



Design Enabler to Recognize Duplicate Geometries in CAD Assemblies

Aravind Shanthakumar¹ and Joshua D. Summers²

¹PartMaker Inc., aravind@partmaker.com

²Clemson University, joshua.summers@ces.clemson.edu

ABSTRACT

This paper presents a method for identifying components in CAD assemblies that have surfaces that have complementary, duplicate surfaces. The method evaluates faces on two parts within a given proximity by measuring the approximate surface that has parallel and opposing surface normal between the faces. This method can be used for applications such as identifying potential lazy parts, a previously developed method used by an automotive OEM, and generating connectivity graphs for use in manufacturing assembly time estimation. The method considers threshold distance between parts, orientation angle between faces, and targeted similarity overlap between geometries. This paper presents the algorithm, a justification, and example test cases and scenarios that demonstrate its utility.

Keywords: feature recognition, duplicate geometry, similarity, lazy parts, assembly models.

DOI: 10.3722/cadaps.2013.889-904

1 MOTIVATION: LAZY PARTS IDENTIFICATION METHOD AND ASSEMBLY TIME ESTIMATION

The method presented in this paper to determine similar, not identical, surfaces between closely spaced parts in an assembly model is motivated by two separate research efforts: (1) Lightweight Engineering and (2) Design for Assembly Time Estimation. The first derives from a new attention focusing design method for automotive OEM teams to identify potential mass savings areas in the vehicle. This effort on lightweight engineering has been developed and validated on complete vehicle and sub-systems [2,7,15] and is currently employed as a set of design guidelines at the sponsoring OEM [17]. The method provides a list of seven identifiers called laziness indicators to select components for mass reduction analysis. One indicator that could be automated and integrated into modeling software is the “duplicate geometry” indicator. Duplicate geometry is generally defined as two closely located geometries that are similar in shape to each other (Fig. 1). Thus, a feature recognition system is needed that can quickly determine which parts have potentially duplicate geometries, highlighting the associated faces so that engineers can examine whether these might be integrated in some manner to reduce mass in the assembly.

A second design method that has been recently developed in which duplicate geometry recognition in assembly models is needed is in the assembly time estimation method based on structural complexity [13]. In this method, assembly graphs are created based on connections between parts, these graphs are converted into a vector of graph-based complexity metrics [12] and then used

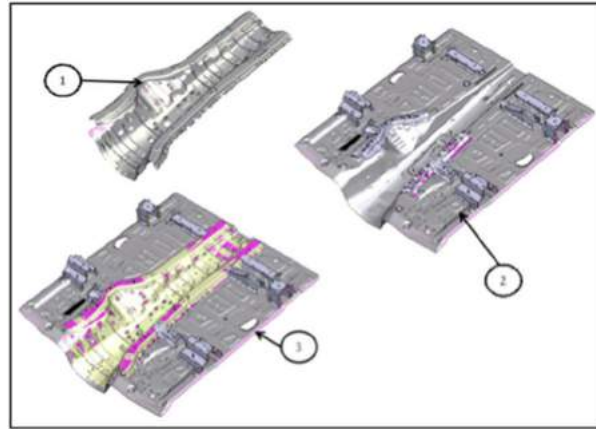


Fig. 1: Heat shield and vehicle undercarriage.

to train artificial neural networks to predict the assembly times from the graphs [14]. It has been shown that this method is useful for automotive sub-systems and consumer products [19]; exploiting both detailed models and abstracted, low-fidelity models [16]. The method is dependent on the generation of the connective graphs from the assembly models. To generate these graphs, an assembly-mate based approach was implemented with SolidWorks and demonstrated to be generally independent of designer modeling choices, but is dependent on the assembly model being mated in some manner [18]. This method is not acceptable in some automotive OEMs in which they do not want to require their engineers to use mate based assembly constraints. Therefore, a method to generate these connectivity graphs based on part locations and geometry is needed.

A feature