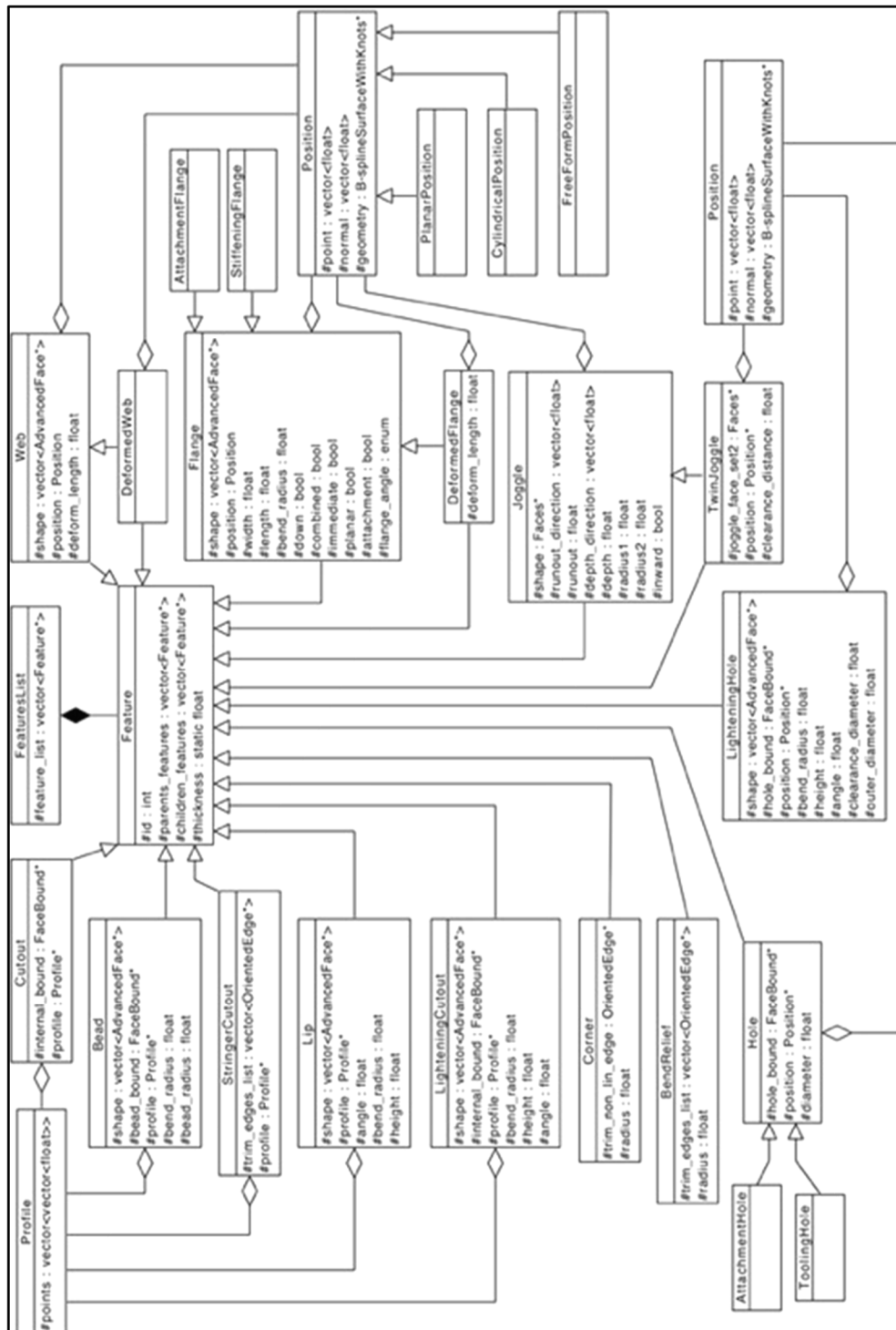


Figure 5.7 Class diagram representing the features and their data structure

5.3.5 The algorithms

Here, we describe the main ideas in the algorithms designed to perform each step in classifying faces and recognizing features. The algorithms to classify boundaries and edges are not discussed due to their simplicity. The algorithm to identify `sheet_faces` and `trim_faces` is described in detail and illustrated in Figure 5.8. The algorithm to recognize joggles on the web, on deformed webs and on flanges is described in detail and illustrated in Figures 5.9 and 5.10. In the case of joggles on flanges, the algorithm also recognizes their related flanges and deformed flanges. The rest of the algorithms for further classifying faces and recognizing the other types of flanges are listed in Appendices I through XII.

Identifying `sheet_faces` and `trim_faces` starts from one of the `web_faces`. The `web_faces` are the planar faces of the B-rep model with the largest surface area. Each `web_face` is stored in a list called `output1` or `output2`, both of which are instances of the `Faces` class. First, the normal vectors of the `web_face`, which is initially the only face in the list, and each of its adjacent faces at a shared point are evaluated for codirectionality. If the normal vectors of a face and an adjacent face at a shared point are not codirectional, the adjacent face is classified as a `trim_face` and the next adjacent face is evaluated. If, however, the normal vectors are codirectional and the adjacent face is not already in the list of faces, it is classified as a `sheet_face` and added to the list. The `sheet_face` and `trim_face` identification function is recursive. As a result, if a face is classified as a `sheet_face`, before moving on to the next adjacent face, its own adjacent faces are evaluated for codirectionality between their normal vectors at a shared point. Figure 5.8 shows a detailed representation of the `sheet_face` and `trim_face` identification algorithm.

Most of the feature recognition algorithms start from a list of faces that are evaluated to check a few conditions and conclude whether or not a feature exists. These lists of faces could be the list of faces on the outside of the part (called `outside_faces`), `joggle_face_sets`, or faces linked to the shape of features. The `outside_faces` could be `output1` or `output2`, depending on the

summation of faces' surface area of which one is larger than the other one. For example, the algorithm to recognize joggles and their related parent or child features (illustrated in Figure 5.9) starts with a `joggle_face_set`. (The `joggle_face_sets` and their identification are detailed in (Ghaffarishahri & Rivest, 2020).) Then, the `bend_faces` in the `joggle_face_set` are checked to find out if any of them is adjacent to the `web_face`, to assign true to a Boolean variable called `cond1`. The `connect_faces` in the `joggle_face_set` are also checked to find out if any of them is adjacent to the `web_face`, to assign false to a Boolean variable called `cond2`. If `cond1` is true and `cond2` is false, the joggle must be on the web, and if both `cond1` and `cond2` are true, the joggle must be on a flange.

1. If the joggle is on the web, the joggle is created with its associated `sheet_faces`. Then, each of the `wall_faces` in the `outside_faces` list is checked to find the one that is adjacent to one of the `bend_faces` and none of the `connect_faces` in the `joggle_face_set`. When found, the deformed web is created with its associated `sheet_faces`.
2. If the joggle is on a flange, the joggle is created with its associated `sheet_faces`. Then, two temporary flanges are created. Note that this part of the algorithm is represented quite abstractly in the highlighted steps in Figure 5.9 to save some space and then detailed in a zoomed-in view in Figure 5.10. Next, the deformed temporary flange is identified and the deformed flange is created based on the temporary flange.

To recognize the temporary flanges, the algorithm represented in Figure 5.10 first checks the `sheet_faces` in the `joggle_face_set` to find the `bend_faces`. Then, the `bend_faces` are checked to find their adjacent `wall_faces` that are not in the `joggle_face_set` and not adjacent to any of the `connect_faces` in the `joggle_face_set`. When any of those `wall_faces` are found, a flange is created and then its faces are added to it. Note that the parent-child relationships between features are identified based on the relationships between the topological elements used in recognizing them. For example, a deformed web is linked to its parent joggle based on the adjacency of the `wall_face` used in its recognition and the `bend_face` of the parent joggle. Similarly, a hole is linked to a flange or web based on the topological relationships between the `hole_bound` and `wall_face` or `web_face`.

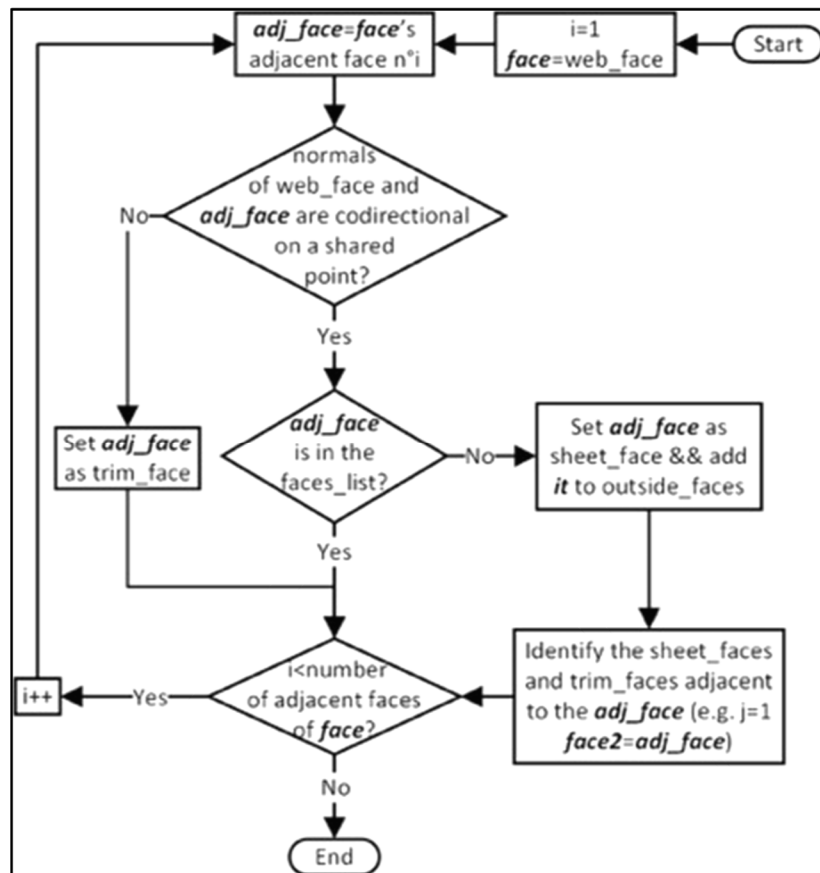


Figure 5.8 Flowchart representing the algorithm to identify sheet_faces and trim_faces

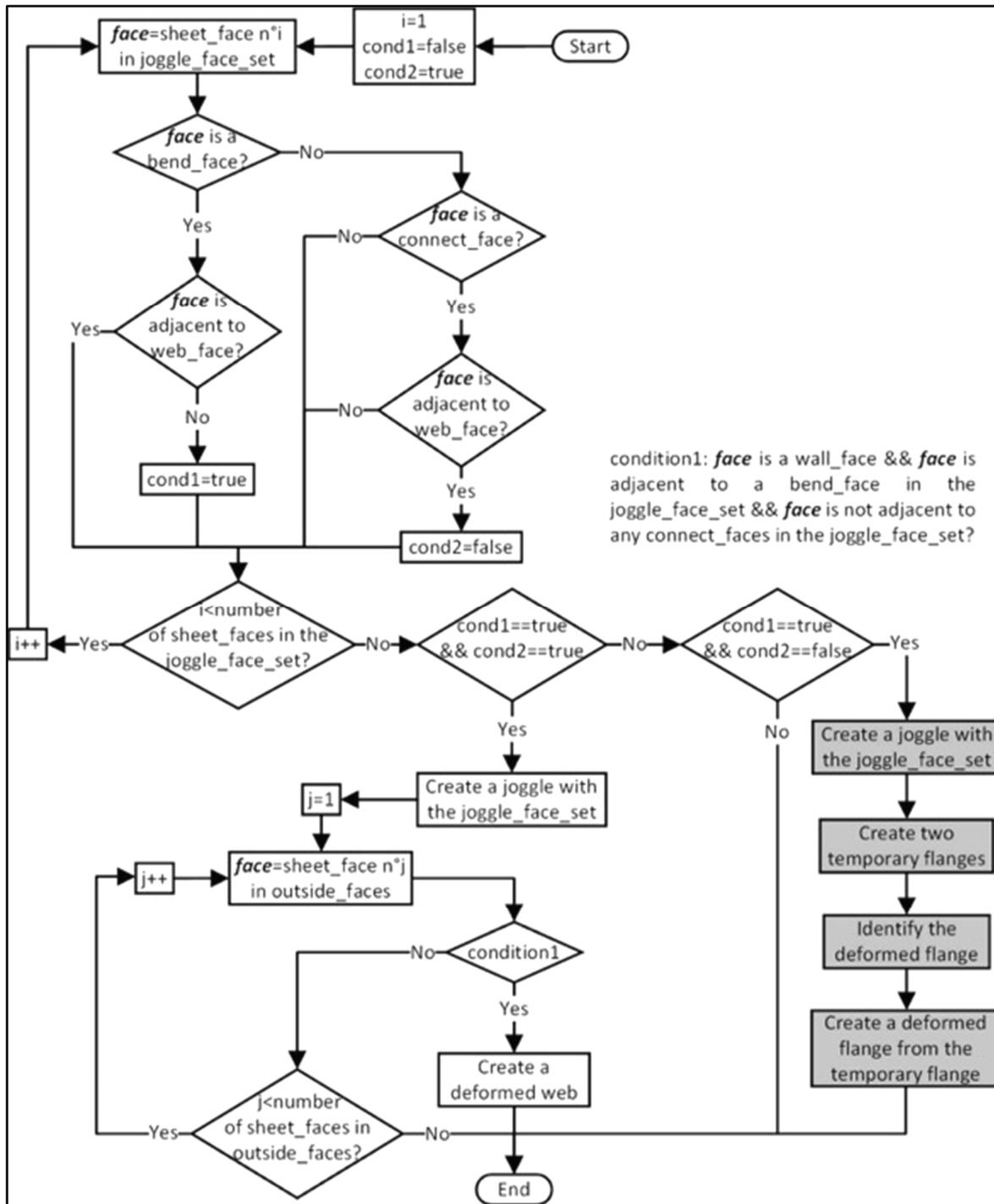


Figure 5.9 Flowchart representing the algorithm to recognize some joggles and flanges
Joggles on the web, on deformed webs and on flanges, as well as the flanges and deformed
flanges related to joggles on flanges

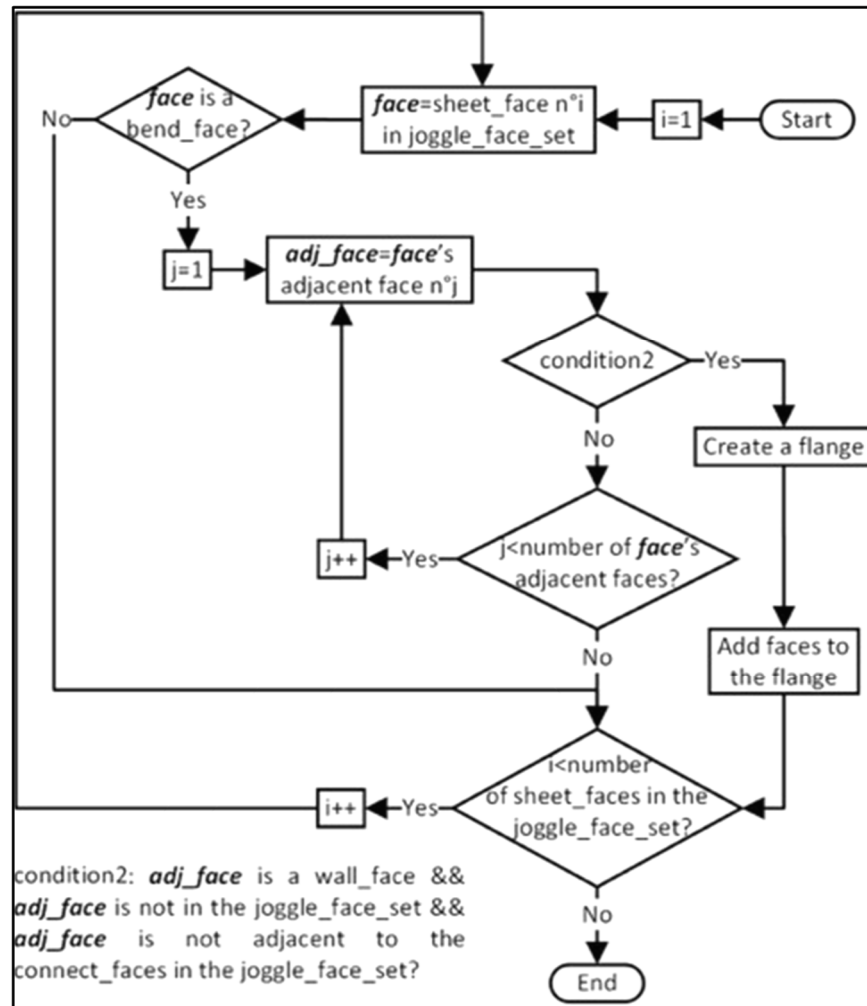


Figure 5.10 Flowchart representing the algorithm to create joggles and temporary flanges and recognize the deformed flange from the temporary flanges
 (A zoomed-in view of the highlighted steps in Figure 5.9)

5.4 Results

The output of the prototype is a text file that is called a feature file. The feature file starts with the name of the part, followed by its thickness and then its features. The features are structured by parent features immediately followed by their children and are further indented as their level in the feature hierarchy increases. Each feature is defined by its name, ID, Parent ID (except

the web) and parameters. For example, Figure 5.11 shows the content of the feature file for the part represented in Figure 5.1 (a).

```

Part name: Val1
Part thickness is: 1.5 mm
web (ID: 1)
  stiffening flange (ID: 2; Parent ID: 1; Width: 60.21 mm; Length: 12.5 mm; Bend radius: 3 mm; Type: Down,
Single, Immediate, Planar, Perpendicular; It has a position)
    corner (ID: 4; Parent ID: 2; Radius: 7.5 mm)
    corner (ID: 5; Parent ID: 2; Radius: 7.5 mm)
  attachment flange (ID: 3; Parent ID: 1; Width: 60 mm; Length: 17.5 mm; Bend radius: 3 mm; Type: Down,
Single, Immediate, Planar, Perpendicular; It has a position)
    corner (ID: 6; Parent ID: 3; Radius: 7.5 mm)
    corner (ID: 7; Parent ID: 3; Radius: 7.5 mm)
    corner (ID: 8; Parent ID: 3; Radius: 7.5 mm)
    attachment hole (ID: 12; Parent ID: 3; Diameter: 4 mm; It has a position)
    attachment hole (ID: 13; Parent ID: 3; Diameter: 4 mm; It has a position)
  attachment hole (ID: 9; Parent ID: 1; Diameter: 4 mm; It has a position)
  attachment hole (ID: 10; Parent ID: 1; Diameter: 4 mm; It has a position)
  attachment hole (ID: 11; Parent ID: 1; Diameter: 4 mm; It has a position)

```

Figure 5.11 The content of the feature file for the part represented in Figure 5.1 (a)

Note that if a feature needs a position or profile to be defined and the position or profile is successfully obtained from the B-rep model, “It has a position” or “It has a profile” holds the place of the position or profile information. Otherwise, “It does NOT have a position” or “It does NOT have a profile” indicates there is no position or profile information for the feature. The position and profile information is quite low-level data, described in the class model represented in Figure 5.7, and therefore not included in the content of feature files, which are supposed to be user readable.

We have selected three parts to represent the prototype’s results. Each of these parts has a number of features, some of which highlight interesting points about the prototype’s output and are therefore explained. In Figures 5.12, 5.13 and 5.14, each part is displayed annotated with its features and followed by its feature file content.

To start with, Figure 5.12 (a) shows a rather typical design of a hydroformed ASM part that is annotated with its features. It has a lightening hole, attachment holes on the web, and an

immediate flange deformed by a joggle with several attachment holes on the attachment flange and deformed flange. Figure 5.12 (b) shows the content of the part's feature file with “~ ~ ~” being used to avoid a long list of features having the same parameters.

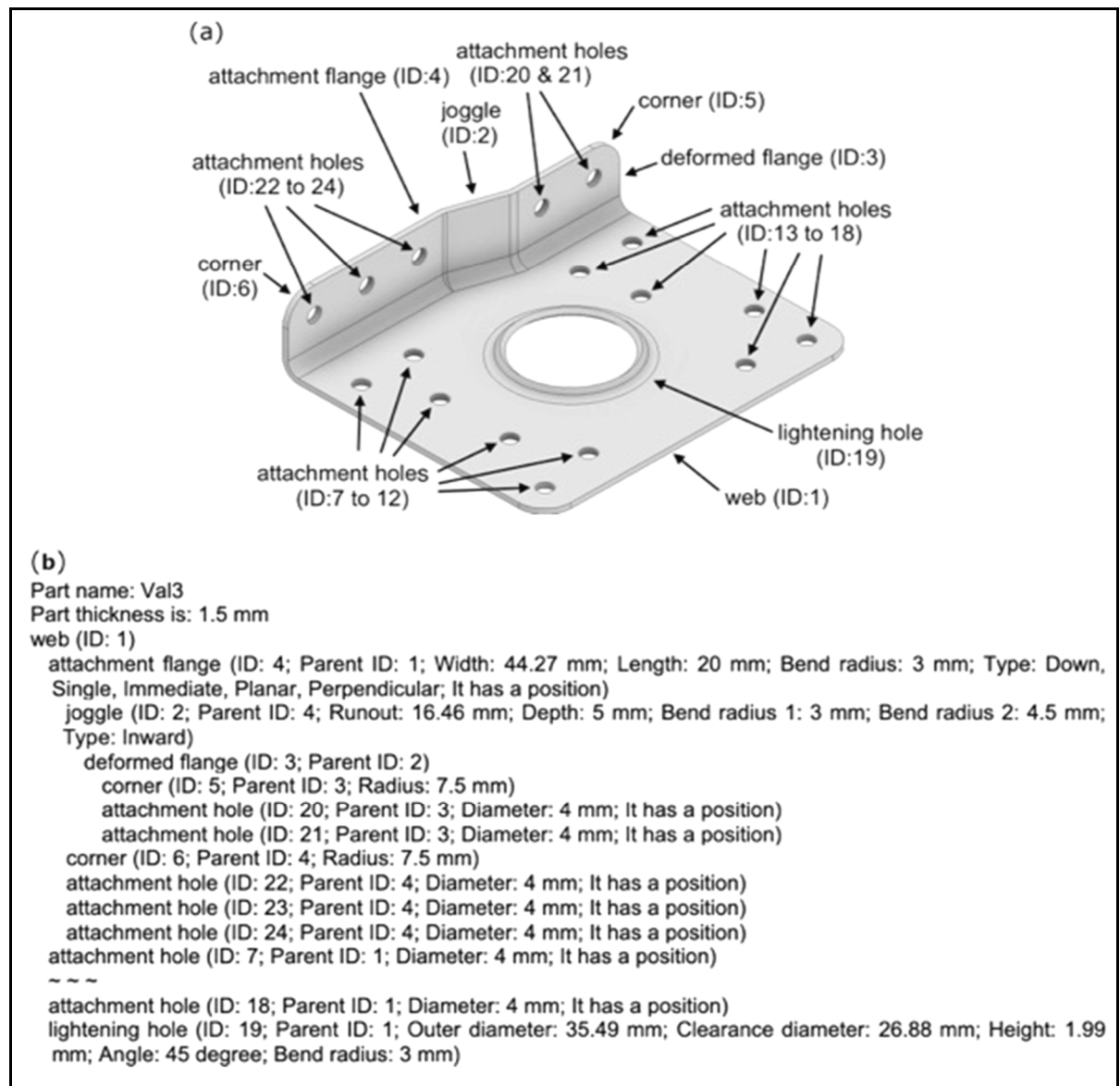


Figure 5.12 A rather typical design of a hydroformed aerospace sheet metal part annotated with its features (a) as well as the content of its feature file (b)

Figure 5.13 (a) shows another rather common design of a hydroformed ASM part that is annotated with its features. It has tooling and attachment holes on the web, and immediate flanges with several attachment holes on them. Due to the arrangement of the flanges, there are bend reliefs between them. The corners are not annotated in Figure 5.13 (a) for simplicity. Figure 5.13 (b) shows the content of the part's feature file with “~ ~ ~” being used to avoid a long list of features having the same parameters.

Figure 5.14 (a) shows a less common, more complex design of a hydroformed ASM part that is annotated with its features. It has attachment holes on the web, which is deformed by a joggle, and an immediate flange with several attachment holes and a stiffening flange on it. Due to the arrangement of the immediate flanges on the web and the deformed web, there is a bend relief between them. The corners are not annotated in Figure 5.14 (a) for simplicity. Figure 5.14 (b) shows the content of the part's feature file with “~ ~ ~” being used to avoid a long list of features having the same parameters.

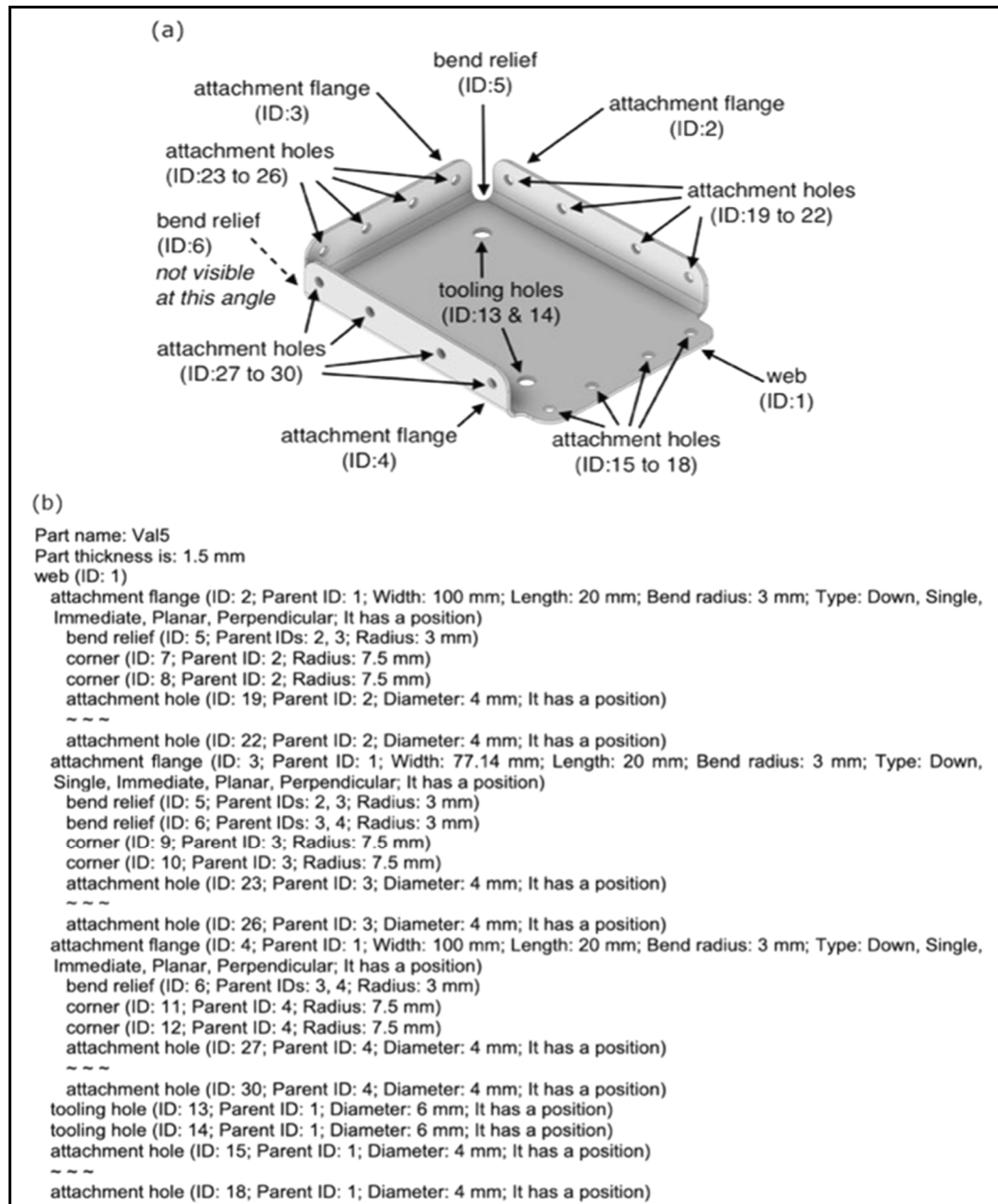


Figure 5.13 A rather common design of a hydroformed aerospace sheet metal part annotated with its features (a) as well as the content of its feature file (b)

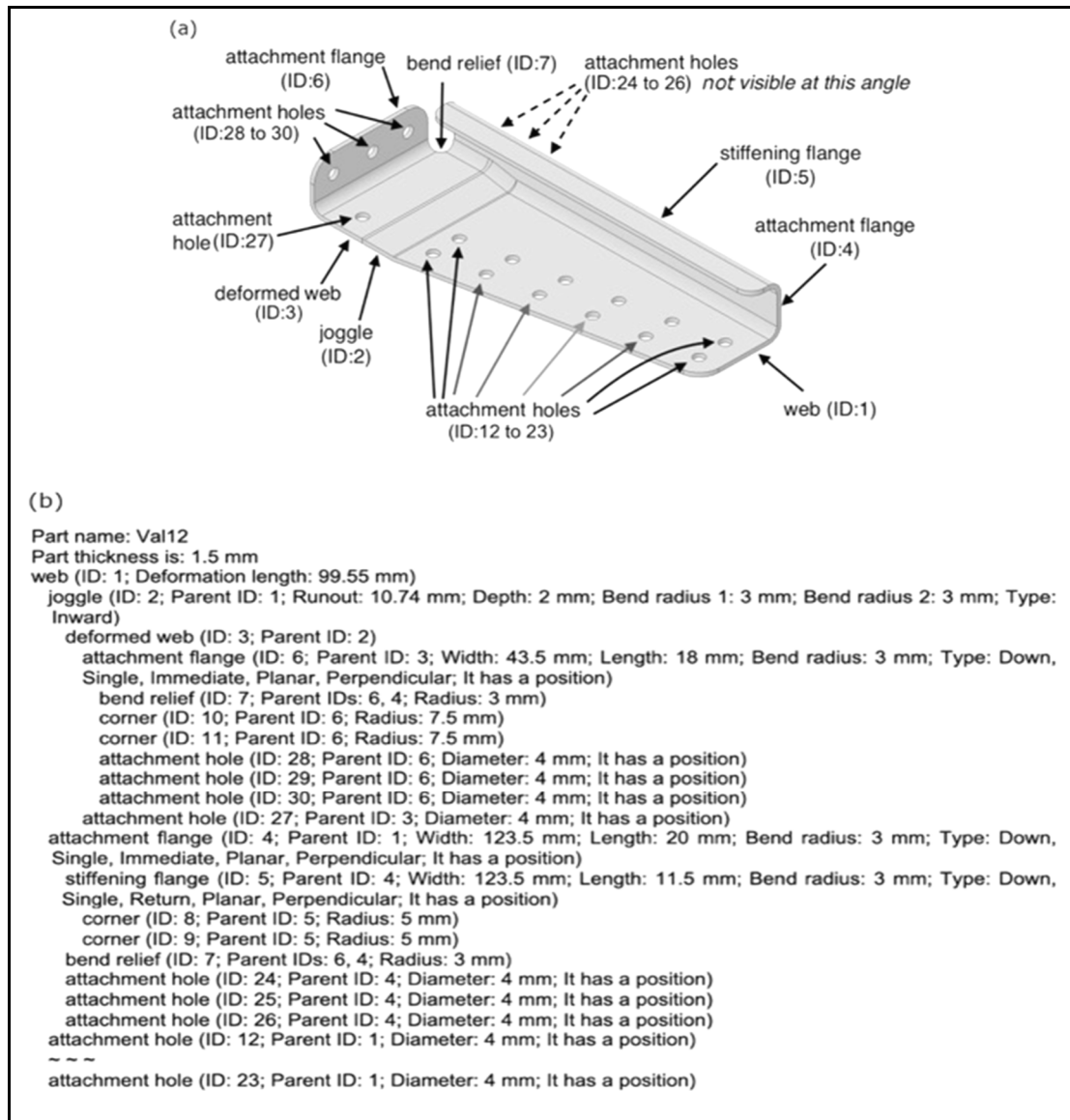


Figure 5.14 A less common design of a hydroformed aerospace sheet metal part annotated with its features (a) as well as the content of its feature file (b)

5.5 Discussion and conclusion

A factor that made prototyping challenging was that the original proposed method was for a set of unprecedented features, ASM part features. This made the development work unique and hence necessitated creativity in solving problems. Of all the ASM part features, joggle was the main differentiator between the original method and similar literature. It was also pivotal

to the development effort. The uniqueness of the original method's scope and by extension the development effort motivated the authors to present this work at a level of detail not seen in the existing literature.

In this work, we represented the successful implementation of an AFR method for ASM parts. The AFR method was implemented by first developing a data structure for the B-rep elements and a data structure for feature definitions. Then, algorithms were developed for classifying B-rep elements and recognizing features. When implementing the AFR method, certain steps in the originally proposed method (Ghaffarishahri & Rivest, 2020) were changed to correct errors or make improvements. A collection of 26 parts was modeled and converted to STEP models to validate the AFR method and verify the accuracy of the AFR prototype.

The results from the AFR prototype show perfect accuracy in recognizing all the features of all 26 parts and confirm there is great potential for further development of AFR algorithms in rather specialized domains of application. Although feature recognition from B-rep models has been investigated for decades, it has not become an integral part of CAD solutions. The authors of this paper believe that works on AFR have been too general to be feasible or accurate enough for commercial solutions. A rather specialized approach like our originally proposed method whose implementation is represented in this paper makes it possible to break down the AFR problem and propose accurate specialized solutions.

CHAPTER 6

RESULTS FROM THE MDI PROTOTYPE

In this chapter, we present the overall testing procedure used to evaluate the MDI prototype and the prototype's output for three test cases.

6.1 Testing procedure

The MDI prototype was developed in C++ using the same data structure as was used for the AFR prototype. The MDI prototype reads the feature structure and definitions of the ASM parts to be compared from their *feature files*. After the parts have been compared, the differences are represented in a text file called a *difference file*.

A modified AFR prototype (with respect to the one presented in Chapter 5) was developed to store all the feature information needed for MDI in the feature files. The original AFR prototype that was presented in the previous chapter was developed to store user-readable and engineeringly meaningful information in the feature file. For example, the original AFR prototype would write “It has a position” in the definition of an attachment hole if it has a valid position, while the modified prototype writes the coordinates of the origin point of the position and the normal axis elements. Figures 6.1 and 6.2 display an ASM part (a) and its feature files to be user-readable (b) and MDI-ready (c).

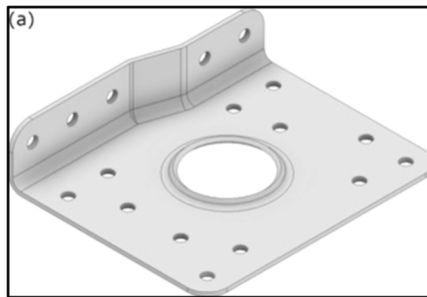


Figure 6.1 A 3D model
representation of an ASM part (a)

(b)

```

Part name: a_ref
Part thickness is: 1.5 mm

web (ID: 1)
  attachment flange (ID: 4; Parent ID: 1; Width: 44.27 mm; Length: 20 mm; Bend radius: 3 mm; Type: Down, Single,
Immediate, Planar, Perpendicular; It has a position)
  joggle (ID: 2; Parent ID: 4; Runout: 16.46 mm; Depth: 5 mm; Bend radius 1: 3 mm; Bend radius 2: 4.5 mm; Type: Inward)
  deformed flange (ID: 3; Parent ID: 2)
    corner (ID: 5; Parent ID: 3; Radius: 7.5 mm)
      attachment hole (ID: 20; Parent ID: 3; Diameter: 4 mm; It has a position)
      attachment hole (ID: 21; Parent ID: 3; Diameter: 4 mm; It has a position)
    corner (ID: 6; Parent ID: 4; Radius: 7.5 mm)
      attachment hole (ID: 22; Parent ID: 4; Diameter: 4 mm; It has a position)
      attachment hole (ID: 23; Parent ID: 4; Diameter: 4 mm; It has a position)
      attachment hole (ID: 24; Parent ID: 4; Diameter: 4 mm; It has a position)
  attachment hole (ID: 7; Parent ID: 1; Diameter: 4 mm; It has a position)
  attachment hole (ID: 8; Parent ID: 1; Diameter: 4 mm; It has a position)
  attachment hole (ID: 9; Parent ID: 1; Diameter: 4 mm; It has a position)
  attachment hole (ID: 10; Parent ID: 1; Diameter: 4 mm; It has a position)
  attachment hole (ID: 11; Parent ID: 1; Diameter: 4 mm; It has a position)
  attachment hole (ID: 12; Parent ID: 1; Diameter: 4 mm; It has a position)
  attachment hole (ID: 13; Parent ID: 1; Diameter: 4 mm; It has a position)
  attachment hole (ID: 14; Parent ID: 1; Diameter: 4 mm; It has a position)
  attachment hole (ID: 15; Parent ID: 1; Diameter: 4 mm; It has a position)
  attachment hole (ID: 16; Parent ID: 1; Diameter: 4 mm; It has a position)
  attachment hole (ID: 17; Parent ID: 1; Diameter: 4 mm; It has a position)
  attachment hole (ID: 18; Parent ID: 1; Diameter: 4 mm; It has a position)
  lightening hole (ID: 19; Parent ID: 1; Outer diameter: 35.49 mm; Clearance diameter: 26.88 mm; Height: 1.99 mm; Angle:
45 degree; Bend radius: 3 mm)

```

(c)

```

Part name: a_ref
Part thickness is: 1.5 mm

web (ID: 1; Position point:(0,0,0); Position normal:(0,1,0))
  attachment flange (ID: 4; Parent ID: 1; Width: 44.27 mm; Length: 20 mm; Bend radius: 3 mm; Type: Down, Single, Immediate,
Planar, Perpendicular; Position point:(-37.5,0,-45); Position normal:(-1,0,0))
  joggle (ID: 2; Parent ID: 4; Runout: 16.46 mm; Runout Direction:(0,0,-1); Depth: 5 mm; Depth Direction:(1,-0,-0); Bend
radius 1: 3 mm; Bend radius 2: 4.5 mm; Type: Inward)
  deformed flange (ID: 3; Parent ID: 2)
    corner (ID: 5; Parent ID: 3; Radius: 7.5 mm)
      attachment hole (ID: 20; Parent ID: 3; Diameter: 4 mm; Position point:(-32.5,12.5,-37.5); Position normal:(1,0,0))
      attachment hole (ID: 21; Parent ID: 3; Diameter: 4 mm; Position point:(-32.5,12.5,-22.5); Position normal:(1,0,0))
    corner (ID: 6; Parent ID: 4; Radius: 7.5 mm)
      attachment hole (ID: 22; Parent ID: 4; Diameter: 4 mm; Position point:(-37.5,12.5,22.5); Position normal:(1,0,0))
      attachment hole (ID: 23; Parent ID: 4; Diameter: 4 mm; Position point:(-37.5,12.5,37.5); Position normal:(1,0,0))
      attachment hole (ID: 24; Parent ID: 4; Diameter: 4 mm; Position point:(-37.5,12.5,7.5); Position normal:(1,0,0))
  attachment hole (ID: 7; Parent ID: 1; Diameter: 4 mm; Position point:(30,0,-37.5); Position normal:(0,1,0))
  attachment hole (ID: 8; Parent ID: 1; Diameter: 4 mm; Position point:(15,0,-37.5); Position normal:(0,1,0))
  attachment hole (ID: 9; Parent ID: 1; Diameter: 4 mm; Position point:(-20,0,-37.5); Position normal:(0,1,0))
  attachment hole (ID: 10; Parent ID: 1; Diameter: 4 mm; Position point:(27.5,0,22.5); Position normal:(0,1,0))
  attachment hole (ID: 11; Parent ID: 1; Diameter: 4 mm; Position point:(12.5,0,30); Position normal:(0,1,0))
  attachment hole (ID: 12; Parent ID: 1; Diameter: 4 mm; Position point:(-22.5,0,37.5); Position normal:(0,1,0))
  attachment hole (ID: 13; Parent ID: 1; Diameter: 4 mm; Position point:(-22.5,0,22.5); Position normal:(0,1,0))
  attachment hole (ID: 14; Parent ID: 1; Diameter: 4 mm; Position point:(-7.5,0,30); Position normal:(0,1,0))
  attachment hole (ID: 15; Parent ID: 1; Diameter: 4 mm; Position point:(30,0,37.5); Position normal:(0,1,0))
  attachment hole (ID: 16; Parent ID: 1; Diameter: 4 mm; Position point:(-20,0,-22.5); Position normal:(0,1,0))
  attachment hole (ID: 17; Parent ID: 1; Diameter: 4 mm; Position point:(-5,0,-25); Position normal:(0,1,0))
  attachment hole (ID: 18; Parent ID: 1; Diameter: 4 mm; Position point:(27.5,0,-22.5); Position normal:(0,1,0))
  lightening hole (ID: 19; Parent ID: 1; Outer diameter: 35.49 mm; Clearance diameter: 26.88 mm; Height: 1.99 mm; Angle: 45
degree; Bend radius: 3 mm; Position point:(-0,0,0); Position normal:(0,-1,0))

```

Figure 6.2 Two other representations of an ASM part

(b) its feature file to be user-readable and (c) its feature file to be MDI-ready

The CAD model comparison method consists of two major portions: pose registration and feature comparison. The method was extensively described in Chapter 4. We focus here on

why there may be differences between two parts, and how these reasons led to modifying the originally proposed comparison method.

Although the differences between two parts could theoretically occur arbitrarily, in practice, there are specific design-driven reasons for the significant differences that can be of help in designing practical tests for the MDI prototype. Significant differences between compared parts could be the result of:

1. Differences in the design of the parts that the compared parts are in contact with, which could cause the attachment holes and the position of the attachment flanges, deformed flanges and deformed webs of the compared parts to be different.
2. Differences between the contact features of the compared parts, perhaps as a result of the differences described above, which could cause the joggles, twin joggles, corners, bend reliefs, stiffening flanges and beads of the compared parts to be different.
3. Differences between the lightening holes, cutouts, lightening cutouts, lips, and stringer cutouts of the compared parts as a result of the differences mentioned above (1 and 2), as these features' definitions could depend on flanges, deformed flanges, joggles, twin joggles etc.
4. Differences between other parts of the structures that the compared parts are used in – other than the ones that the compared parts are directly in contact with – which could cause the lightening holes, cutouts, lightening cutouts, lips, and stringer cutouts to be different.
5. Differences that are manufacturing-driven, like differences affecting the tooling holes.

The three examples that were provided in the previous chapter to represent the output of the AFR prototype in detail are modified in this chapter to create the difference identification scenarios. These three examples were selected because they could be changed in such a way that all the above-mentioned design-driven reasons for significant differences are encompassed. Certainly, the more examples that are used to test the prototype, the better; however, we opted to use the minimum number of examples to validate the MDI method. Figure 6.3 shows the part models used and a relatively similar real ASM part for each.

1. The 3D model of the part represented in Figure 6.3 (a) was modified by increasing the depth of the joggle from 5 mm to 7.5 mm to displace the deformed flange and increasing the diameter of the attachment holes on the flange and deformed flange from 4 mm to 5 mm, which increased the length of the flange and the deformed flange from 20 mm to 22 mm. It is expected that increasing the diameter of the attachment holes on the attachment flange and the deformed flange will increase the length of the flange. Because the radius of the corners is a product of the diameter of the attachment holes, they will also increase.
2. The 3D model of the part represented in Figure 6.3 (c) was modified by increasing the radius of the bend reliefs from 3 mm to 4 mm and changing one of the attachment flanges from perpendicular to closed with a 95° angle. The change in the bend relief radius alters the width of the flange that has a bend relief on both sides. If a flange keeps the same width, it must shift, and the position of its child attachment holes therefore becomes different. Also, the change in the angle of the one attachment flange results in its length and position being different. These differences may cause the tooling holes, attachment holes or corners on the web to be different due to displacement.
3. The 3D model of the part represented in Figure 6.3 (e) was modified by increasing the diameter of the attachment holes on the immediate attachment flanges from 4 mm to 5 mm, which increases the length of the flange from 20 mm to 24 mm and consequently displaces the return flange.

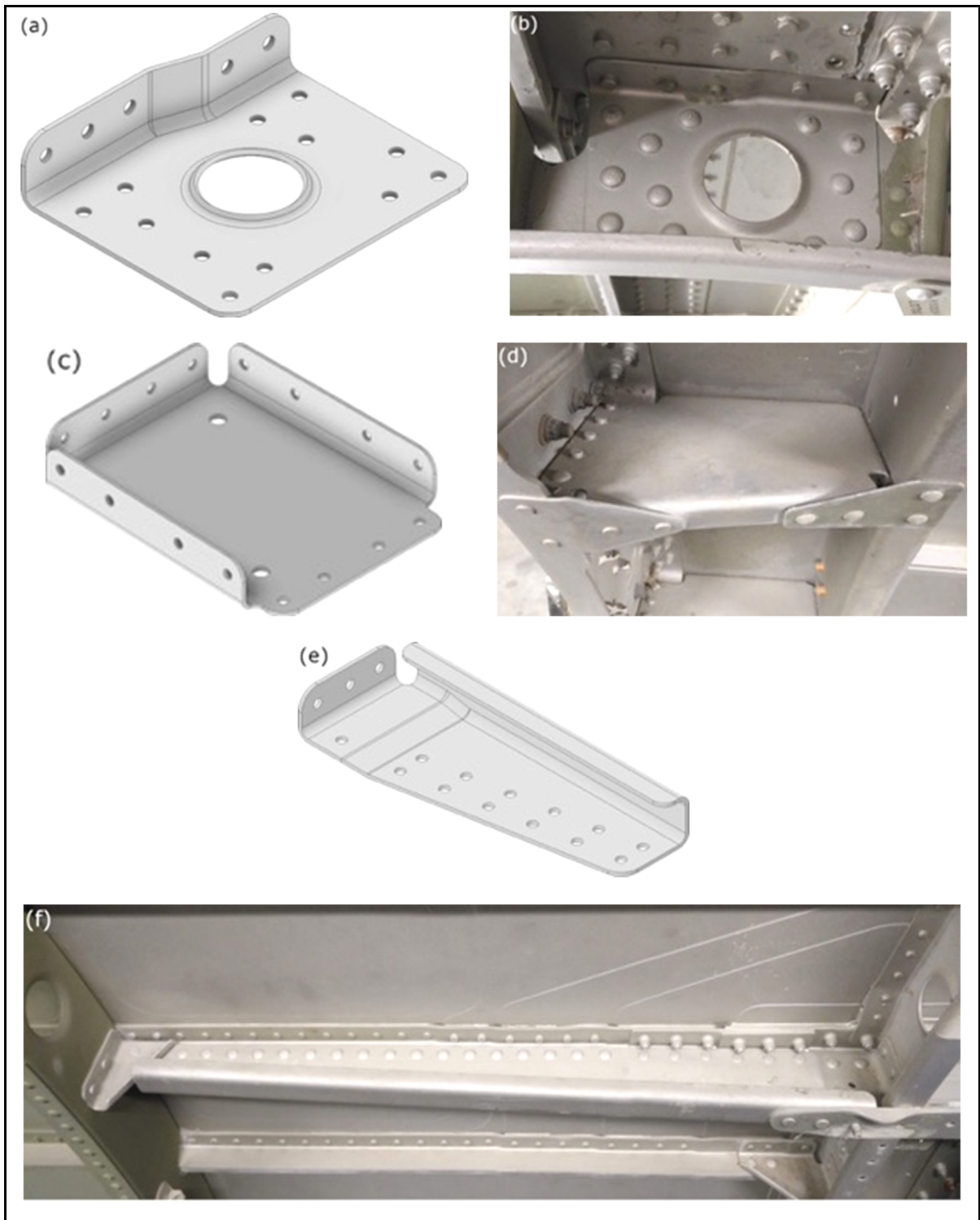


Figure 6.3 The part models used to test the MDI prototype and a relatively similar real ASM part for each

6.2 Results

This section presents the results obtained from the MDI prototype for all three test cases mentioned above. Figure 6.4 shows the MDI prototype's output for the part represented in Figure 6.3 (a) and its modified version. Figure 6.4 (a) shows the color-coded results of the volumetric comparison to illustrate the differences and guide the reader. The commonalities are in yellow, the areas specific to the original part are in red, and the areas specific to the modified version are in green. We can interpret from that image that: the web, the lightening hole and attachment holes on the web are the same for both parts; the location of the flange is the same for both parts, while the length of the flanges differs; the deformed flange and the rest of the features differ too. Figure 6.4 (b) shows the MDI prototype's output, which describes the differences between the parts at the feature level.

As expected, increasing the diameter of the attachment holes on the attachment flange and the deformed flange increases the length of the flanges and thereby makes it differed. The radius of the corners is a product of the diameter of the attachment holes and increases as well, which makes the corners differed too. The difference in the depth of the joggles is also recognized and reported correctly. It is worth noting that although the deformed flanges are in completely different positions, which is apparent in Figure 6.4 (a), because their position is not a descriptive parameter, they are identified as being the same (commonality). This exemplifies one way that our proposed MDI method is superior – when it comes to representing the semantic differences between the compared models.

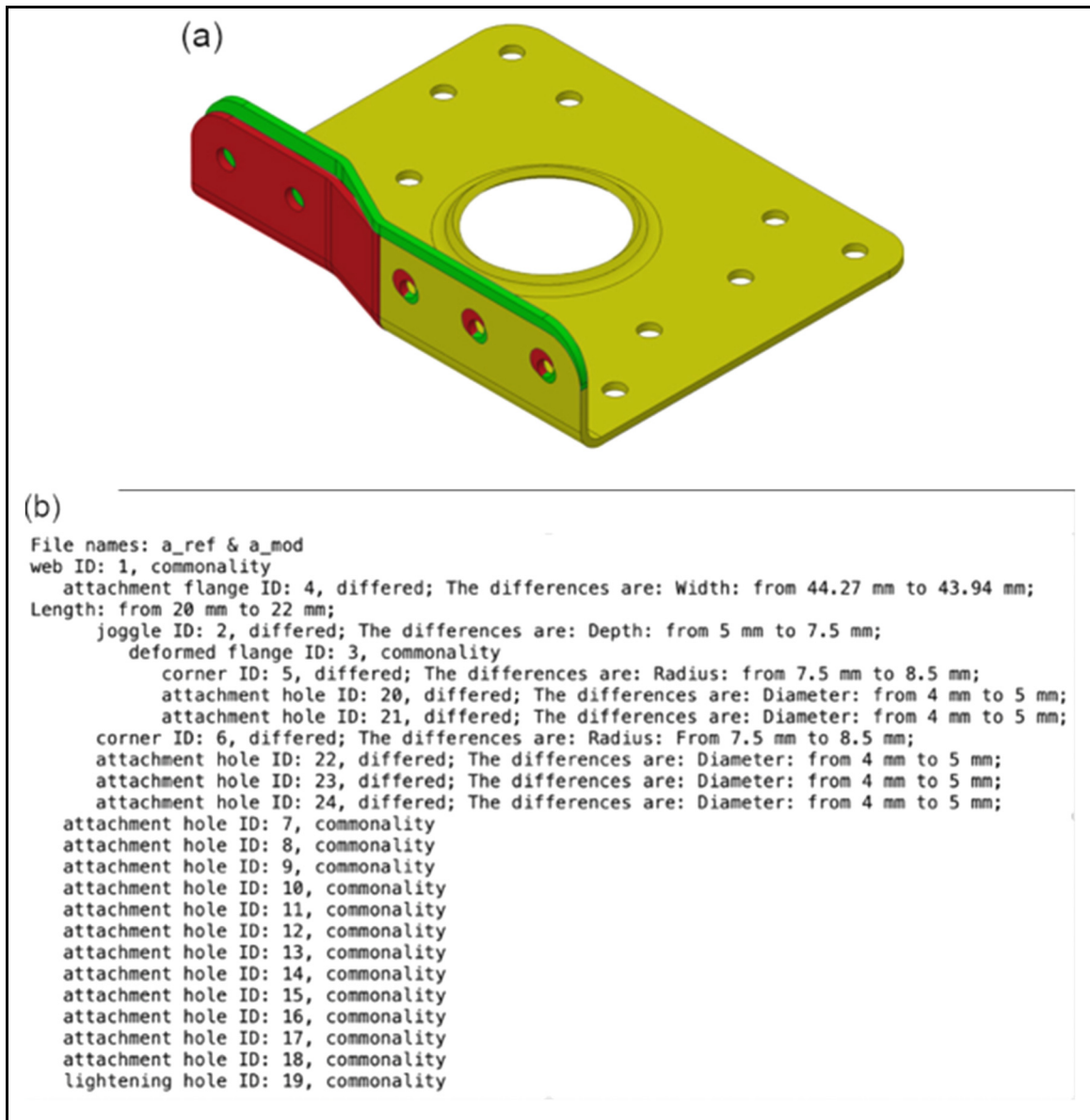


Figure 6.4 The color-coded representation of (a) the volumetric comparison of the part model presented in Figure 6.3 (a) and its modified version, and (b) the MDI prototype's output

Figure 6.4 shows the MDI prototype's output for the part represented in Figure 6.3 (c) and its modified version. Figure 6.5 (a) shows the color-coded results of the volumetric comparison to illustrate the differences and guide the reader. The commonalities are in yellow, the areas specific to the original part are in red, and the area specific to the modified version are in green.

We can see from that image that: the web, one of its tooling holes and two of its attachment holes are the same for both parts; and the rest of the features are different in various ways.

Figure 6.5 (b) shows the MDI prototype's output, which describes the differences between the parts at the feature level. The change in the bend relief radius alters the width of the flange whose ID is 3 (shown on the left-hand side of Figure 6.5 (a)). The flange whose ID is 2 (at the bottom of Figure 6.5 (a)) keeps the same width, but its child attachment holes are differed because their position is different. Also, a change in the angle of one of the attachment flanges (ID 4) alters the flange's length and position. The web, the attachment holes on it and one of its tooling holes as well as the corners of the flanges are all identified as commonalities. The altered tooling hole is correctly identified as differed since it was displaced due to the flange next to it becoming closed.

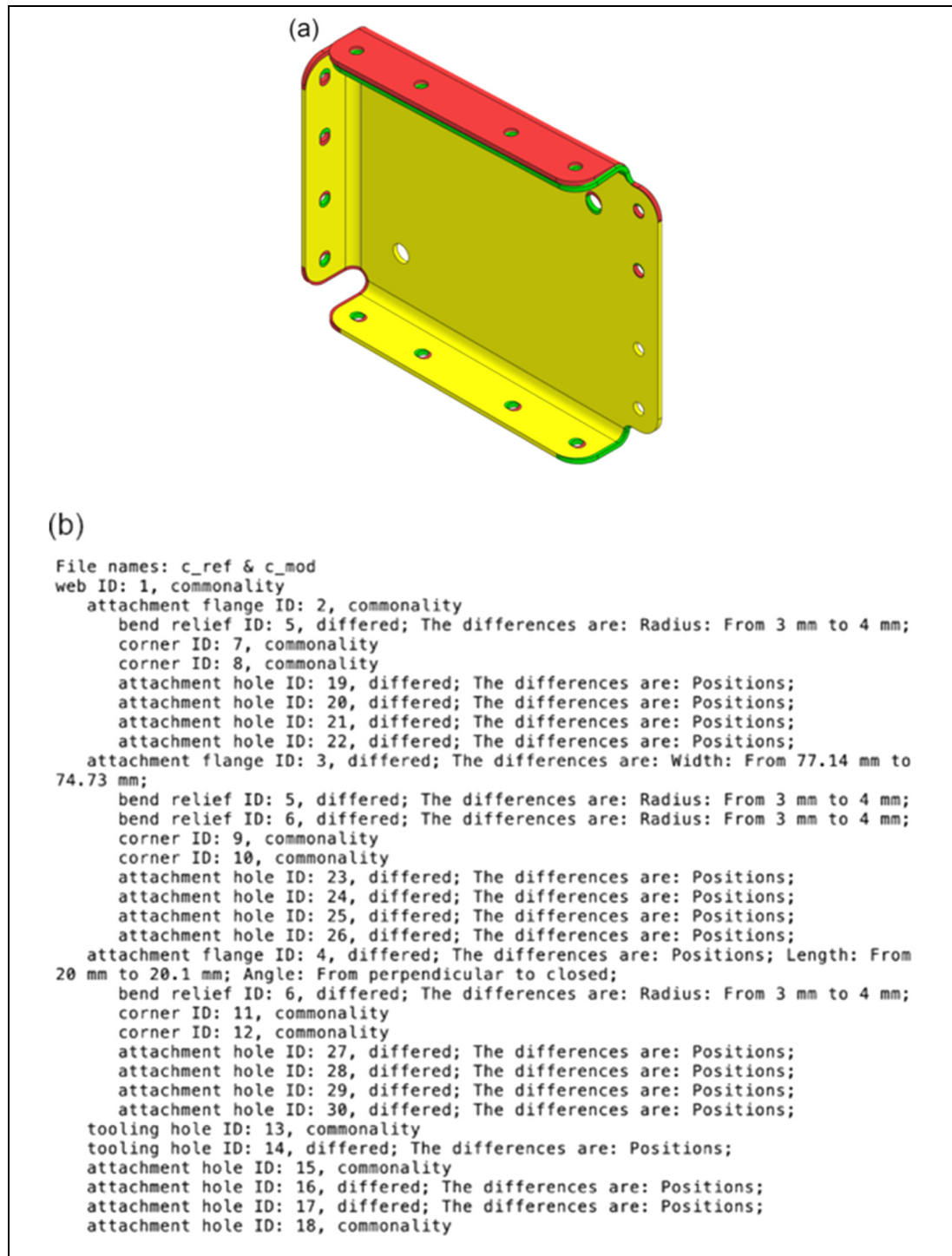


Figure 6.5 The color-coded representation of (a) the volumetric comparison of the part model presented in Figure 6.3 (c) and its modified version, and (b) the MDI prototype's output

Figure 6.6 shows the MDI prototype's output for the part represented in Figure 6.3 (e) and its modified version. Figure 6.6 (a) shows the color-coded results of the volumetric comparison to illustrate the differences and guide the reader. The commonalities are in yellow, the areas specific to the original part are in red, and the area specific to the modified version are in green. We can interpret from that image that the flange on the web and its return flange and attachment holes are differed, while the rest of the features seem to be the same.

Figure 6.6 (b) shows the MDI prototype's output. Apart from the immediate attachment flange on the web and its attachment holes and return stiffening flange, the rest of the compared parts' features are identified as commonalities. As expected, the MDI prototype reveals the diameter of the attachment holes on the attachment flange (ID 4) are different, the attachment flange's length has increased, and the return stiffening flange has been displaced.

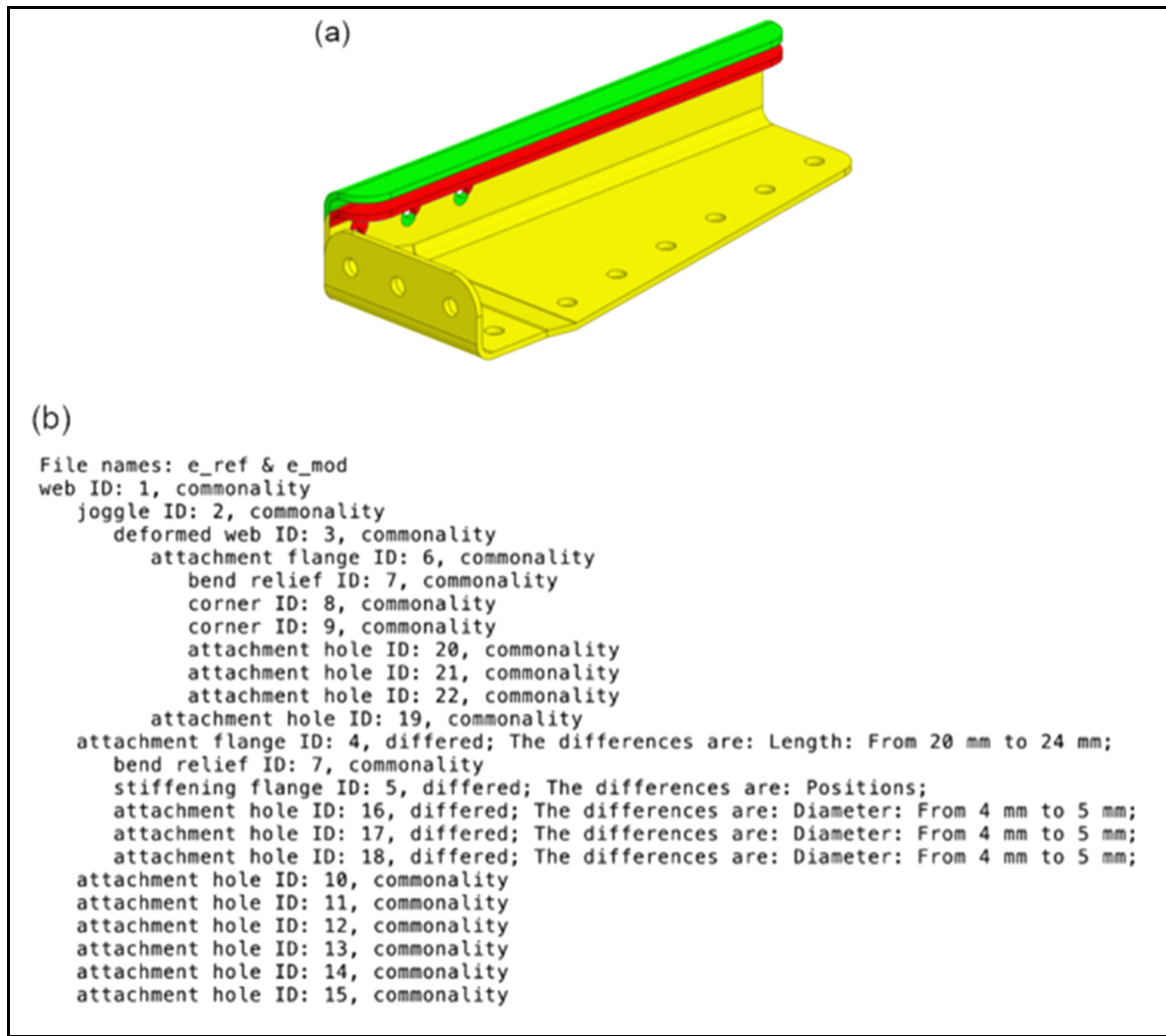


Figure 6.6 The color-coded representation of (a) the volumetric comparison of the part model presented in Figure 6.3 (e) and its modified version, and (b) the MDI prototype's output

6.3 Summary

In this chapter, we presented the results of a successful implementation of our MDI method for ASM parts. The details about pose registration and MDI implementation are presented in Appendices XIV and XV, respectively. The MDI prototype was developed using the same data structure as was used for the AFR prototype. The MDI prototype reads the feature structure and definitions of the ASM parts to be compared from their detailed *feature files*. A modified

AFR prototype was developed to store all the feature information needed for MDI in the detailed feature files. After the parts have been compared, the differences are represented in a text file called a *difference file*.

The results show perfect accuracy when it comes to identifying all the features of the compared parts and confirm there is great potential for further development of MDI algorithms based on feature-based models of parts from specialized domains of application. Although MDI is thoroughly investigated in the literature, feature-based MDI has not been investigated as extensively and has not yet become an integral part of CAD solutions. A rather specialized approach like the method we propose in Chapter 4, whose implementation results are represented in this chapter, makes it possible to break the MDI problem down into steps that are based on the feature structure of the part models to be compared and propose accurate specialized feature-based solutions.

CHAPTER 7

DISCUSSION OF THE RESULTS

In the course of this thesis, an AFR method was proposed to elevate the level of semantics of ASM part models communicated in B-rep STEP files and a semantic MDI method was subsequently proposed to compare CAD models of ASM parts and represent their differences semantically. Afterwards, prototypes of the proposed AFR and MDI methods were developed to verify them and validate their results. In proposing the MDI method, the feature taxonomy originally proposed for the AFR method was modified to better fit the needs of the MDI method. In this chapter, we take a look at the evolution of the original AFR method, highlight the scientific contributions of the AFR and MDI methods and position them relative to the literature, review the benefits of the proposed MDI method, and consider the assumptions and limitations of this work. Finally, possible future research directions are suggested.

7.1 Aerospace sheet metal feature taxonomy

The original feature taxonomy developed for the proposed AFR method was modified to better fit the needs of an MDI method. The original taxonomy was developed based on feature formation operations – trimming and forming – and served as an avenue to organize the features considered at the beginning of this study. However, the feature taxonomy was found to be able to contribute to the semantic MDI method by organizing the features considered based on their associated functions / design intents. Given that the features and their functions / design intents constitute very important semantic information, it was decided reformulating the taxonomy based on them should be the first step in proposing a semantic MDI method. The modified taxonomy incorporated a base feature and refinement / contact features. The base feature and contact features were further broken down as they are the essential features required to form any ASM part. This premise was used to establish semantic pose registration and comparability (the existence of enough similarity to be able to identify differences) based on the base feature and contact features. A similar MDI approach could be adopted to compare

various types of parts if they are designed based on features having distinct functions or design intents. For example, the features of rotary cylindrical parts (shafts) could also be categorized based on their functions: threads for fastening, grooves and steps for retaining, reference surfaces for bearings, slots for locking, etc. Assuming such a taxonomy is used, the shafts' pose registration and feature-based comparison could be rationalized for semanticity. Pose registration could be based only on reference surfaces, with comparability established when there are at least two matching reference surfaces, and so on and so forth.

Compared to machined mechanical parts, which have historically been the focus of AFR and MDI research, structural ASM parts have distinguishing characteristics. A limited number of design features is one of their most important distinguishing characteristics. The abundance of features that are possible in machined mechanical parts usually requires that these parts be further categorized as rotary parts, prismatic parts, etc., which is unnecessary for structural ASM parts. Also, structural ASM parts have few intersecting features, i.e., twin joggles, stringer cutouts, or a combination thereof. Not only are these features' intersections with other features predictable, but their occurrence also originates from specific design intents. On the contrary, feature intersection in machined mechanical parts can involve multiple features and occur for a multitude of reasons, including design creativity. Speaking of creative designs, features are applied creatively in machined mechanical parts to implement a multitude of functions, e.g., a hole could be used for fastening, bearing, accessing, tooling or manufacturing, among other purposes. Hence, ASM part design can be viewed as distinct in nature from machined part design.

7.2 Contributions

In the context of the literature review conducted in Chapter 2, this work is unique in certain ways. It is the first work to address structural ASM part feature taxonomy and AFR. With respect to the literature on MDI and CAD model comparison in general, this work has some points in common with the existing literature. In terms of the comparison interface, which is

the type of data that the user interacts with in the comparison solution, the MDI method proposed herein could be said to belong to B-rep 3D CAD models if the AFR method is considered part of the final solution or unspecified CAD models if the feature files are considered the interface. The approach considered to organize the literature on CAD model comparison itself using the subject, interface, medium, problem, solving method and purpose is entirely new. Also, the feature files that are introduced in this thesis to contain the feature structure and definitions of the parts are not in a standard file format, but rather text files that are read and converted to structured data. This work's comparison medium is semantic information, as is that of the design features. The comparison problem formulation used in this work is feature comparison. The comparison solution proposed here is unique as it relies on the feature structure and definitions for semantic pose registration and the pose registration to structure the comparison approach from the base feature to the child features.

The semantic pose registration method proposed has one main advantage over the geometric pose registration methods that are mainly rooted in registering the position of the compared parts based on their features that have essential functions. Unlike the geometric pose registration methods, the semantic pose registration method is indifferent to the features that have low functional significance and are rather geometrically significant. For example, a stiffening flange in an ASM part is geometrically more significant – simply bigger – than an attachment hole on an attachment flange, but far less functionally significant when attaching ASM parts is a key function to create ASM structures. This also means that when semantic pose registering any specific type of part, identifying the features involved in the main functions is the first step to simplify the problem and solve it. Like the semantic pose registration method proposed, the semantic MDI method proposed in this work has a main advantage over the geometric (shape) MDI methods used in the literature that are rooted in comparing parts based on their features. Unlike the geometric MDI methods, feature-based semantic MDI does not rely on color coding to communicate the differences between the compared parts. The differences could be presented in terms of features being added, removed, or modified in the parameters in addition to being visualized by color coding the features'

geometry. This makes it possible to accurately quantify numeric differences without losing difference visualization.

7.3 Assumptions and limitations

The work presented in this thesis is based on some underlying assumptions whose impact should be discussed. The first assumption that was made was that the webs of the structural ASM part models are the parts' largest planar feature. Although the vast majority of ASM parts do not challenge this assumption, there are cases in which the design context necessitates webs that do not fit this assumption, e.g., deep U-shaped or Z-shaped parts. Also, it was assumed that the bend radius is constant and the same for all the deformed features. Although this assumption was not challenged by any of the sample parts studied or design guides, it certainly could not be made for *all* sheet metal parts. The assumptions underlying the comparison of two models are that the compared models represent parts with similar-enough functions and that the functions are served by the same features. It was assumed that all the features that compose a part are purposeful. This provided the opportunity to semantically register the feature-based 3D CAD models of ASM parts according to the unique purposes of their features.

Any of the assumptions mentioned above, when contradicted, could become a limitation of the methods proposed and the approach taken in this work. Apart from this, the very specific domain of application, structural ASM parts, inherently limits the applicability of this work somewhat. Structural ASM parts are designed with very specific design guidelines and feature functions, a limited number of features and a limited freeform portion, which is not the case for other types of parts, even other types of sheet metal parts used in automotive or computing infrastructure. The fact that most structural ASM features have a single function, as opposed to the multitude of functions associated with the features of other types of parts, makes robust semantic MDI possible. It would not be possible otherwise. For example, tooling holes and attachment holes have specific diameters and functions, and when attachment holes are used for tooling purposes in addition to their main function (attachment), the absence of tooling

holes on their parent web reveals that. Furthermore, an attachment hole's tooling purpose and attachment function apply to different phases of its lifecycle – fabrication and operation, respectively. On the contrary, a hole on a machined part of an engine might be of any diameter regardless of whether its function is attachment, axle mounting, bearing, lubricant passage, etc.

7.4 Future works

Looking at the limitations and assumptions of this research, one avenue for future work would be to mitigate them. In addition, the MDI method proposed in this thesis was tested with a limited number of samples, and more test cases could be recommended for future work. Another avenue would be to pursue future work that could be envisioned with similar assumptions, albeit with similar limitations. Repeating similar highly successful studies, although they are rather specific in application, will add up to a significant practical contribution. For example, if similar success is found for structural aerospace composite and machine part AFR and subsequently MDI, the results of this research will collectively contribute to the efficient design and development of aircraft structures.

In addition, although structural ASM part AFR is used here only to enhance the semanticity of those parts' MDI, it has many other potential applications like automatic part family creation, part search engine enhancement, and design reuse. The features recognized from parts' 3D CAD models can be used to form part families that have certain functionalities and express a specific design intent. These features could also be used to create search predicates to look for parts with certain functionalities in a database. The automatic part family creation and search engine enhancements can collectively increase part design reuse significantly by facilitating access to existing relevant design information.

As mentioned above, this research proves that differences can be presented in terms of features being added, removed, or modified in the parameters in addition to being visualized by color coding the features' geometry, and hence makes it possible to accurately quantify differences

without losing difference visualization. For example, the difference between two holes of different diameters could be quantified and presented as an x -mm difference in diameter as well as colouring the faces in question yellow. This possibility was not showcased or capitalized on in developing the prototype. Also, the fact that differences can be quantified suggests that similarities could also be quantified, which opens up many opportunities for part search engine improvement to refine search results. For example, the search results for ASM parts having attachment holes within a certain range could be refined to eliminate ones having different applications than the design intended to be found.

CONCLUSION

In the course of this thesis, the general objective of proposing a method for identifying the differences between two CAD models and representing them semantically for structural aerospace sheet metal parts was successfully achieved. The semantical identification and representation of the differences was through design features, hence, an AFR method was also successfully proposed to elevate the level of semanticity of the B-rep models which usually ASM parts are communicated in. Both specific objectives: (1) propose an AFR method for ASM parts and (2) propose a semantic CAD MDI method based on the feature models of the structural ASM parts were achieved.

The AFR method proposed in this research for ASM part models encompasses two major rule-based steps. The first major step details classifying and grouping the elements of 3D B-rep model. The second major step describes the ASM features recognition in detail. To propose the AFR method for structural ASM part models, which has not been addressed by any previous work, the real-world structural ASM parts, models and designs were used. The AFR method addresses recognizing features of structural ASM parts which have not been covered by previous work i.e., web, joggle, twin joggle, stringer cutout, corner and bend relief.

Based on the design guides and the 168 studied samples, it was assumed that a web is the planar portion of an ASM part with the highest surface area. This assumption was satisfied by a great majority, 98%, of our sample structural ASM parts. It was also assumed that all the bends have constant bend radii. Also, features like hem and curl, lance and louver that have been included in some previous scientific literatures are not observed in the design of our sample structural ASM parts and design guidelines and are therefore excluded from the scope of this work.

The recognized features were described through their geometry, feature relationships and parameters, which enables describing any ASM part only by its features. The AFR method

defines the features' relationships based on their engineering semantics rather than recognition order, or modeling order if they were extracted from native models. The combination of features' geometry, semantic relationships and parameters provides all the necessary semantic information – as comparison medium – for a semantic CAD model comparison.

The AFR prototype was implemented by first developing a data structure for the B-rep elements and a data structure for feature definitions. Next, algorithms for classifying B-rep elements and recognizing features were developed. The results showed perfect accuracy in recognizing all the features of almost all sample parts and confirmed great potential for further development of AFR algorithms in specialized domains of application. Although feature recognition from B-rep models has been investigated for decades, it has not become an integral part of CAD solutions. Here it is suspected that works on AFR have been usually too general in the literature and focus on a specific category of parts like the structural ASM parts is key. For example, focusing on structural ASM helped this research by limiting the number of design features between which there is usually predictable intersections.

The semantic MDI method proposed in this research approaches 3D CAD MDI as comparing feature-based part models. The proposed method includes two major stages: semantic pose registration and difference identification, both based on semantics of the models' design features. The semantic pose registration relied on the compared models feature structure and features definition. It is a unique approach that takes advantage of parts' function conveyed through its design features to semantically register the models. Factoring in parts' function in pose registration instead of parts' shape is a valuable differentiator as the ability to pose register is considered the indicator of comparability of the parts.

The proposed difference identification method starts with eliminating commonalities between the models. Excluding the commonalities of two objects from their comparison results effectively identifies their differences. Next, the differences are categorised as modification, addition and removal differences and are characterized. Both the semantic registration and the

difference identification methods are essentially novel for their integration of the design features and their functional implication in the comparison process. Also, the notion of segregating commonalities in order to group differences of CAD models had not yet been explored, and this work shows its useful application.

The MDI prototype was developed using the same data structure of the AFR prototype. It reads the feature structures and definitions of the ASM part models from their detailed feature files. After comparing the parts, the differences are represented in a text file called difference file. The results showed perfect accuracy in identifying all the differences between the compared parts and confirm there is great potential for further development of MDI algorithms based on feature-based models of parts from specialized domains of application. Just like the AFR method, the MDI method proposed here takes advantage of a specialized approach that appears to conclusively enable accurate feature-based difference identification.

The proposed semantic MDI method could be considered in some future work. For example, the application of our semantic pose registration method for finding similar models could be of interest. Given that two models are comparable if they are pose-registerable and assuming that a pose registration based on models features and their implied functions guarantees matching fundamental functionalities between the part models, the pose registerable part models could be acceptable as results of a 3D model search engine. A semantic MDI has many advantages over the non-semantic ones. The semanticity in the difference representation mitigates the risk of misinterpretation of the differences and engineers' errors, saves time, reduces non-added value activities, etc. This may reduce development iterations, development cost, time-to-market and most importantly product competitiveness.

APPENDIX I

CLASSIFYING SHEET_FACES ADJACENT TO WEB_FACE

The algorithm for classifying sheet_faces adjacent to web_face starts with checking if each of the faces adjacent to the web_face is a sheet_face. If the sheet_face is adjacent to the web_face through its outer_bound, it is classified as a bend_face, otherwise it is classified as an internal_face and all its adjacent faces that are sheet_faces are also classified as internal_faces.

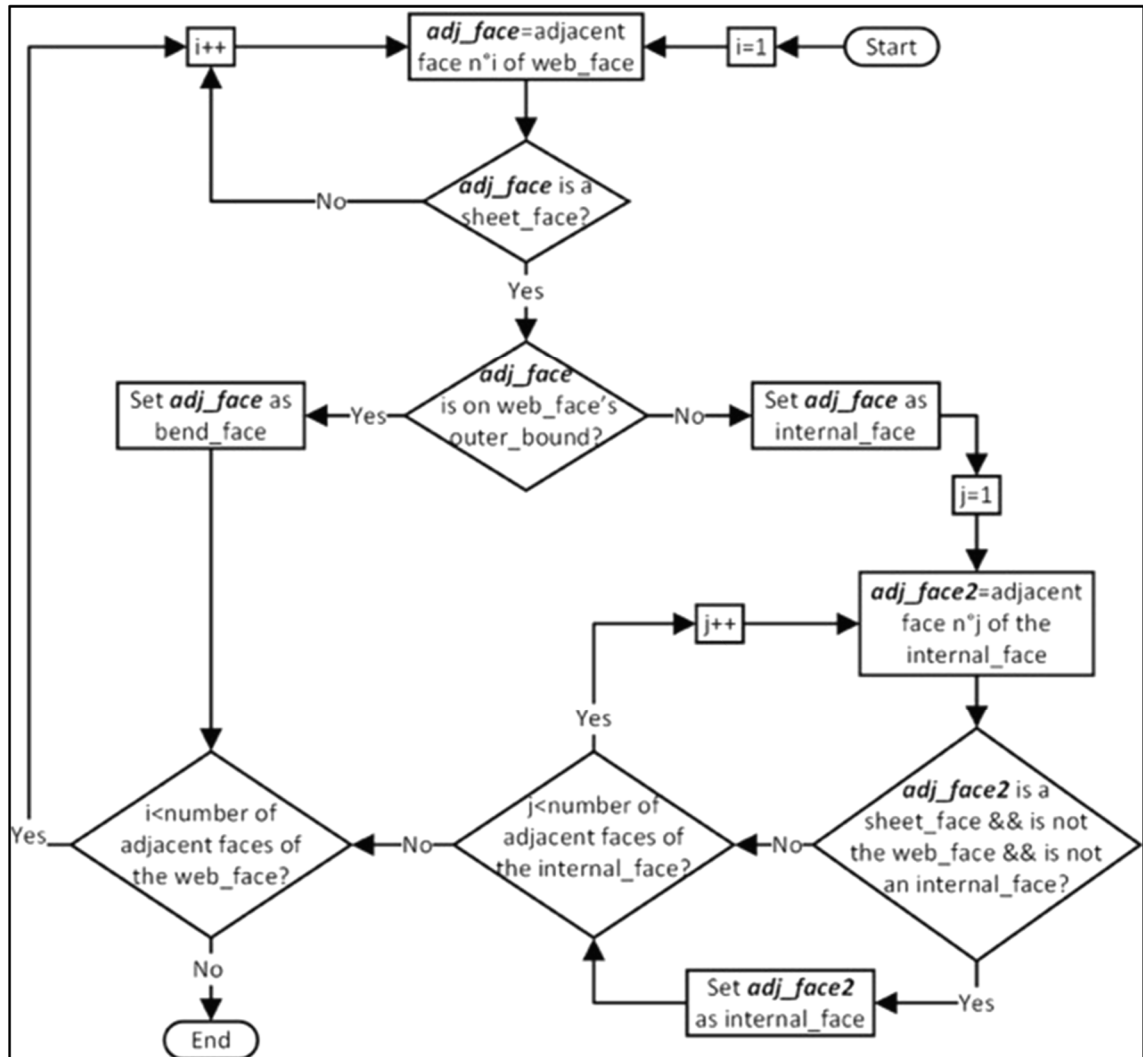


Figure-A I-1 Algorithm for classifying sheet_faces adjacent to web_face

APPENDIX II

CLASSIFYING REMAINING SHEET_FACES

The algorithm for classifying the remaining sheet_faces as wall_faces, bend_faces, internal_faces or detained_faces starts with the outside_faces list. Some of these sheet_faces are already classified by their subtypes and will be used to classify their adjacent sheet_faces. If a face is adjacent to only one bend_face, it is classified as a wall_face, and if it is adjacent to more than one bend_face, it is classified as a detained_face. Once all the faces in the outside_faces list have been checked to classify those that are adjacent to a bend_face, the second round of the algorithm starts. In the second round, each face in the outside_faces list is checked to determine whether it is adjacent to a wall_face. If a face is not adjacent to a wall_face but is adjacent to a detained_face, it is classified as a bend_face. This condition and its associated step are missing from the originally proposed method and highlighted in the flowchart below. If a face is adjacent to a wall_face through its outer_bound, it is classified as a bend_face, otherwise it is classified as an internal_face.

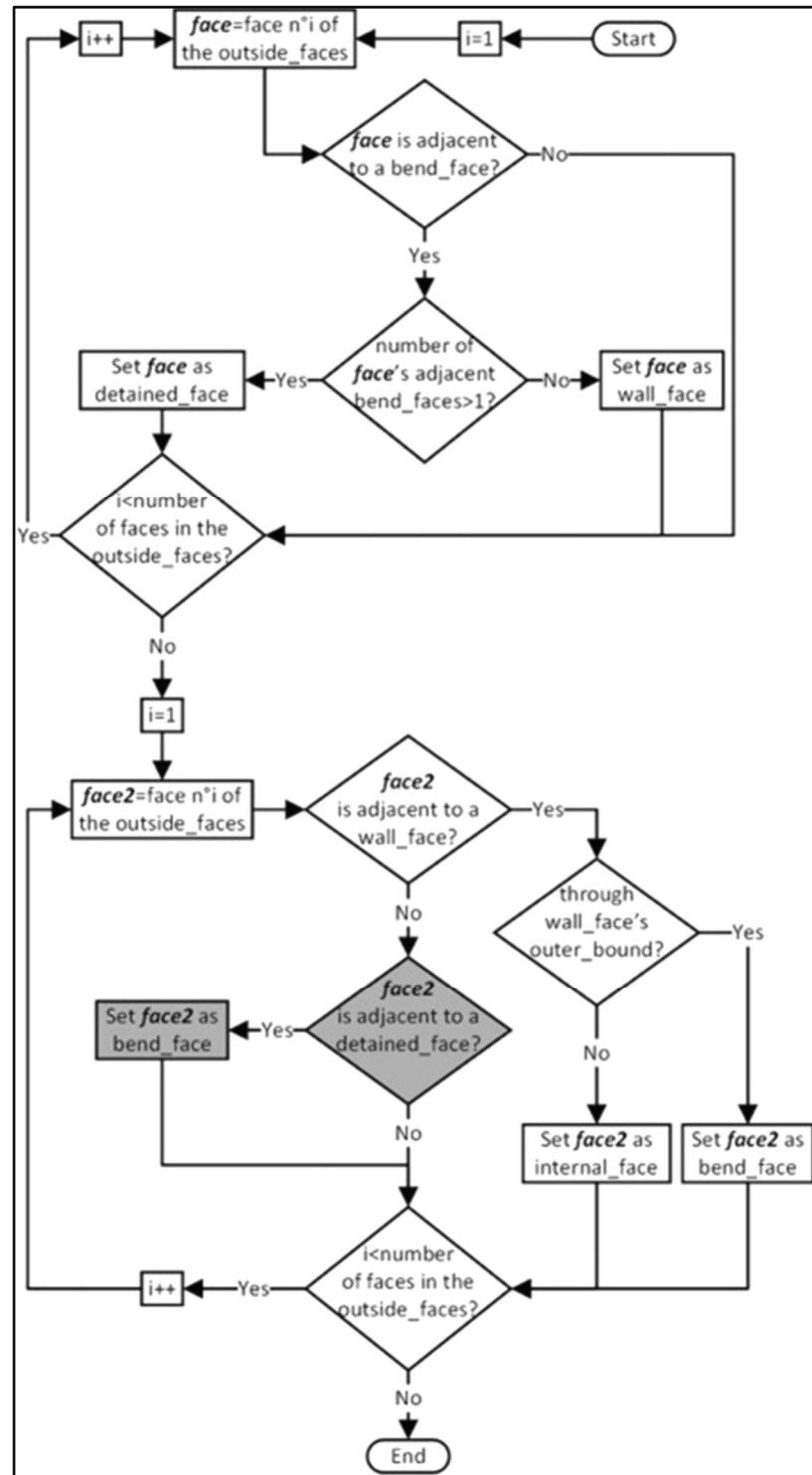


Figure-A II-1 Algorithm for classifying the remaining sheet_faces

APPENDIX III

CHANGING BEND_FACES TO CONNECT_FACES

The algorithm for changing eligible bend_faces to connect_faces starts with the outside_faces list. If a face is a bend_face and is also adjacent to more than one other bend_faces, it is reclassified as a connect_face. The outside_faces list is rechecked until no new connect_face is identified.

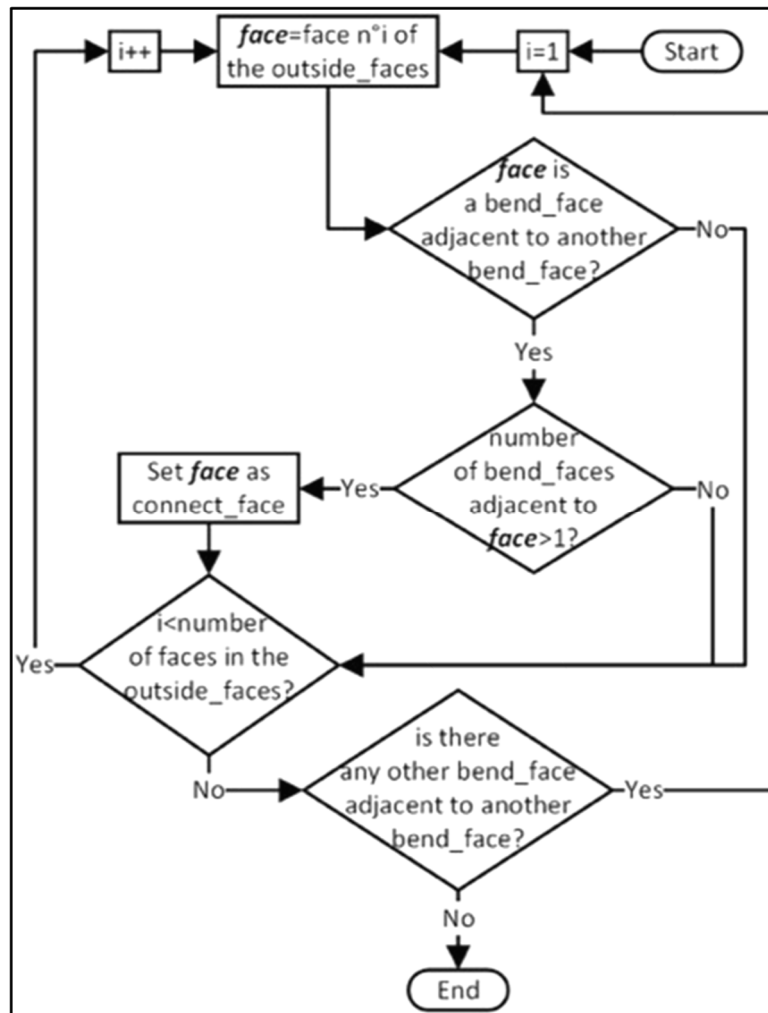


Figure-A III-1 Algorithm for changing eligible bend_faces to connect_faces

APPENDIX IV

CHANGING DETAINED_FACES TO CONNECT_FACES

The algorithm for changing eligible detained_faces to connect_faces starts with the outside_faces list. If a face is a detained_face and is surrounded by bend_faces and wall_faces so it does not have any adjacent trim_face on its outer_bound, it is re-classified as a connect_face. The condition of being surrounded by bend_faces and wall_faces is missing in the original method and highlighted in the flowchart below.

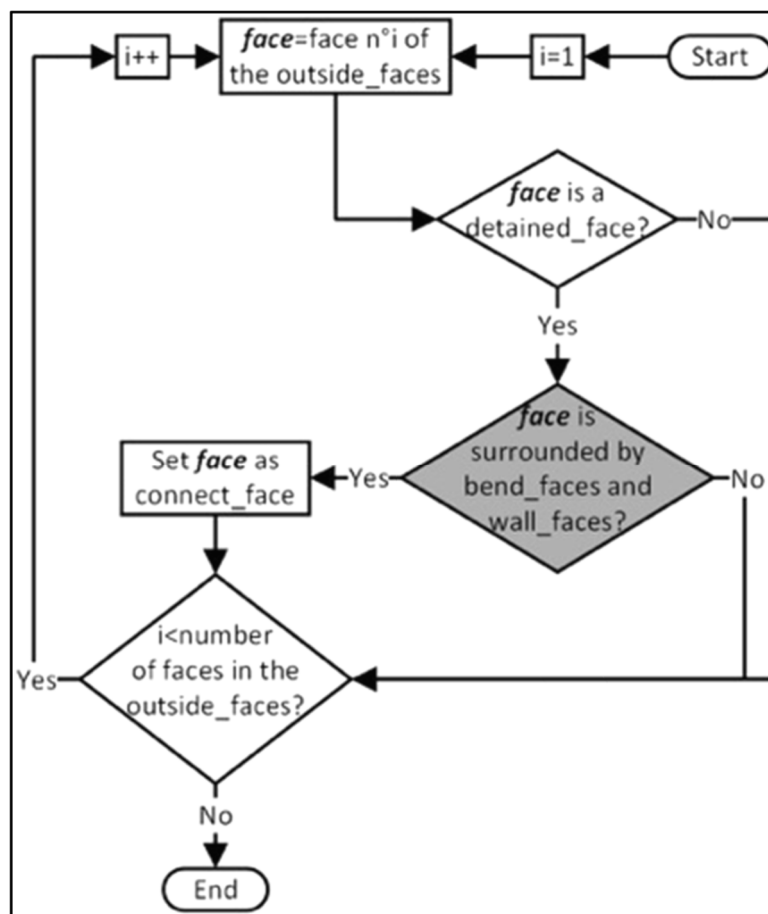


Figure-A IV-1 Algorithm for changing eligible detained_faces to connect_faces

APPENDIX V

CHANGING DETAINED_FACES TO BEND_FACES OR WALL_FACES

The algorithm for changing eligible detained_faces to bend_faces or wall_faces starts with the outside_faces list. If a face is a detained_face and is adjacent to a connect_face, it is re-classified as a bend_face. If a face is a detained_face and is not adjacent to a connect_face but is adjacent to a bend_face, it is re-classified as a wall_face.

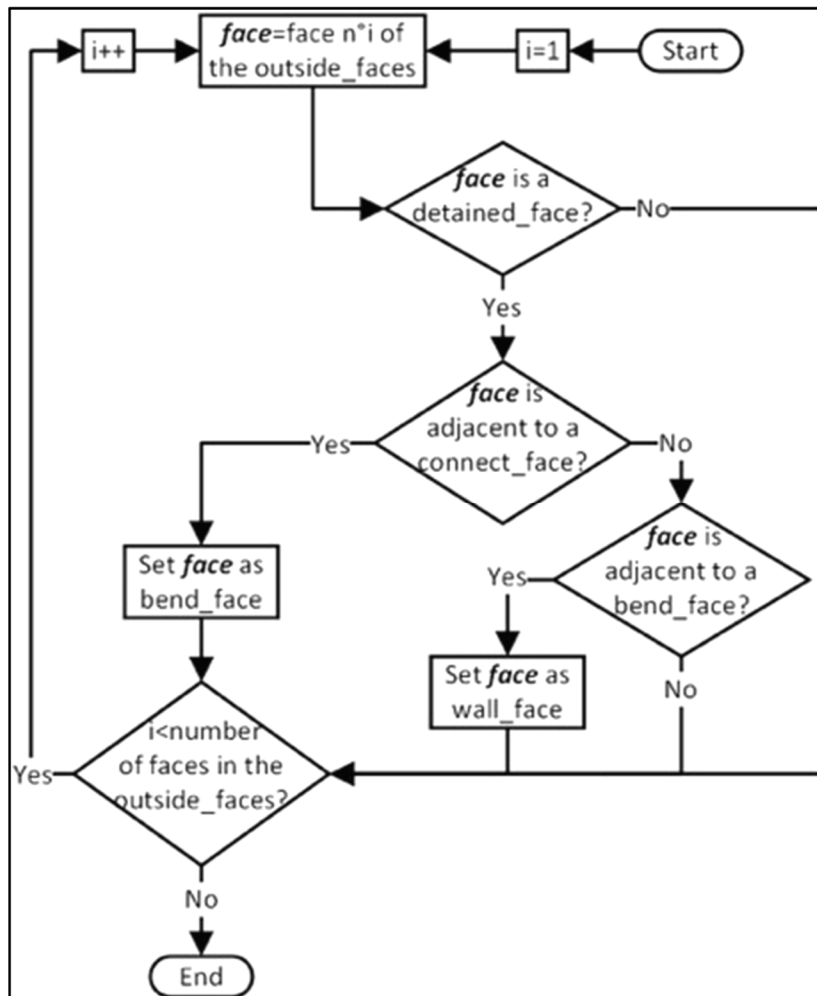


Figure-A V-1 Algorithm for changing eligible detained_faces to bend_faces or wall_faces

APPENDIX VI

CHANGING WALL_FACES TO BEND_FACES

The algorithm for changing eligible wall_faces to bend_faces starts with the outside_faces list. If a face is a wall_face adjacent to more than one other wall_faces, it is classified as a bend_face.

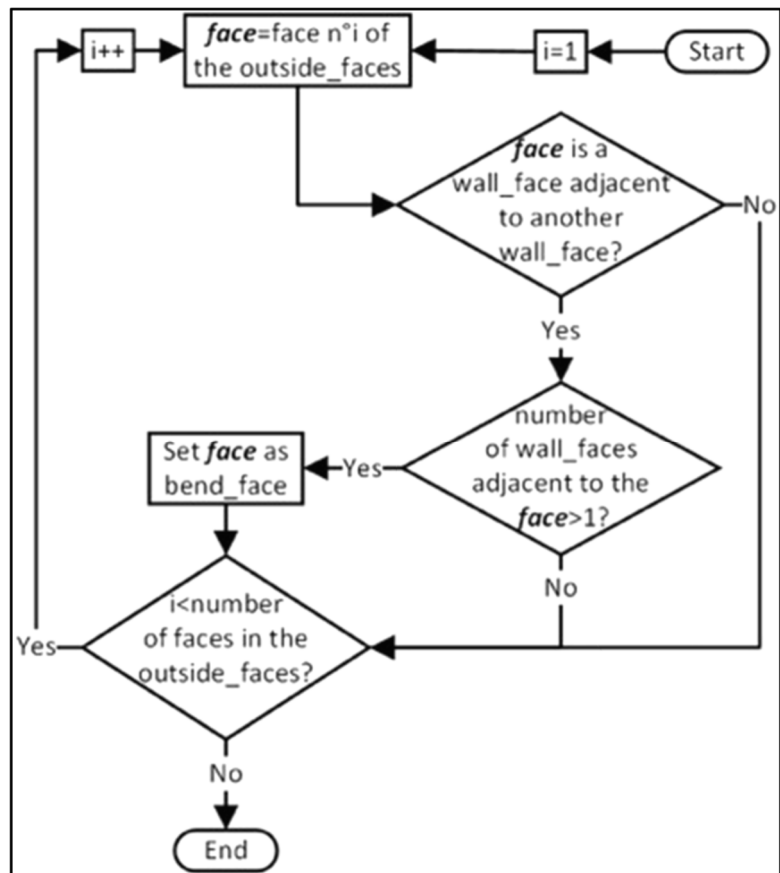


Figure-A VI-1 Algorithm for changing eligible wall_faces to bend_faces

APPENDIX VII

RECOGNIZING TWIN JOGGLES

The algorithm for recognizing twin joggles starts with the the_features list. If a feature is a deformed flange, the other deformed flanges in the list are checked to determine whether they represent the same deformed flange. They are checked by verifying whether they are linked to the same wall_face as the original deformed flange. If they are, a twin joggle is created based on the joggles that were the parents of these deformed flanges, and the twin joggle is made the parent of one of the deformed flanges (in the flowchart below, it is referred to as feature2). Also, a flange is created based on the flanges that were the original parents of the joggles that were combined to create the twin joggle. This new flange is made the parent of the twin joggle. At the end, the deformed flange that was NOT made the child of the twin joggle, the original joggles (NOT the newly created twin joggle) and the original flanges (NOT the new flange created based on the original flanges) that were the parents of the original joggles are all deleted from the the_features list.

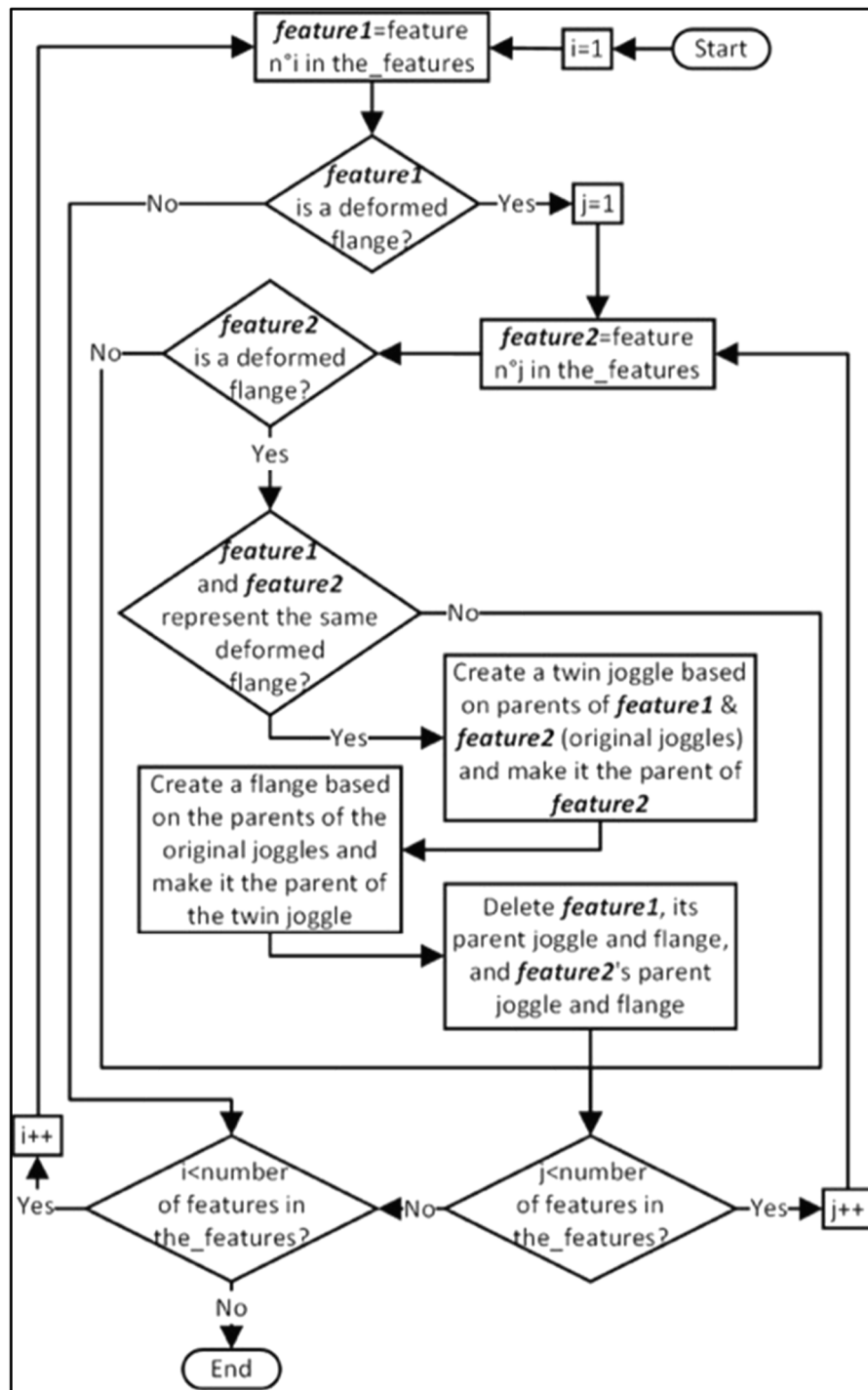


Figure-A VII-1 Algorithm for recognizing twin joggles

APPENDIX VIII

RECOGNIZING THE REMAINING FLANGES

The algorithm for recognizing the remaining flanges starts with the `outside_faces` list. If a face is a `web_face` or `wall_face` that has an adjacent `bend_face` that is not included in a previously recognized flange, deformed flange or `joggle_face_set`, a new flange is created. If the face is a `web_face`, make the web the parent of the newly created flange. If it is not a `web_face`, therefore a `wall_face`, and it is included in a previously recognized flange, the previously recognized flange is made the parent of the newly created flange. If the face is not a `web_face` and is included in a previously recognized deformed web, the previously recognized deformed web is made the parent of the newly created flange.

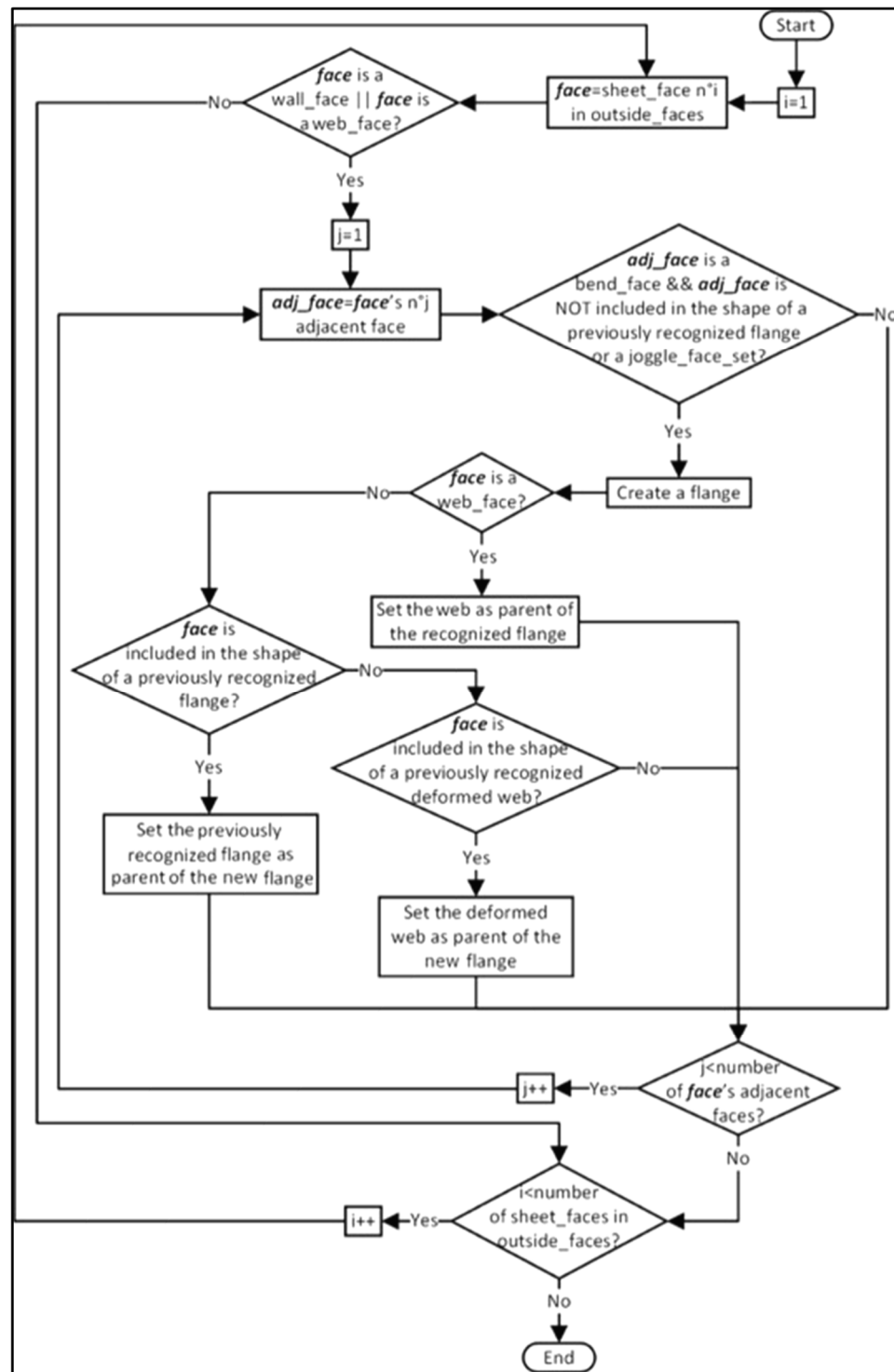


Figure-A VIII-1 Algorithm for recognizing the remaining flanges

APPENDIX IX

RECOGNIZING COMBINED FLANGES

The algorithm for recognizing combined flanges starts with the `outside_faces` list. Each of the `bend_faces` in the list is checked to determine whether any of its adjacent faces is a `wall_face` that is included in a previously recognized flange. Such adjacent faces are counted. Also, the face's adjacent faces that are not `wall_faces` are checked to determine whether they are `connect_faces` that are not included in a `joggle_face_set`. If such an adjacent `connect_face` is found and the number of adjacent `wall_faces` included in previously recognized flanges is 2, a new flange is created by combining the previously recognized flanges, and the face and its adjacent `connect_face` are added to the newly created flange.

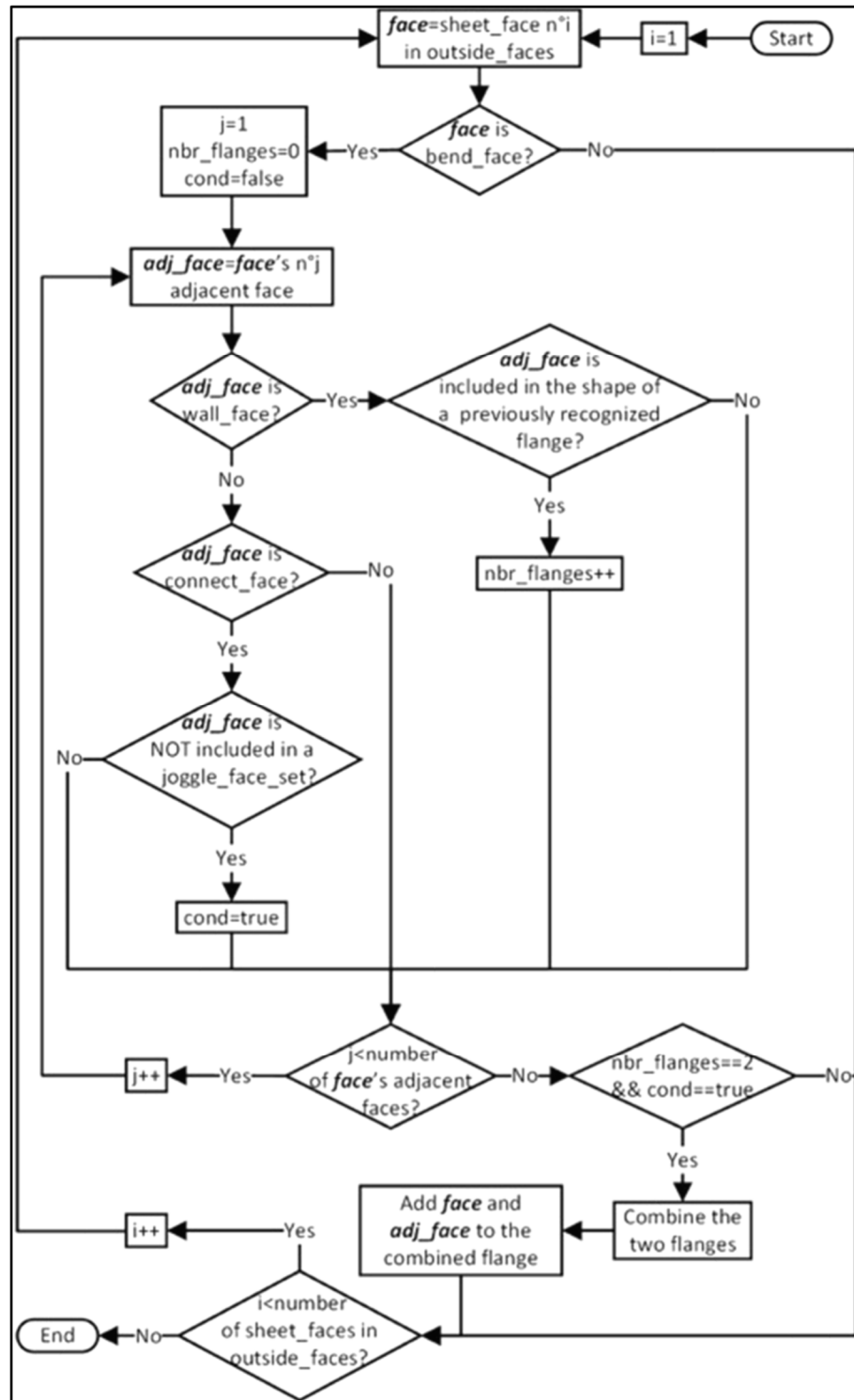


Figure-A IX-1 Algorithm for recognizing combined flanges

APPENDIX X

RECOGNIZING STRINGER CUTOUTS AND BEND RELIEFS

The algorithm for recognizing stringer cutouts and bend reliefs can be divided into two parts. The first part starts with the `outside_faces` list and a web or deformed web. Each `sheet_face` of these features that is a `web_face` or `wall_face` is checked to make a list of trim edges in its `outer_bound` that are between two edges that are each shared with a flange or deformed flange. If the flanges or deformed flanges have identical supporting geometries and the number of edges in the list is more than 2, a stringer cutout is created based on the list of edges. If the stringer cutout is between two flanges, they are merged into one, and if the stringer cutout is between two deformed flanges, they are merged into one, their parent joggles are merged into one twin joggle, and the joggles' parent flanges are merged into one flange. If the flanges or deformed flanges do not have identical supporting geometries, there is only one edge in the list, and the edge is a non-lin edge that is not G1 continuous with its adjacent edges, a bend relief is created based on the trim edge.

The second part starts with the `the_features` list and a flange. Each `wall_face` of the flange is checked to make a list of trim edges in its `outer_bound` that are between two edges that are each shared with a child flange, a deformed flange or a `bend_face` that is adjacent to its parent feature. If the child flange, the deformed flange or the parent feature do not have identical supporting geometries, there is only one edge in the list, and the edge is a non-lin edge that is not G1 continuous with its adjacent edges, a bend relief is created based on the trim edge.

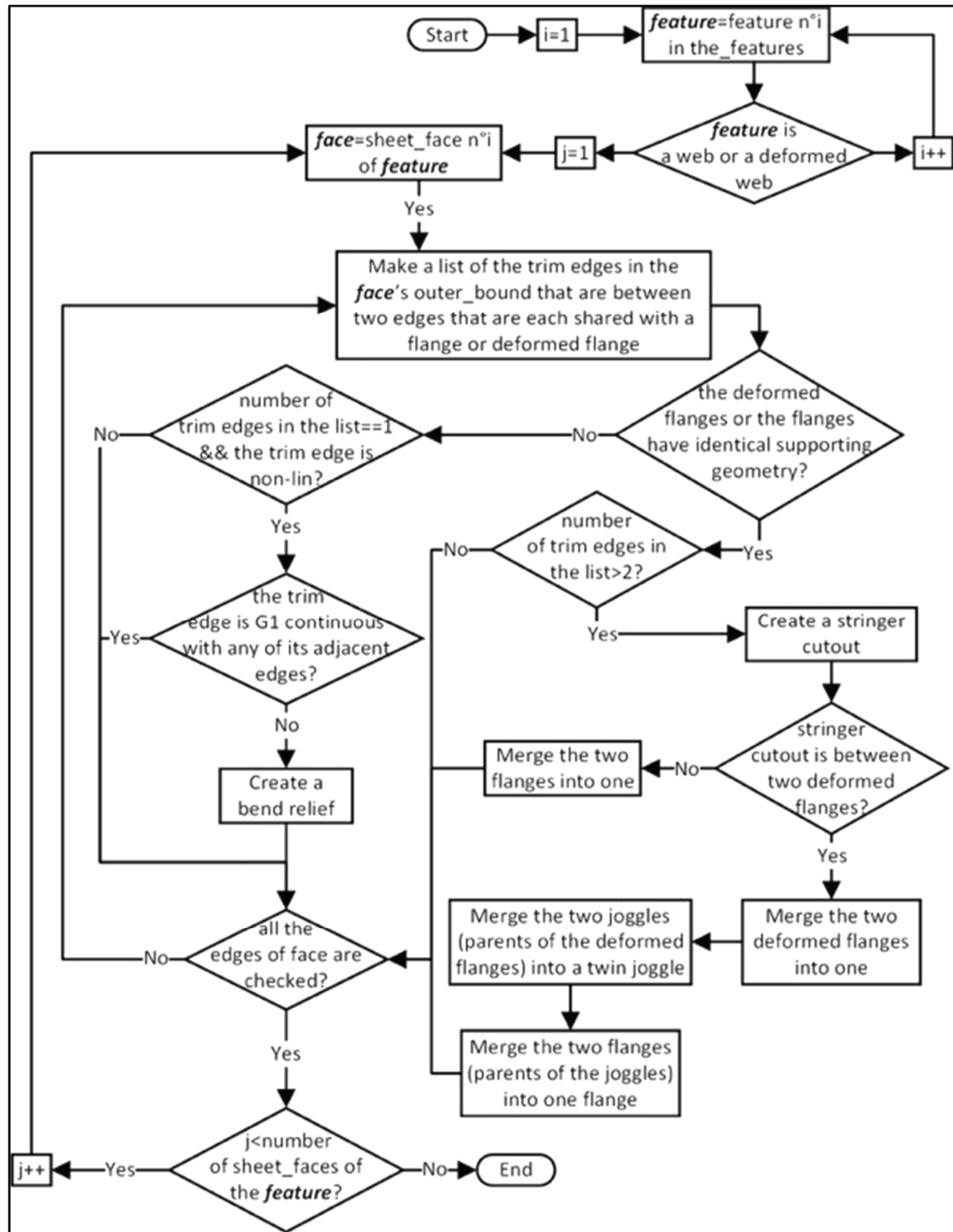


Figure-A X-1 Part 1 of the algorithm for recognizing stringer cutouts and bend reliefs

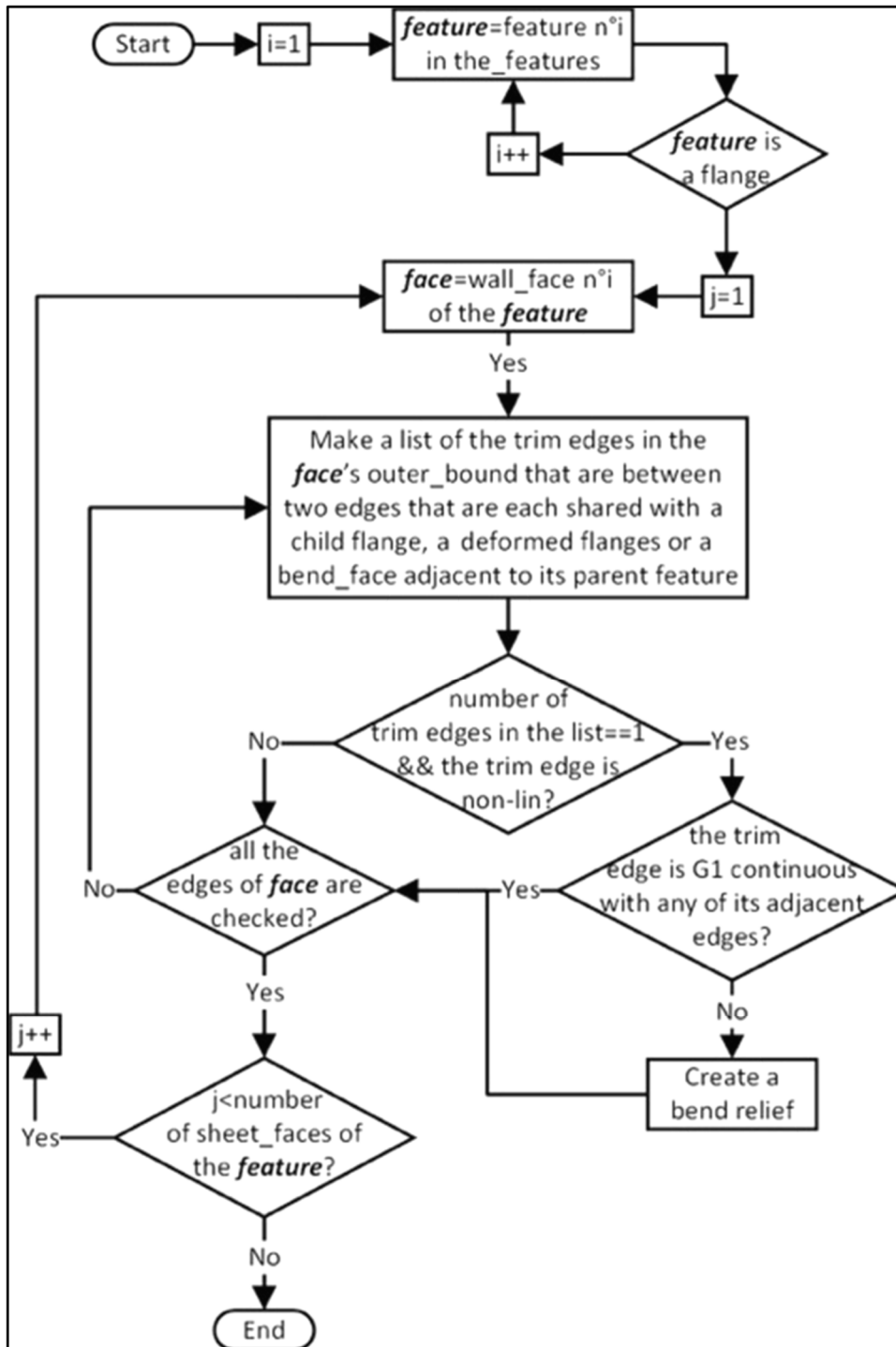


Figure-A X-2 Part 2 of the algorithm for recognizing stringer cutouts and bend reliefs

APPENDIX XI

RECOGNIZING CORNERS

The algorithm for recognizing corners starts with the `outside_faces` list. The edges in the `outer_bound` of each `wall_face` in the list are checked to determine whether they are trim non-lin edges and G1 continuous with their adjacent edges. If an edge is both a trim non-lin edge and G1 continuous, a corner is created based on the edge.

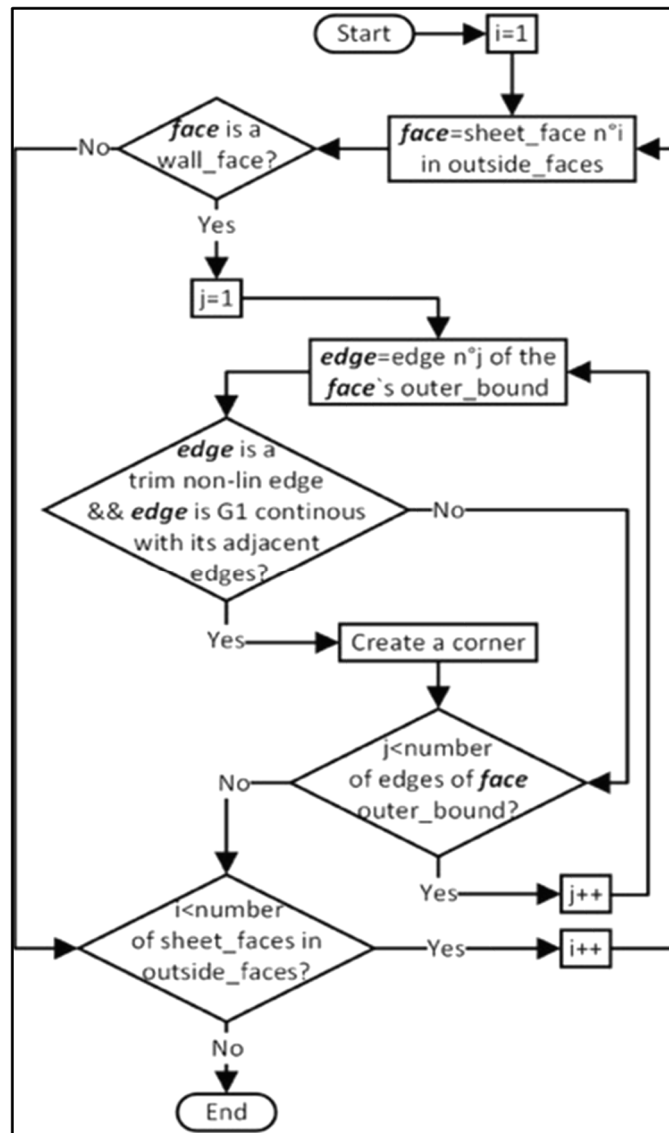


Figure-A XI-1 Algorithm for recognizing corners

APPENDIX XII

RECOGNIZING HOLES, CUTOUTS, LIGHTENING HOLES, LIGHTENING CUTOUTS AND BEADS

The algorithm for recognizing holes, cutouts, lightening holes, lightening cutouts and beads starts with the `outside_faces` list. Each bound of each `web_face` and `wall_face` in the list is checked to determine whether it is an `outer_bound`. If a bound is a `bead_bound`, a bead is created based on it. If a bound is a `hole_bound` and is made of trim edges, a hole is created based on it, or if it is instead made of non-trim edges, a lightening hole is created based on it. If a bound is an `internal_bound` and is made of trim edges, a cutout is created based on it, or if it is made of non-trim edges, a lightening cutout is created based on it.

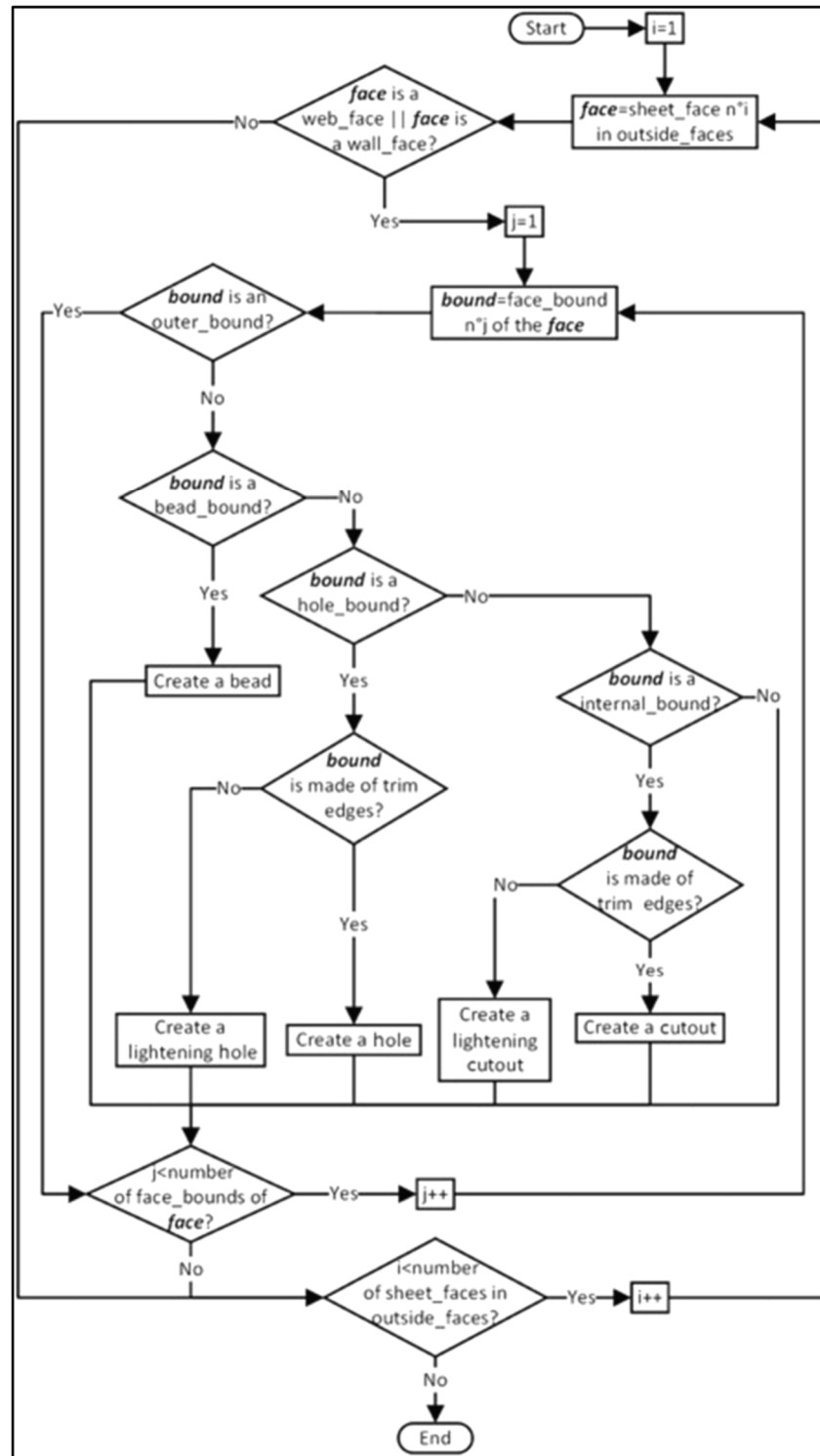


Figure-A XII-1 Algorithm for recognizing holes, cutouts, lightening holes, lightening cutouts and beads

APPENDIX XIII

RECOGNIZING LIPS

The algorithm for recognizing lips starts with the `the_features` list. Each feature that is a combined-open-immediate-stiffening flange is used to create a lip based on it and then deleted.

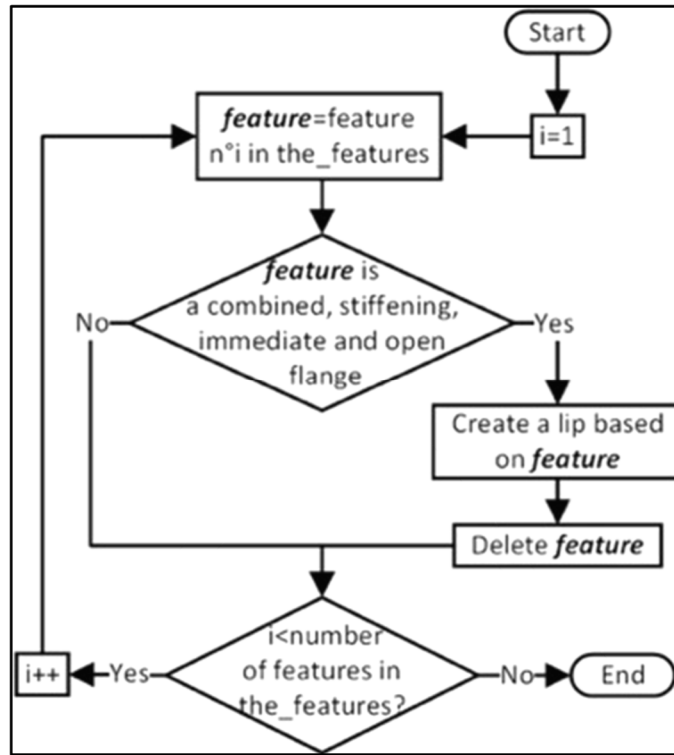


Figure-A XIII-1 Algorithm for recognizing lips

APPENDIX XIV

POSE REGISTRATION DETAILS

Pose registration starts with transforming both models to a normalized position. This normalization is achieved by making web faces of the two models and the XY plane coplanar. Notably, the following steps are executed based on the features recognized in the AFR process. This pose registration method exploits the fact that all the features of a part serve specific functions, some of which are fundamental to the part's essential functionality, and that they are intertwined with the design intent of the part. This provides an uninvestigated opportunity to semantically register feature-based 3D CAD models according to the unique purpose of their features in their specific domain of application.

If there are any attachment flanges on the webs, the scenarios for pairing the immediate attachment flanges on the webs are created and ranked based on the number of coinciding attachment flanges. The attachment flanges are essential to the interfacing of ASM parts and are, hence, considered as major functional features of sheet metal parts. All the scenarios that are not at the highest rank are removed; it is possible that there are multiple scenarios with the highest rank. If there is one scenario at the highest rank and the models are fully constrained, the transformation is performed based on that scenario.

If there are multiple scenarios at the highest rank or the models are not fully constrained by the highest ranked scenario, the presence of any attachment hole on the web is checked. If there are any attachment holes on the webs, they are getting involved in creating sub-scenarios for each of the existing scenarios and re-ranking them. If there is not any attachment hole on the webs, the attachment holes on the flanges are getting involved in creating sub-scenarios for each of the existing scenarios and re-ranking them. Then, the scenarios that are not at the highest rank are eliminated from the list, and the transformation is performed based on the highest ranked scenario or any random one of the scenarios at the highest rank if there are multiple scenarios at the highest rank. The presence of multiple scenarios with the highest rank will occur when the parts are very similar and symmetric at least about one plane.

If there are not any attachment flanges on the webs, the scenarios for pairing the attachment holes on the webs are created and ranked based on the number of superposed attachment holes. Then, the scenarios that are not at the highest rank are eliminated from the list, and the transformation is performed based on the highest ranked scenario or any random one of the scenarios at the highest rank if there are multiple scenarios at the highest rank. Here again, the presence of multiple scenarios with the highest rank could occur when the parts are polygonal or circular with organized and evenly distanced attachment holes.

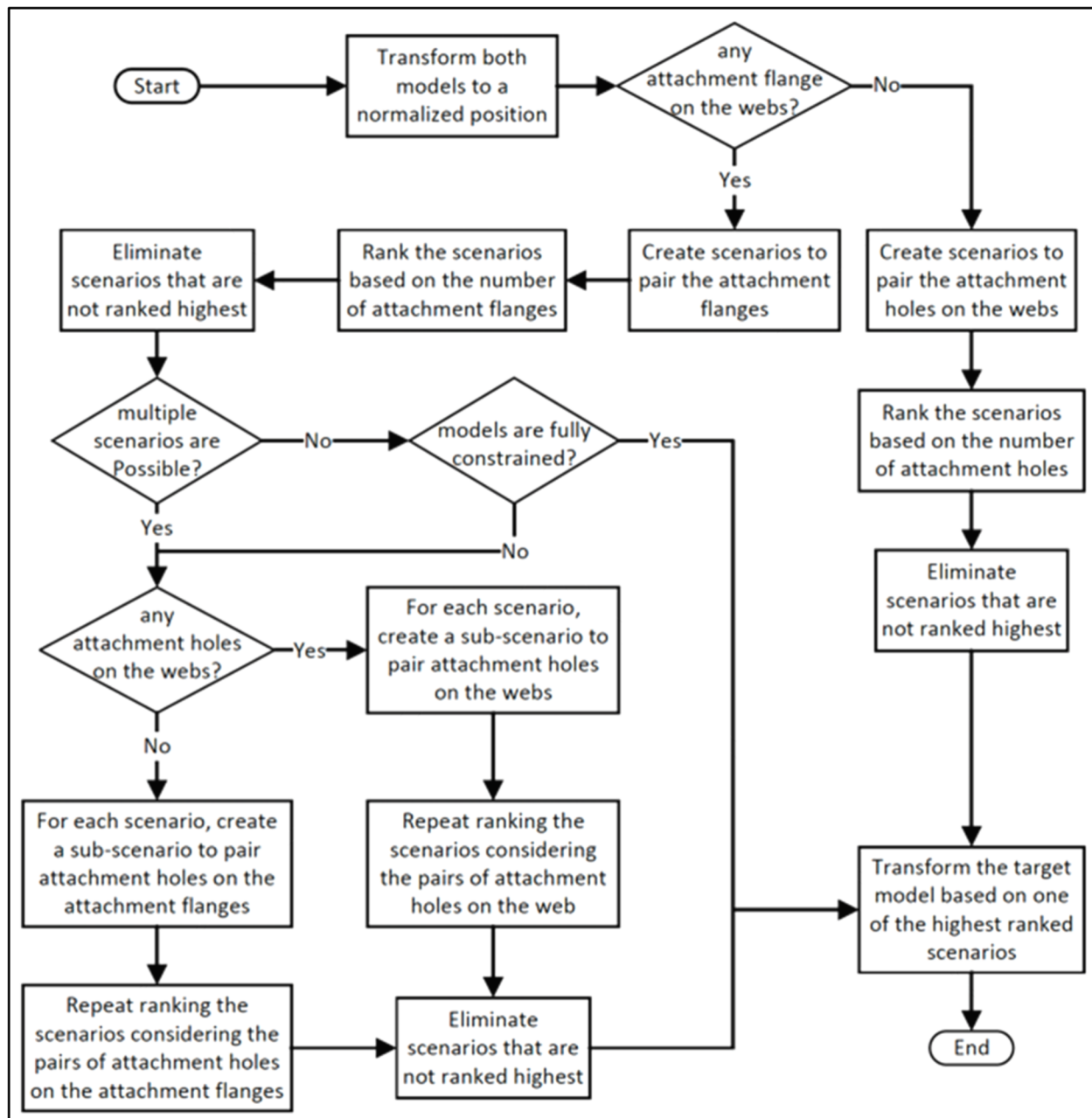


Figure-A XIV-1 The modified pose registration algorithm implemented in the prototype

Before describing MDI, it is worth representing the details of the scenario class declaration. This will be useful in case any future work tries to replicate the above-explained pose registration algorithm and needs guidance. Each scenario is instantiated from a scenario class with the following data members:

Scenario
<pre>Matrix44 transformation; std::vector <AliasPos> alias_position_list; int rank;</pre>

where:

1. *transformation* is a matrix that represents the transformation scenario.
2. *alias_position_list* is the list of the positions of the target model features involved in pose registration after the transformation.
3. *paired_features_list* is a list of the features pairs.
4. *rank* is an integer calculated for each scenario to compare scenarios one to another. The rank is calculated from comparing the alias-position to their equivalent feature position from the reference model.
5. *subscenarios_list* is a list of multiple scenarios that are possible for each scenario when more features get involved.

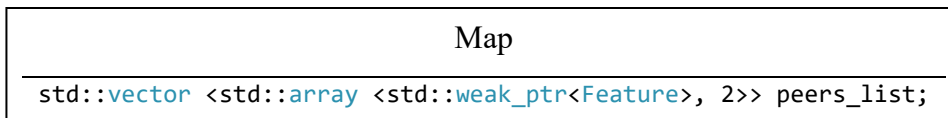
APPENDIX XV

MODEL DIFFERENCE IDENTIFICATION DETAILS

The model difference identification (MDI) algorithm implemented is slightly different than the one proposed in the theory paper (Chapter 4). During implementation, we opted for fewer steps and as a result the prototype was developed based on the following steps:

1. **Labelling and mapping commonality features:** Labelling and mapping the webs, as well as labelling and mapping any peer subordinate feature of the webs that have identical parameters as commonality features. Peer subordinate features of a feature are its children features in the same rank in feature structure.
2. **Labelling and mapping differed features:** Labelling and mapping any unmapped/unlabelled feature with one or more identical parameter as differed features.
3. **Labelling and mapping remaining differed features processing:** Labelling and mapping any unmapped/unlabelled feature with commonality subordinate, and labelling and mapping corners and bend reliefs with differed parent as differed features. Commonality subordinates are any immediate children or descendant features that are labelled commonality.
4. **Labelling added/removed features:** Labelling unmapped/unlabelled features of the reference model as removed features and labelling unmapped/unlabelled features of the target model as added features.

In this process, each feature is mapped by being added to an instance of the Map class defined by *peers_list*:



Labelling and mapping commonality features

Figure-A XV-1 represents the algorithm for labelling and mapping webs and their peer subordinate features that have identical parameters. The flowchart is segmented and is exemplified by the comparison represented in Figure-A XV-2 which comes from the parts in Figure 6.3. The segments of the flowchart and their corresponding elements in the example are colored the same.

Web labelling and mapping: The algorithm implemented to label and map the webs and their peer subordinate features that have identical parameters starts with checking if the webs have identical parameters and labelling them accordingly. The webs are added to the list of mapped features regardless of being commonality or not. These steps are colored red in Figure-A XV-1

matching the red elements indicating labelling and mapping webs in Figure-A XV-2. Two lists of the webs' children features are created.

Subordinate features labelling and mapping: Next, each of the features on the list is checked against the features of the other list to identify the ones that are of the same type and have identical parameters. If so, they are labelled as commonality and added to the list of mapped features. These steps are colored blue in Figure-A XV-1 matching the blue elements in Figure-A XV-2; including the elements indicating failing to label and map an attachment flange with features of the compared part and succeeding for an attachment hole.

Recursive feature labelling and mapping: Whether or not the two compared features are identified as commonality, if they have subordinate features, all of their subordinate features will also be evaluated against their peers in turn to identify all possible commonality peer features before checking the next subordinate feature of the webs. This recursion is abstracted by the steps colored green in Figure-A XV-1 to match the green elements in Figure-A XV-2 which indicate the labelling and mapping the deformed flange as a subordinate feature of the attachment flange and its subordinate joggle.

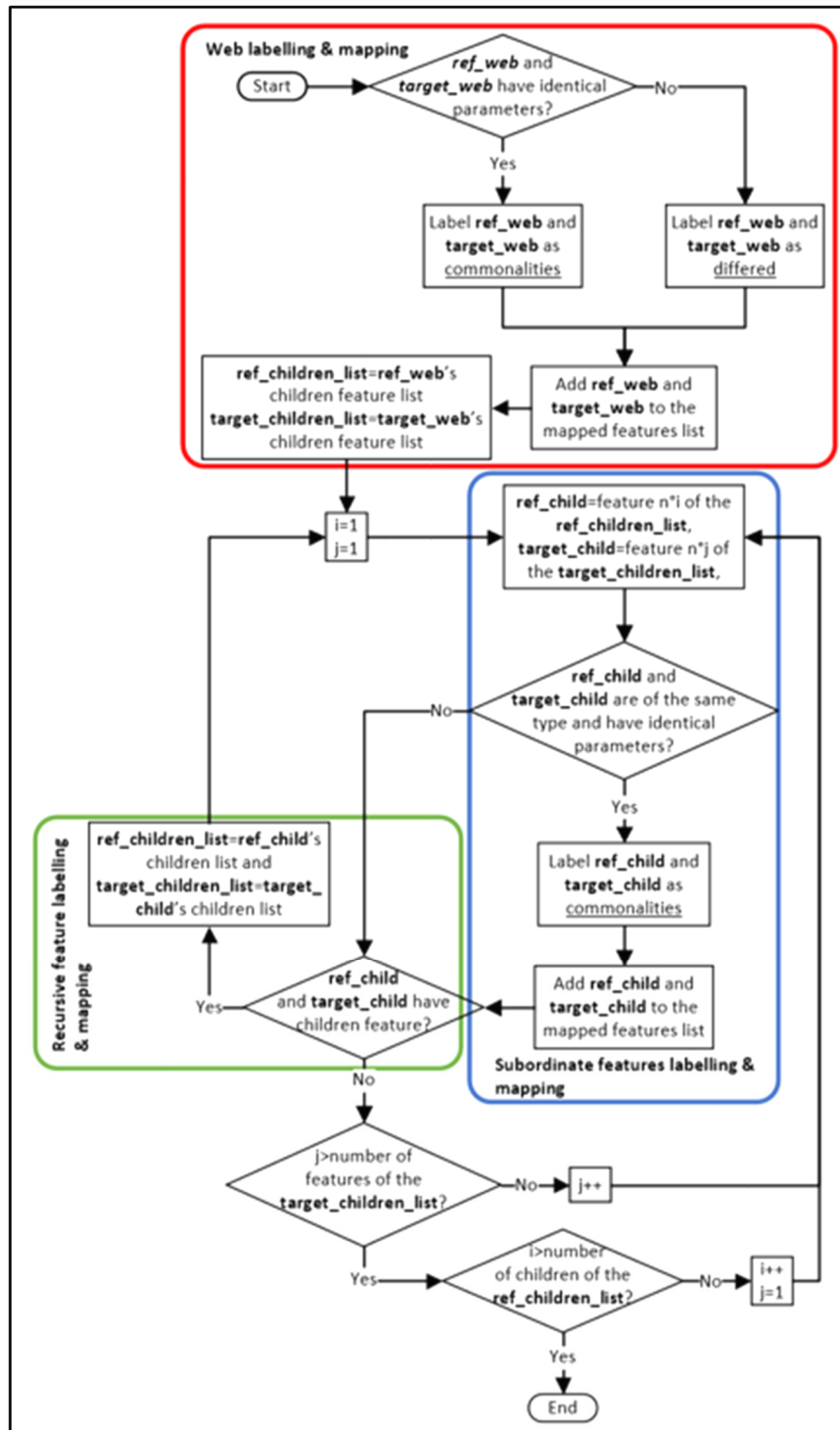


Figure-A XV-1 The algorithm for labelling and mapping webs and their peer subordinate features that have identical parameters

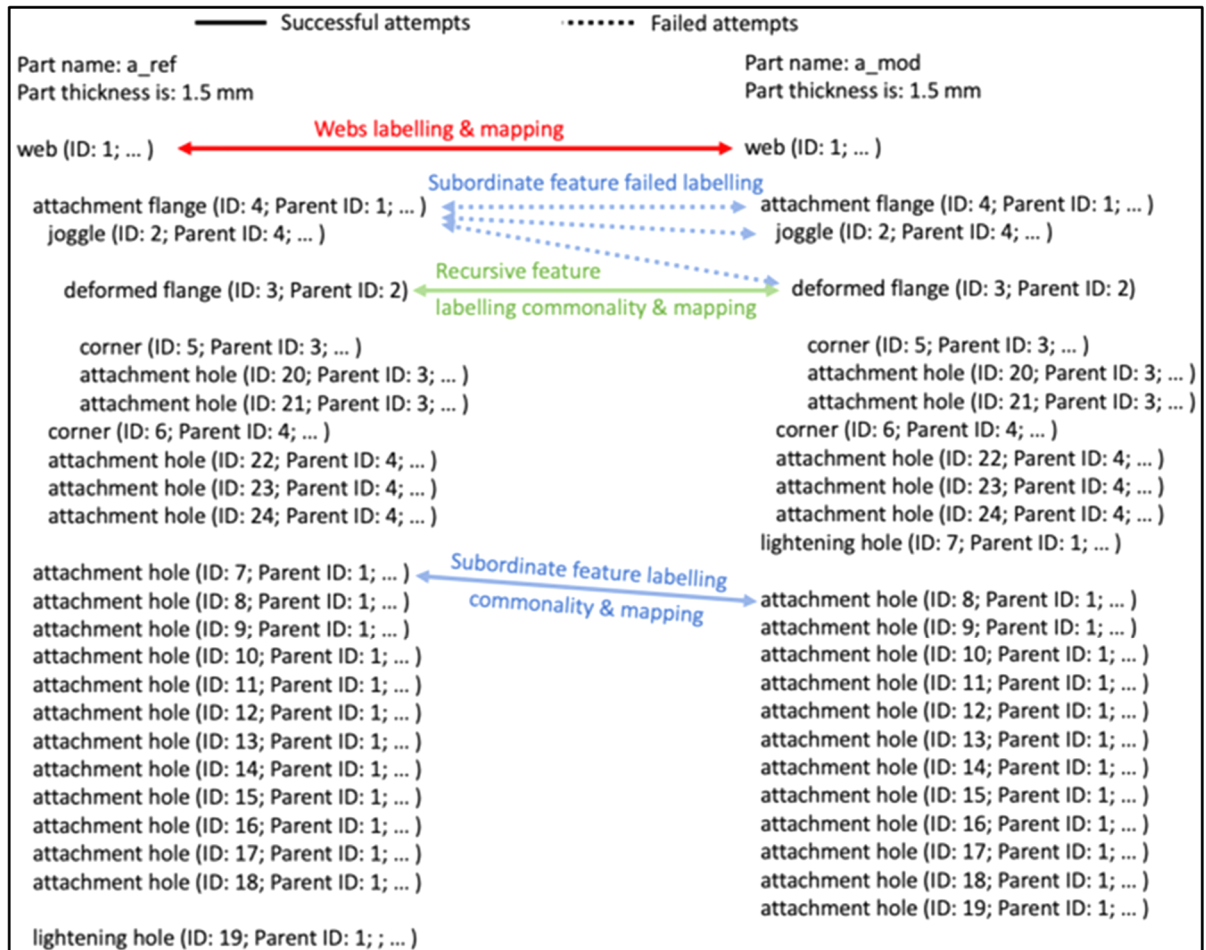


Figure-A XV-2 An example of webs and their peer subordinate features that have identical parameters being labelled commonality and mapped

Labelling and mapping differed features

Figure-A XV-3 represents the algorithm for labelling and mapping differed features. The flowchart is segmented and is exemplified by the comparison of the parts represented in Figure-A XV-4 which comes from the parts in Figure 6.4. The segments of the flowchart and their corresponding elements in the example are colored the same when needed.

Setup: The algorithm implemented to map features with at least one identical parameter starts with checking any of the features of the reference model that is not labelled. Each of such unlabelled features is checked against each of the features of the target model that are also not labelled. An integer variable, here called *Max*, to store the highest number of shared parameters

between the compared features is declared and equated to zero at the beginning. These steps are colored red in Figure-A XV-3. Note that if Max remains equal to zero for a feature in the reference model, it means that there is no feature in the target model that has any similarity with it, hence it is not a differed feature.

Type check and parameter count: If these unlabelled features are of the same type, their identical parameters are counted and stored as *NBR*. Each time a feature is found in target model that has more parameters in common with the feature in reference model, the value of *Max* is changed to the *NBR* and the feature in target model is registered as *peer_feature*. These steps are colored blue in Figure-A XV-3 matching the blue elements indicating type checking and parameter counting between compared features in Figure-A XV-4.

Labelling and mapping: When each unlabelled feature is finished being compared against all of the unlabelled features of the target model, if Max is not equal to zero, this feature and the *peer_feature* (registered previously) are labelled differed and mapped. These steps are colored green in Figure-A XV-3 matching the green elements indicating labelling and mapping the differed features in Figure-A XV-4.

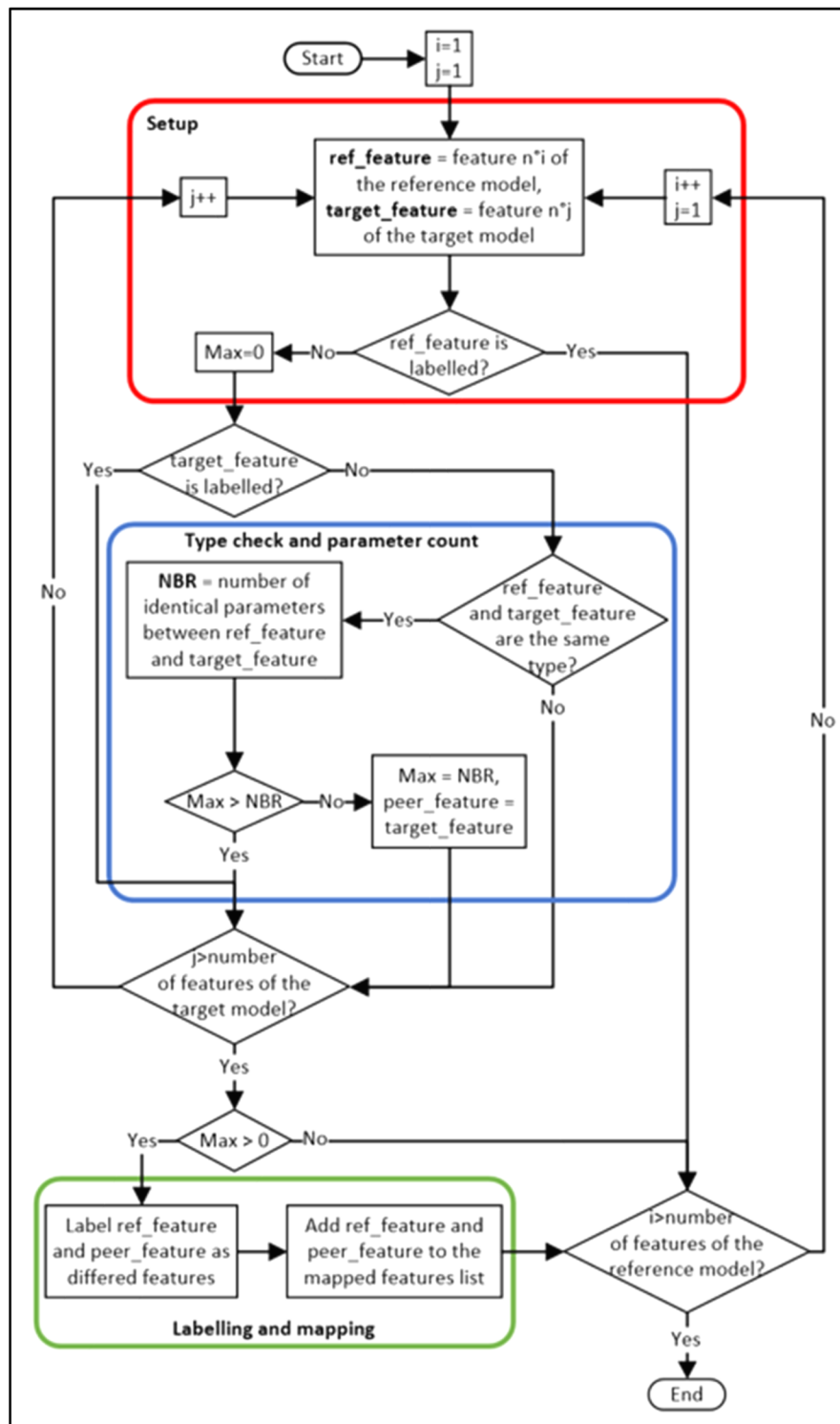


Figure-A XV-3 The algorithm for labelling and mapping differed features

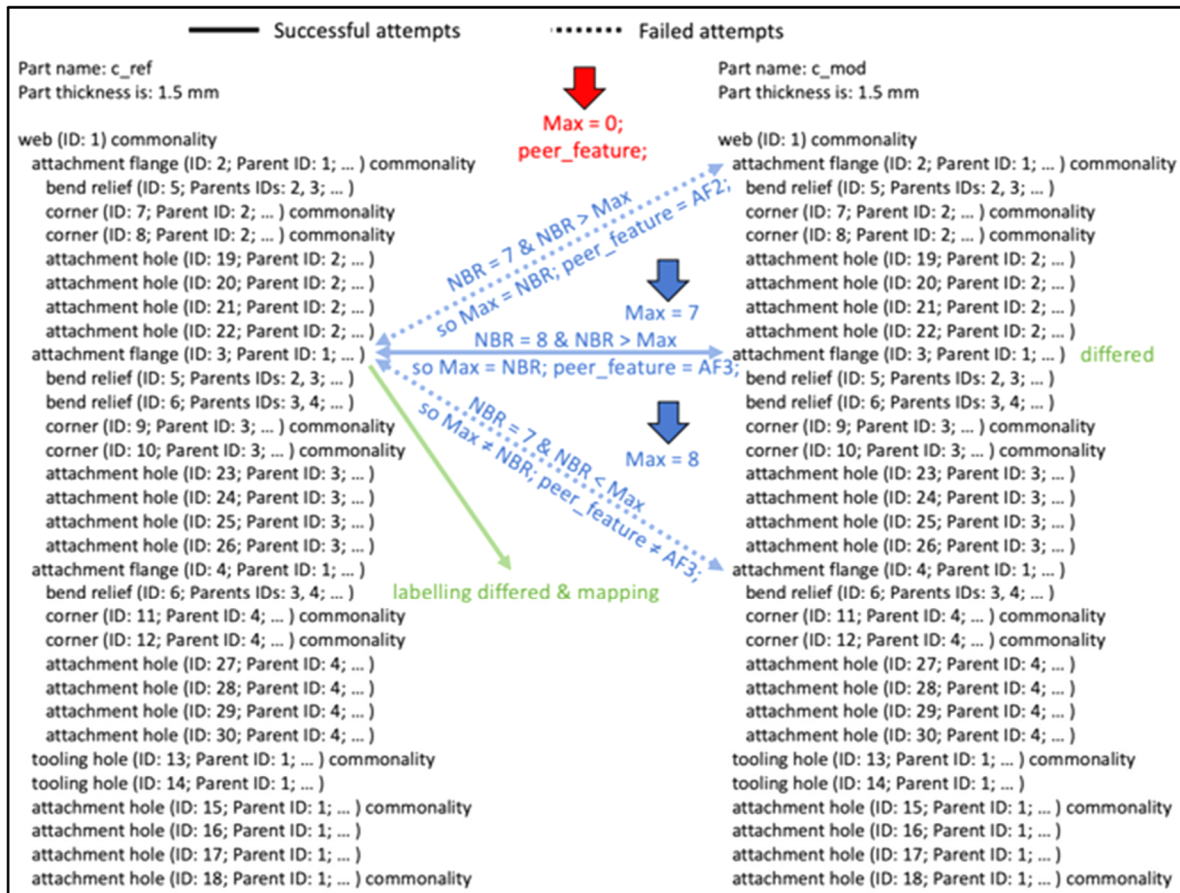


Figure-A XV-4 An example of features that have a number of identical parameters being labelled differed and mapped

Labelling and mapping remaining differed features

Figure-A XV-5 represents the algorithm implemented for labelling and mapping any unlabelled feature with commonality subordinate, as well as corners and bend reliefs with differed parents as differed features. The flowchart is segmented and is exemplified by the comparison represented in Figure-A XV-6 which comes from the parts in Figure 6.3. The segments of the flowchart and their corresponding elements in the example are colored the same when needed.

Type check and subordinate commonality finding: The algorithm starts with checking any of the features of the reference model that is not labelled. Each of such unlabelled features is checked against each of the features of the target model that are also not labelled. If these

unlabelled features are of the same type, their subordinate features are checked for any commonalities. These steps are colored red in Figure-A XV-5.

Eligible corners and bend reliefs finding: If these unlabelled features are verified to not have commonality subordinate features, they are checked for both being corners or bend reliefs. If so, their parent features are checked for being differed features. These steps are colored blue in Figure-A XV-5.

Labelling and mapping: In both above cases, the unlabelled features are labelled differed and mapped by adding them to the list of mapped features. These steps are colored green in Figure-A XV-5.

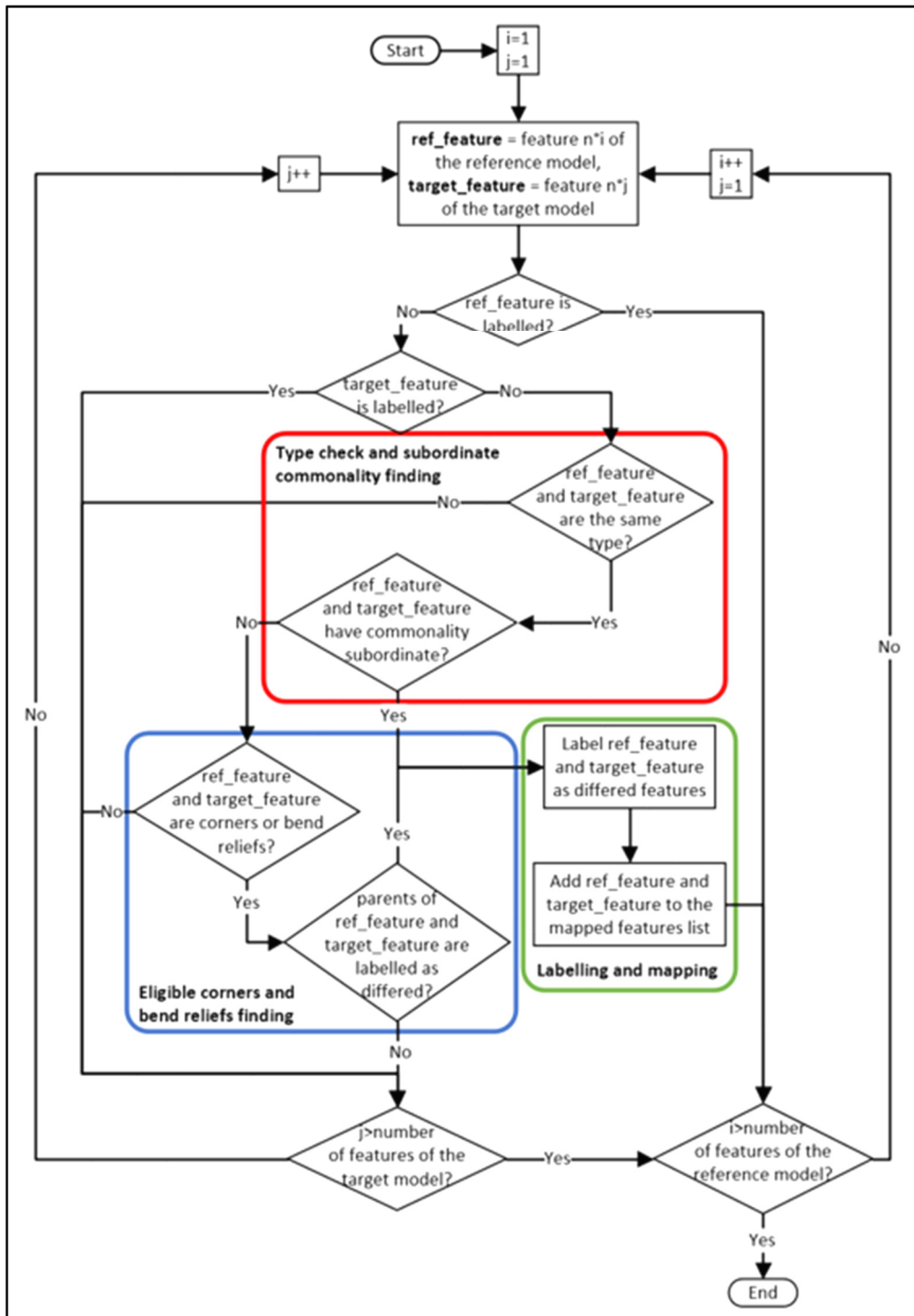


Figure-A XV-5 The algorithm implemented for labelling and mapping any unlabelled feature with commonality subordinate, as well as corners and bend reliefs with differed parents

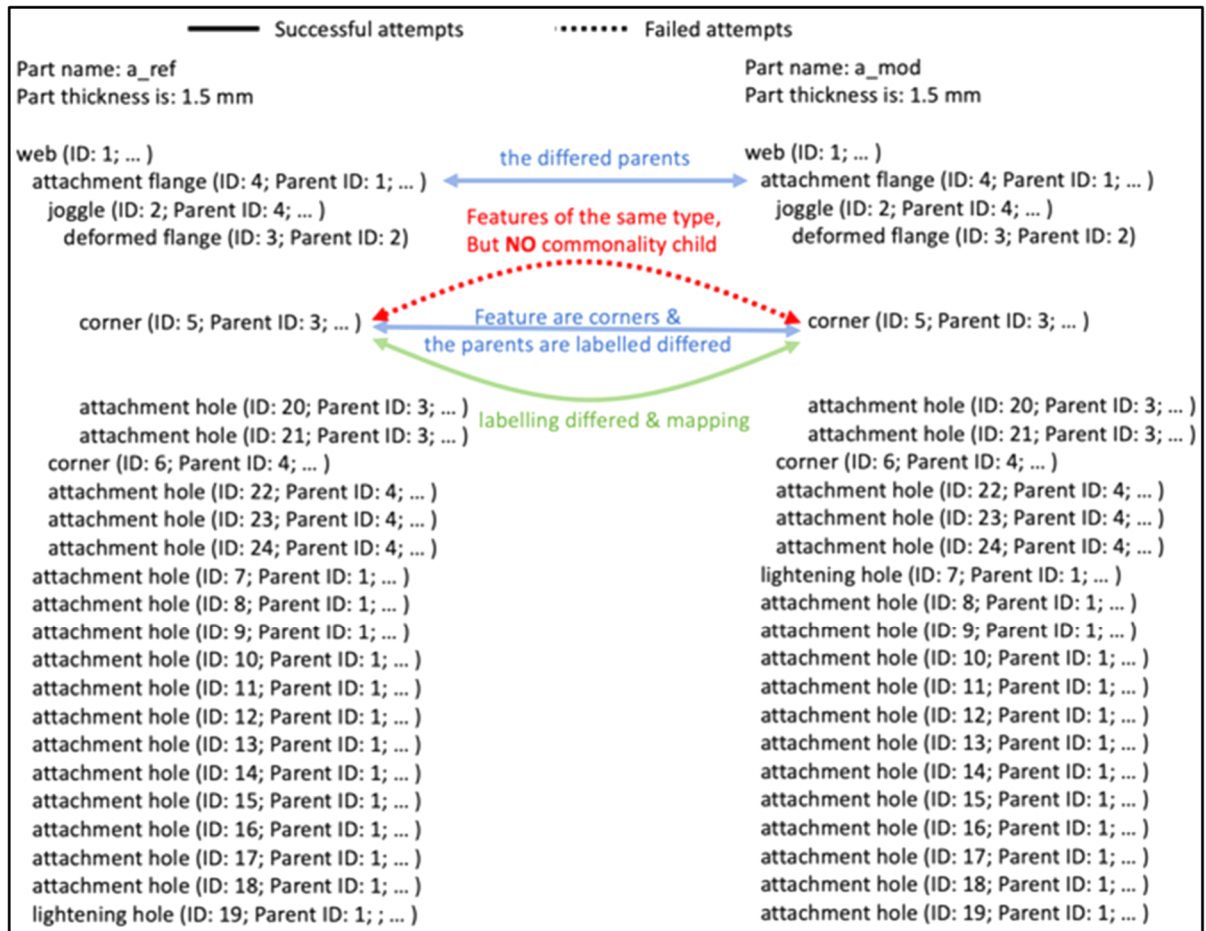


Figure-A XV-6 An example of corners with differed parents being labelled differed and mapped

Labelling added/removed features

The algorithm implemented to label unlabelled features of the reference model as removed features and labelling unlabelled features of the target model as added features starts with all of the unlabelled features of each model and labeling each feature accordingly. This step, being the last one does not include any calculations, and does not include mapping.

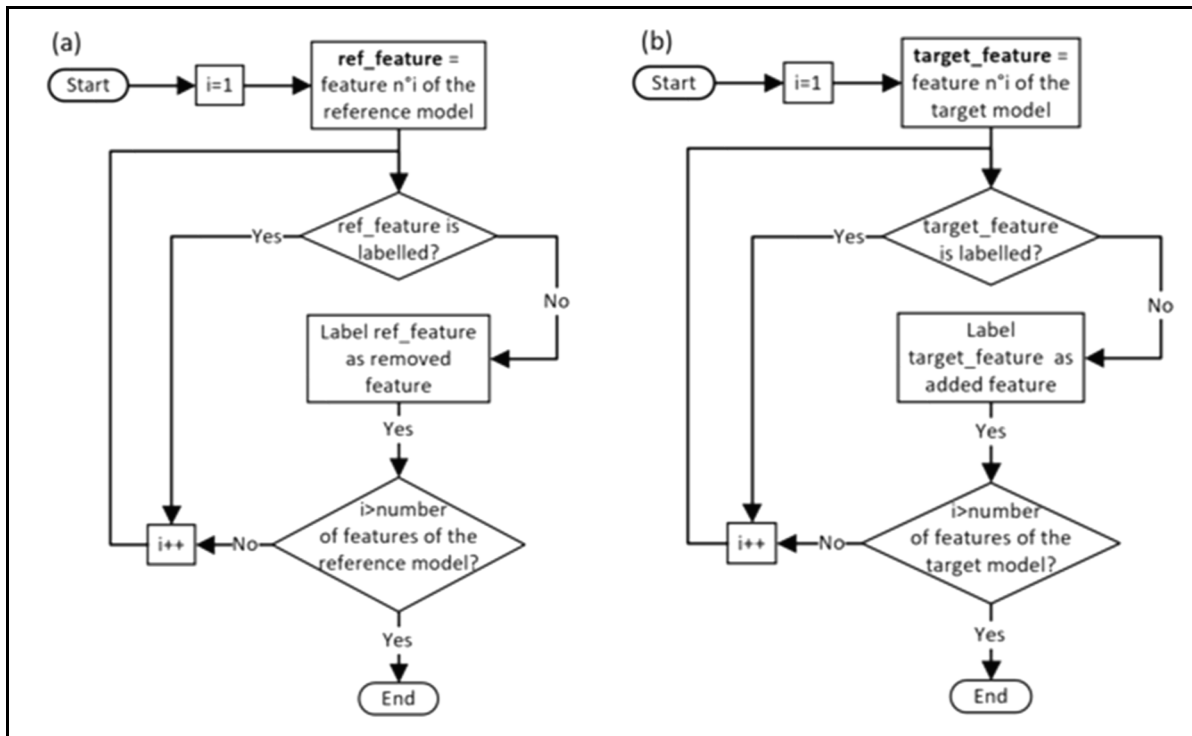


Figure-A XV-7 The algorithms to label (a) removed features of the reference model and (b) added features of the target model

LIST OF REFERENCES

- Alibre LLC. (2017). Alibre Design. Repéré à www.alibre.com
- Aspert, N., Santa-Cruz, D., & Ebrahimi, T. (2002). Mesh: Measuring errors between surfaces using the hausdorff distance. Dans *Proceedings. IEEE international conference on multimedia and expo* (Vol. 1, pp. 705-708). IEEE.
- Autodesk. (2018). Inventor. Repéré à www.autodesk.ca
- Babic, B. (1996). Development of an intelligent CAD-CAPP interface. Dans *Proceedings of the International Conference on Intelligent Technologies in Human-Related Sciences* (pp. 351-357).
- Babic, B., & Miljkovic, Z. (1997). Feature recognition as the basis for integration of CAD and CAPP Systems. Dans *The second world congress on intelligent manufacturing processes & systems (Budapest, 10-13 June 1997)* (pp. 596-601).
- Babic, B., Nesic, N., & Miljkovic, Z. (2008). A review of automated feature recognition with rule-based pattern recognition. *Computers in Industry*, 59(4), 321-337. doi: <https://doi.org/10.1016/j.compind.2007.09.001>. Repéré à <https://www.sciencedirect.com/science/article/pii/S0166361507001327>
- Babić, B. R., Nešić, N., & Miljković, Z. (2011). Automatic feature recognition using artificial neural networks to integrate design and manufacturing: Review of automatic feature recognition systems. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 25(3), 289-304. doi: <https://doi.org/10.1017/S0890060410000545>
- Besl, P. J., & McKay, N. D. (1992). Method for registration of 3-D shapes. Dans *Sensor Fusion IV: Control Paradigms and Data Structures* (Vol. 1611, pp. 586-607). International Society for Optics and Photonics. doi: <https://doi.org/10.1117/12.57955>
- Bouzakis, H., & Andreadis, G. (2000). A feature-based algorithm for computer aided process planning for prismatic parts. *International Journal of Production Engineering and Computers*, 3(3), 17-22.
- Bricsys. (2017). BricsCAD Platinum. Repéré à www.bricsys.com
- Brière-Côté, A., Rivest, L., & Maranzana, R. (2011). A Three-Step Approach to Structuring 3D CAD Model Comparison Scenarios. *IFIP (ed.) IFIP WG5*, 429-443.

- Brière-Côté, A., Rivest, L., & Maranzana, R. (2012a). Comparing 3d cad models: Uses, methods, tools and perspectives. *Computer-Aided Design and Applications*, 9(6), 771-794. doi: <http://doi.org/10.3722/cadaps.2012.771-794>
- Brière-Côté, A., Rivest, L., & Maranzana, R. (2012b). Integration of explicit geometric constraints in the comparison of 3D CAD models for part design reuse. Dans *IFIP International Conference on Product Lifecycle Management* (pp. 445-457). Springer.
- Brière-Côté, A., Rivest, L., & Maranzana, R. (2013). 3D CAD Model Comparison: An Evaluation of Model Difference Identification Technologies. *Computer-Aided Design and Applications*, 10(2), 173-195. doi: <http://doi.org/10.3722/cadaps.2013.173-195>
- Brunetti, G., & Grimm, S. (2005). Feature ontologies for the explicit representation of shape semantics. *International Journal of Computer Applications in Technology*, 23(2-4), 192-202. doi: <https://doi.org/10.1504/IJCAT.2005.006481>
- Campana, G., & Mele, M. (2020). An application to Stereolithography of a feature recognition algorithm for manufacturability evaluation. *Journal of Intelligent Manufacturing*, 31(1), 199-214. doi: <https://doi.org/10.1007/s10845-018-1441-8>
- Cardone, A., Gupta, S. K., & Karnik, M. (2003). A survey of shape similarity assessment algorithms for product design and manufacturing applications. *J. Comput. Inf. Sci. Eng.*, 3(2), 109-118.
- Chatelain, J.-F., Maranzana, R., & St-Martin, S. (2002). A solid model comparison approach based on a model tree analysis. Dans *4th WSEAS International Conference on Mechanical Engineering*. Citeseer.
- Chen, X., Guo, S., Bai, J., & Gao, S. (2010). Assembly Retrieval in Top-down Product Design. *한국 CDE 학회 국제 학술 발표 논문집*, 39-52.
- Cheng, Z., & Ma, Y. (2017). A functional feature modeling method. *Advanced Engineering Informatics*, 33, 1-15. doi: <https://doi.org/10.1016/j.aei.2017.04.003>
- Chowdhury, S., & Siddique, Z. (2009). Development and implementation of shape comparison of 3D CAD models. Dans *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* (Vol. 48999, pp. 1085-1094).
- Chu, C.-H., & Hsu, Y.-C. (2006). Similarity assessment of 3D mechanical components for design reuse. *Robotics and Computer-Integrated Manufacturing*, 22(4), 332-341. doi: <https://doi.org/10.1016/j.rcim.2005.07.005>

- Cicirello, V., & Regli, W. C. (2001). Machining feature-based comparisons of mechanical parts. Dans *SMI*. IEEE. doi: <http://doi.ieeecomputersociety.org/10.1109/SMA.2001.923388>
- Cicirello, V. A., & Regli, W. C. (2002). An approach to a feature-based comparison of solid models of machined parts. *AI EDAM*, 16(5), 385-399. doi: <https://doi.org/10.1017/S0890060402165048>
- Cuillière, J.-C., François, V., Souaissa, K., Benamara, A., & Bel Hadj Salah, H. (2011). Automatic comparison and remeshing applied to CAD model modification. *Computer-Aided Design*, 43(12), 1545-1560.
- Dassault Systems. CATIA 3DExperience 2017X. Repéré à https://cloud.academy.3ds.com/r2017x/role_smh.html
- Dassault Systems. (2017). CATIA V5-V6. Repéré à www.3ds.com/products-services/catia/products/v5/portfolio/
- Dassault Systems. (2018). SOLIDWORKS. Repéré à www.solidworks.com
- Dassault Systems. (2019). FeatureWorks. Repéré à https://help.solidworks.com/2021/english/SolidWorks/fworks/IDH_OPTIONS.htm
- Dawei, W., Guangrong, Y., Yi, L., & Zhang, J. (2012). A retrieval algorithm of sheet metal parts based on relationships of features. *Chinese Journal of Aeronautics*, 25(3), 453-472.
- Devarajan, M., Kamran, M., & Nnaji, B. O. (1997). Profile offsetting for feature extraction and feature tool mapping in sheet metal. *International Journal of Production Research*, 35(6), 1593-1608. doi: <https://doi.org/10.1080/002075497195146>
- Ding, B., Yu, X.-y., & Liu, L. (2014). 3D CAD Model Representation and Retrieval based on Hierarchical Graph. *JSW*, 9(10), 2499-2506.
- Ding, B., Zhang, Z., Yu, X.-y., & He, Y.-b. (2013). 3D CAD model retrieval based on GA-ACO. Dans *Ifost* (Vol. 2, pp. 36-41). IEEE.
- Elinson, A., Nau, D. S., & Regli, W. C. (1997). Feature-based similarity assessment of solid models. Dans *Proceedings of the fourth ACM symposium on Solid modeling and applications* (pp. 297-310). Association for Computing Machinery. doi: <https://doi.org/10.1145/267734.267806>

- Fukunaga, K. (1990). *Introduction to statistical pattern recognition* (Second edition éd.). Elsevier.
- Gao, W., Gao, S., & Liu, Y. (2005). 3D CAD model similarity assessment and retrieval using DBS. Dans *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* (Vol. 47403, pp. 211-219).
- Gao, W., Gao, S., Liu, Y., Bai, J., & Hu, B. (2006). Multiresolutional similarity assessment and retrieval of solid models based on DBMS. *Computer-Aided Design*, 38(9), 985-1001.
- Gao, X.-Y., Li, J.-W., & Zhang, C.-X. (2020). Similarity Calculation of 3D Model By Integrating Improved ACO Into HNN. *IEEE Access*, 8, 155378-155388.
- Gao, X.-Y., Zhang, C.-X., & Lu, Z.-M. (2015). Compute Similarity of CAD Models Based on Bipartite Graph. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 8(2), 49-56.
- Gear, P. J., & Visser, B. J. (2003). Method of comparing parts: Google Patents.
- Ghaffarishahri, S., & Rivest, L. (2019). Feature Recognition for Structural Aerospace Sheet Metal Parts. *Computer-Aided Design and Applications*.
- Ghaffarishahri, S., & Rivest, L. (2020). Feature Recognition for Structural Aerospace Sheet Metal Parts. *Computer-Aided Design & Applications*, 17(1), 16-43. doi: <https://doi.org/10.14733/cadaps.2020.16-43>
- Ghaffarishahri, S., & Rivest, L. (2021). Feature-Based Model Difference Identification for Aerospace Sheet Metal Parts. *Computer-Aided Design & Applications*, 18(3), 443-467. doi: <https://doi.org/10.14733/cadaps.2021.443-467>
- Gupta, R. K., & Gurumoorthy, B. (2012). Automatic extraction of free-form surface features (FFSFs). *Computer-Aided Design*, 44(2), 99-112. doi: <https://doi.org/10.1016/j.cad.2011.09.012>
- Gupta, R. K., & Gurumoorthy, B. (2013). Classification, representation, and automatic extraction of deformation features in sheet metal parts. *Computer-Aided Design*, 45(11), 1469-1484. doi: <https://doi.org/10.1016/j.cad.2013.06.010>
- Han, J., & Requicha, A. A. (1997). Integration of feature based design and feature recognition. *Computer-Aided Design*, 29(5), 393-403.

- Han, Z., Mo, R., & Hao, L. (2019). Clustering and retrieval of mechanical CAD assembly models based on multi-source attributes information. *Robotics and Computer-Integrated Manufacturing*, 58, 220-229.
- Han, Z., Mo, R., Yang, H., & Hao, L. (2018). CAD assembly model retrieval based on multi-source semantics information and weighted bipartite graph. *Computers in Industry*, 96, 54-65.
- Harik, R. F., & Barakat, F. G. (2010). Part Similarity Algorithm Based on Manufacturing Features Similarity. *Computer-Aided Design and Applications*, 7(5), 663-674.
- HCL Technologies. (2013). Sheet Metal Feature Recognition Library. Repéré à <https://feature.geometricglobal.com/feature-recognition-suite/sheet-metal-feature-recognition-library/>
- Henderson, M. R., & Anderson, D. C. (1984). Computer recognition and extraction of form features: a CAD/CAM link. *Computers in Industry*, 5(4), 329-339.
- Hicks, B. J., Culley, S. J., Allen, R. D., & Mullineux, G. (2002). A framework for the requirements of capturing, storing and reusing information and knowledge in engineering design. *International Journal of Information Management*, 22(4), 263-280. doi: [https://doi.org/10.1016/S0268-4012\(02\)00012-9](https://doi.org/10.1016/S0268-4012(02)00012-9)
- Hong, T., Lee, K., Kim, S., Chu, C., & Lee, H. (2005). Similarity Comparison of Mechanical Parts. *Computer-Aided Design and Applications*, 2(6), 759-768. doi: <https://doi.org/10.1080/16864360.2005.10738339>
- Hou, J. P. S. J. S., & Ramani, K. (2006). 3D CAD model retrieval based on multiple levels of detail.
- Huang, R., Zhang, S., Bai, X., Xu, C., & Huang, B. (2015). An effective subpart retrieval approach of 3D CAD models for manufacturing process reuse. *Computers in Industry*, 67, 38-53.
- Huangfu, Z.-M., Zhang, S.-S., & Yan, L.-H. (2017). A method of 3D CAD model retrieval based on spatial bag of words. *Multimedia Tools and Applications*, 76(6), 8145-8173.
- IMSI/Design. (2017). TurboCAD. Repéré à www.turbocad.com
- Industrial automation systems and integration -- Product data representation and exchange - Part 42: Integrated generic resource: Geometric and topological representation - STEP 10303-42.* (2003). International Organization for Standardization.

- IronCAD LLC. (2017). IronCAD. Repéré à www.ironcad.com
- Ismail, N., Bakar, N. A., & Juri, A. (2002). Feature recognition patterns for form features using boundary representation models. *The International Journal of Advanced Manufacturing Technology*, 20(8), 553-556.
- Ismail, N., Bakar, N. A., & Juri, A. (2005). Recognition of cylindrical and conical features using edge boundary classification. *International Journal of Machine Tools and Manufacture*, 45(6), 649-655.
- Jagirdar, R., Jain, V. K., & Batra, J. L. (2001). Characterization and identification of forming features for 3-D sheet metal components. *International Journal of Machine Tools and Manufacture*, 41(9), 1295-1322. doi: [https://doi.org/10.1016/S0890-6955\(01\)00006-2](https://doi.org/10.1016/S0890-6955(01)00006-2)
- Jagirdar, R., Jain, V. K., Batra, J. L., & Dhande, S. G. (1995). Feature recognition methodology for shearing operations for sheet metal components. *Computer Integrated Manufacturing Systems*, 8(1), 51-62. doi: [https://doi.org/10.1016/0951-5240\(95\)92813-A](https://doi.org/10.1016/0951-5240(95)92813-A)
- Jain, P. K., & Kumar, S. (1998). Automatic feature extraction in PRIZCAPP. *International Journal of Computer Integrated Manufacturing*, 11(6), 500-512.
- Jeon, S. M., Lee, J. H., Hahm, G. J., & Suh, H. W. (2016). Automatic CAD model retrieval based on design documents using semantic processing and rule processing. *Computers in Industry*, 77, 29-47.
- Joshi, S. B. (1987). *CAD interface for automated process planning* (Purdue University).
- Kannan, T. R., & Shunmugam, M. S. (2009a). Processing of 3D sheet metal components in STEP AP-203 format. Part I: feature recognition system. *International Journal of Production Research*, 47(4), 941-964. doi: <https://doi.org/10.1080/00207540701510055>
- Kannan, T. R., & Shunmugam, M. S. (2009b). Processing of 3D sheet metal components in STEP AP-203 format. Part II: feature reasoning system. *International Journal of Production Research*, 47(5), 1287-1308. doi: <https://doi.org/10.1080/00207540701510063>
- Kim, B. C., & Mun, D. (2015). Enhanced volume decomposition minimizing overlapping volumes for the recognition of design features. *Journal of Mechanical Science and Technology*, 29(12), 5289-5298. doi: <https://doi.org/10.1007/s12206-015-1131-9>

- Kim, H., Cha, M., & Mun, D. (2017). Shape distribution-based approach to comparing 3D CAD assembly models. *Journal of Mechanical Science and Technology*, 31(12), 5627-5638.
- Kim, J., Pratt, M. J., Iyer, R. G., & Sriram, R. D. (2008). Standardized data exchange of CAD models with design intent. *Computer-Aided Design*, 40(7), 760-777.
- Kim, Y. S. (1992). Recognition of form features using convex decomposition. *Computer-Aided Design*, 24(9), 461-476.
- Lai, J.-Y., Wang, M.-H., Song, P.-P., Hsu, C.-H., & Tsai, Y.-C. (2018). Automatic recognition and decomposition of rib features in thin-shell parts for mold flow analysis. *Engineering with Computers*, 34(4), 801-820. doi: <https://doi.org/10.1007/s00366-017-0574-2>
- Langerak, T. R. (2010). Local parameterization of freeform shapes using freeform feature recognition. *Computer-Aided Design*, 42(8), 682-692. doi: <https://doi.org/10.1016/j.cad.2010.02.004>
- Li, M., Fuh, J., Zhang, Y., & Qiu, Z. (2008). General and partial shape matching approaches on feature-based CAD models to support efficient part retrieval. Dans *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* (Vol. 43277, pp. 121-130).
- Li, M., Zhang, Y., & Fuh, J. (2010). Retrieving reusable 3D CAD models using knowledge-driven dependency graph partitioning. *Computer-Aided Design and Applications*, 7(3), 417-430.
- Li, M., Zhang, Y., Fuh, J., & Qiu, Z. (2009). Toward effective mechanical design reuse: CAD model retrieval based on general and partial shapes. *Journal of Mechanical Design*, 131(12).
- Li, M., Zhang, Y., Fuh, J. Y., & Qiu, Z. (2011). Design reusability assessment for effective CAD model retrieval and reuse. *International Journal of Computer Applications in Technology*, 40(1-2), 3-12.
- Li, Y. G., Ding, Y. F., Mou, W. P., & Guo, H. (2009). Feature recognition technology for aircraft structural parts based on a holistic attribute adjacency graph. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 224(2), 271-278. doi: <https://doi.org/10.1243/09544054JEM1634>. Repéré à <https://doi.org/10.1243/09544054JEM1634>

- Li, Y. G., Ding, Y. F., Mou, W. P., & Guo, H. (2010). Feature recognition technology for aircraft structural parts based on a holistic attribute adjacency graph. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 224(2), 271-278. doi: [doi:10.1243/09544054JEM1634](https://doi.org/10.1243/09544054JEM1634). Repéré à <http://journals.sagepub.com/doi/abs/10.1243/09544054JEM1634>
- Li, Z., Zhou, X., & Liu, W. (2015). A geometric reasoning approach to hierarchical representation for B-rep model retrieval. *Computer-Aided Design*, 62, 190-202.
- Liu, J., Liu, X., Ni, Z., & Zhou, H. (2018). A new method of reusing the manufacturing information for the slightly changed 3D CAD model. *Journal of Intelligent Manufacturing*, 29(8), 1827-1844. doi: <https://doi.org/10.1007/s10845-016-1220-3>
- Liu, Y.-J., Luo, X., Joneja, A., Ma, C.-X., Fu, X.-L., & Song, D. (2013). User-adaptive sketch-based 3-D CAD model retrieval. *IEEE Transactions on Automation Science and Engineering*, 10(3), 783-795.
- Liu, Z., Li, J., Wang, Y., Li, C., & Xiao, X. (2004). Automatically extracting sheet-metal features from solid model. *Journal of Zhejiang University-Science A*, 5(11), 1456-1465. doi: <https://doi.org/10.1631/jzus.2004.14>
- Lupinetti, K., Pernot, J.-P., Monti, M., & Giannini, F. (2019). Content-based CAD assembly model retrieval: Survey and future challenges. *Computer-Aided Design*, 113, 62-81.
- Ma, W., Wang, P., Cai, D., & Wang, D. (2019). Research on 3D CAD Model Retrieval Algorithm Based on Global and Local Similarity. Dans *2019 IEEE International Conferences on Ubiquitous Computing & Communications (IUCC) and Data Science and Computational Intelligence (DSCI) and Smart Computing, Networking and Services (SmartCNS)* (pp. 349-355). IEEE.
- Malyshev, A., Slyadnev, S., & Turlapov, V. (2017). Graph-based feature recognition and suppression on the solid models. Dans *GraphiCon* (Vol. 2017, pp. 319-322).
- McWherter, D., Peabody, M., Shokoufandeh, A. C., & Regli, W. (2001). Database techniques for archival of solid models. Dans *Proceedings of the sixth ACM symposium on Solid modeling and applications* (pp. 78-87). Association for Computing Machinery. doi: <https://doi.org/10.1145/376957.376968>
- Messmer, B. T., & Bunke, H. (1995). *Subgraph isomorphism in polynomial time*. Citeseer.
- Min, L. (2011). *Similarity Assessment and Retrieval of CAD Models* (National University of Singapore).

- Msaaf, O., Maranzana, R., & Rivest, L. (2007). Part data mining for information re-use in a PLM context. *Proceedings of GT2007, May*, 14-17.
- Niu, C. (1999). *Airframe structural design: practical design information and data on aircraft structures*. Hong Kong: Connilit Press Limited.
- Nnaji, B. O., Kang, T.-S., Yeh, S., & Chen, J.-P. (1991). Feature reasoning for sheet metal components. *International Journal of Production Research*, 29(9), 1867-1896. doi: <https://doi.org/10.1080/00207549108948055>
- Onshape. Onshape. Repéré à www.onshape.com
- Paterson, D., & Corney, J. (2016). Feature based search of 3D databases. Dans *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* (Vol. 50084, pp. V01BT02A010). American Society of Mechanical Engineers.
- Pottmann, H., Leopoldseder, S., & Hofer, M. (2004). Registration without ICP. *Computer Vision and Image Understanding*, 95(1), 54-71. doi: <https://doi.org/10.1016/j.cviu.2004.04.002>
- Pratt, M. J. (2001). Introduction to ISO 10303—the STEP standard for product data exchange. *Journal of Computing and Information Science in Engineering*, 1(1), 102-103.
- PRO/Engineer Wildfire 5.0. (2009) (Version Version 5.0): Parametric Technology Corporation. Repéré à <https://www.ptc.com/en/support/article/CS142741>
- PTC. (2017). Creo Parametric. Repéré à www.ptc.com/en/products/creo/parametric
- Qin, F., Gao, S., Yang, X., Li, M., & Bai, J. (2016). An ontology-based semantic retrieval approach for heterogeneous 3D CAD models. *Advanced Engineering Informatics*, 30(4), 751-768.
- Reddy, K. J., Adithan, M., & Radhakrishnan, P. (2011). Development of a methodology for retrieval of similarly shaped CAD models. *International Journal of Computer Applications in Technology*, 40(4), 288-294.
- Roy, U., & Bharadwaj, B. (2002). Design with part behaviors: behavior model, representation and applications. *Computer-Aided Design*, 34(9), 613-636. doi: [https://doi.org/10.1016/S0010-4485\(01\)00129-4](https://doi.org/10.1016/S0010-4485(01)00129-4)

- Shi, M., & Zhang, S. (2013). A method of 3D CAD model retrieval based on feature adjacent graph. Dans *International Conference on Graphic and Image Processing (ICGIP 2012)* (Vol. 8768, pp. 87681L). International Society for Optics and Photonics.
- Shi, Y., Zhang, Y., Xia, K., & Harik, R. (2020). A Critical Review of Feature Recognition Techniques. *Computer-Aided Design and Applications*, 17(5), 861-899. doi: <http://doi.org/10.14733/cadaps.2020.861-899>
- Siemens PLM Software. (2017a). NX for Design. Repéré à www.plm.automation.siemens.com/global/en/products/nx/nx-for-design.html
- Siemens PLM Software. (2017b). Solid Edge. Repéré à <https://solidedge.siemens.com/en/>
- Smit, M. S., & Bronsvort, W. F. (2007). The difference between two feature models. *Computer-Aided Design and Applications*, 4(6), 843-851. doi: <https://doi.org/10.1080/16864360.2007.10738516>
- Šormaz, D. N., & Tennety, C. (2010). Recognition of interacting volumetric features using 2D hints. *Assembly Automation*. doi: <https://doi.org/10.1108/01445151011029763>
- SpaceClaim Corporation. (2017). Ansys SpaceClaim. Repéré à <https://www.ansys.com/products/3d-design/ansys-spaceclaim>
- Sun, Y., Huang, Y., Chen, L., Wang, Q., & Wan, S. (2012). A New Multi-Level Attributed Graph Based Shape Matching Approach to Complex Template Design Feature Recognition. Dans *ASME 2012 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* (Vol. Volume 2: 32nd Computers and Information in Engineering Conference, Parts A and B, pp. 137-146). doi: <https://doi.org/10.1115/DETC2012-70804>
- Sunil, V., Agarwal, R., & Pande, S. (2010). An approach to recognize interacting features from B-Rep CAD models of prismatic machined parts using a hybrid (graph and rule based) technique. *Computers in Industry*, 61(7), 686-701. doi: <https://doi.org/10.1016/j.compind.2010.03.011>
- Sunil, V., & Pande, S. (2008). Automatic recognition of features from freeform surface CAD models. *Computer-Aided Design*, 40(4), 502-517. doi: <https://doi.org/10.1016/j.cad.2008.01.006>
- Sunil, V. B., Agarwal, R., & Pande, S. S. (2010). An approach to recognize interacting features from B-Rep CAD models of prismatic machined parts using a hybrid (graph and rule based) technique. *Computers in Industry*, 61(7), 686-701. doi:

<https://doi.org/10.1016/j.compind.2010.03.011>.

Repéré

à

<https://www.sciencedirect.com/science/article/pii/S016636151000028X>

- Tao, S. (2014). General and partial retrieval of CAD models based on surface region partition. *Computer-Aided Design and Applications*, 11(1), 32-42.
- Tao, S. (2015). CAD model retrieval based on graduated assignment algorithm. *3D Research*, 6(2), 1-11.
- Tao, S. (2018). 3D CAD model retrieval based on the softassign quadratic assignment algorithm. *Multimedia Tools and Applications*, 77(13), 16249-16265.
- Tao, S., Huang, Z., Ma, L., Guo, S., Wang, S., & Xie, Y. (2013). Partial retrieval of CAD models based on local surface region decomposition. *Computer-Aided Design*, 45(11), 1239-1252.
- Tao, S., Wang, S., & Chen, A. (2017). 3D CAD solid model retrieval based on region segmentation. *Multimedia Tools and Applications*, 76(1), 103-121.
- Tao, S. Q., & He, W. (2012). Similarity Assessment for Assembly Model Based on Component Attributed Relational Graph Matching. Dans *Applied Mechanics and Materials* (Vol. 215, pp. 270-274). Trans Tech Publ.
- Tao, S. Q., & Huang, Z. (2012). Assembly model retrieval based on optimal matching. Dans *Software engineering and knowledge engineering: Theory and practice* (pp. 327-336). Springer.
- Tarbox, G. H., Gottschlich, S. N., & Gerhardt, L. A. (1993). Registration of dissimilar featureless models for inspection. Dans *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on* (pp. 686-691). IEEE. doi: <http://doi.org/10.1109/ROBOT.1993.292058>
- Terzi, S., Bouras, A., Dutta, D., Garetti, M., & Kiritsis, D. (2010). Product lifecycle management—from its history to its new role. *International Journal of Product Lifecycle Management*, 4(4), 360-389.
- V5-R2016, C. (2016). Repéré à <https://www.3ds.com/products-services/catia/products/v5/portfolio/>
- Vandenbrande, J. H., & Requicha, A. A. (1993). Spatial reasoning for the automatic recognition of machinable features in solid models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(12), 1269-1285.

- Venu, B., Komma, V. R., & Srivastava, D. (2018). STEP-based feature recognition system for B-spline surface features. *International Journal of Automation and Computing*, 15(4), 500-512. doi: <https://doi.org/10.1007/s11633-018-1116-0>
- Verma, A. K., & Rajotia, S. (2010). A review of machining feature recognition methodologies. *International Journal of Computer Integrated Manufacturing*, 23(4), 353-368. doi: <https://doi.org/10.1080/09511921003642121>
- Viswanathan, K., Chowdhury, S., & Siddique, Z. (2008). Shape comparison of 3d models based on features and parameters. Dans *ASME 2008 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* (pp. 647-657). American Society of Mechanical Engineers. doi: <https://doi.org/10.1115/DETC2008-49948>
- Vranic, D. V., & Saupe, D. (2004). *3D model retrieval* (Ph.D. Thesis, University of Leipzig, Leipzig, Germany).
- Vranic, D. V., Saupe, D., & Richter, J. (2001). Tools for 3D-object retrieval: Karhunen-Loeve transform and spherical harmonics. Dans *2001 IEEE Fourth Workshop on Multimedia Signal Processing* (pp. 293-298). IEEE. doi: <http://doi.org/10.1109/MMSP.2001.962749>
- Wang, E., & Kim, Y. S. (1998). Form feature recognition using convex decomposition: results presented at the 1997 ASME CIE Feature Panel Session. *Computer-Aided Design*, 30(13), 983-989. doi: [https://doi.org/10.1016/S0010-4485\(98\)00058-X](https://doi.org/10.1016/S0010-4485(98)00058-X). Repéré à <https://www.sciencedirect.com/science/article/pii/S001044859800058X>
- Wang, J., Jiang, B., & He, Y. (2010). Shape-based search of mechanical CAD models for product data management. *International Journal of Computer Applications in Technology*, 37(2), 125-131.
- Wang, J., & Zhou, L. (2018). Algorithm for automatic boss feature recognition and ejector sleeve design. *The International Journal of Advanced Manufacturing Technology*, 97(1-4), 583-597. doi: <https://doi.org/10.1007/s00170-018-1918-9>
- Wang, Q., & Yu, X. (2014). Ontology based automatic feature recognition framework. *Computers in Industry*, 65(7), 1041-1052. doi: <https://doi.org/10.1016/j.compind.2014.04.004>
- Wang, X., Wang, J., & Pan, W. (2014). A sketch-based query interface for 3D CAD model retrieval. Dans *The 2014 2nd International Conference on Systems and Informatics (ICSAI 2014)* (pp. 881-885). IEEE.

- Watkins, M. A. (1988). Problems in geometric continuity. *Computer-Aided Design*, 20(8), 499-502. doi: [https://doi.org/10.1016/0010-4485\(88\)90012-7](https://doi.org/10.1016/0010-4485(88)90012-7). Repéré à <http://www.sciencedirect.com/science/article/pii/0010448588900127>
- Wei, L., & Yuanjun, H. (2006). Retrieving 3D CAD model with regional entropy distributions. Dans *International Conference on Artificial Reality and Telexistence* (pp. 871-879). Springer.
- Woo, Y. (2003). Fast cell-based decomposition and applications to solid modeling. *Computer-Aided Design*, 35(11), 969-977. doi: [https://doi.org/10.1016/S0010-4485\(02\)00144-6](https://doi.org/10.1016/S0010-4485(02)00144-6). Repéré à <https://www.sciencedirect.com/science/article/pii/S0010448502001446>
- Xiaoliang, B., Shusheng, Z., & Kaixing, Z. (2010). A method of 3D CAD model retrieval based on genetic algorithm. Dans *2010 International Conference on Electronics and Information Engineering* (Vol. 1, pp. V1-562-V561-565). IEEE.
- Xu, L., & Jiang, Y. (2018). Research on 3D CAD model retrieval based on ant colony algorithm. Dans *2018 8th International Conference on Management, Education and Information (MEICI 2018)* (pp. 636-639). Atlantis Press.
- Yang, Y., Lin, H., & Zhang, Y. (2007). Content-based 3-D model retrieval: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(6), 1081-1098. doi: <http://doi.org/10.1109/TSMCC.2007.905756>
- Yin, Y., & Guo, L. (2018). An Algorithm for Similar 3D Model Difference Examination Using Geometric Matching. Dans *ASME International Mechanical Engineering Congress and Exposition* (Vol. 52187, pp. V013T005A003). American Society of Mechanical Engineers.
- Zehtaban, L., Elazhary, O., & Roller, D. (2016). A framework for similarity recognition of CAD models. *Journal of Computational Design and Engineering*, 3(3), 274-285.
- Zhang, C., & Zhou, G. (2019). A view-based 3D CAD model reuse framework enabling product lifecycle reuse. *Advances in Engineering Software*, 127, 82-89.
- Zhang, C., Zhou, G., Yang, H., Xiao, Z., & Yang, X. (2019). View-based 3-d cad model retrieval with deep residual networks. *IEEE Transactions on Industrial Informatics*, 16(4), 2335-2345.
- Zhang, C., Zhou, X.-h., & Li, C.-x. (2009). Automatic recognition of intersecting features of freeform sheet metal parts. *Journal of Zhejiang University-Science A*, 10(10), 1439-1449. doi: <https://doi.org/10.1631/jzus.A0820705>

- Zhang, J., & Li, Y. (2016). Region segmentation and shape characterisation for tessellated CAD models. *International Journal of Computer Integrated Manufacturing*, 29(8), 907-915. doi: <https://doi.org/10.1080/0951192X.2015.1130249>
- Zhang, J., Zuo, M., Wang, P., Yu, J.-f., & Li, Y. (2016). A method for common design structure discovery in assembly models using information from multiple sources. *Assembly Automation*.
- Zhao, X., Liu, X., & Zhang, K. (2017). A structure-based 3d CAD model similarity assessment approach. Dans *Proceedings of the 6th International Conference on Software and Computer Applications* (pp. 281-284).
- Zhu, K., San Wong, Y., Loh, H. T., & Lu, W. F. (2012). 3D CAD model retrieval with perturbed Laplacian spectra. *Computers in Industry*, 63(1), 1-11.
- Zhuang, T., Zhang, X., Hou, Z., Zuo, W., & Liu, Y. (2017). A Novel 3D CAD model retrieval method based on vertices classification and weights combination optimization. *Mathematical Problems in Engineering*, 2017.
- Zubair, A. F., & Mansor, M. S. A. (2018). Automatic feature recognition of regular features for symmetrical and non-symmetrical cylinder part using volume decomposition method. *Engineering with Computers*, 34(4), 843-863. doi: <https://doi.org/10.1007/s00366-018-0576-8>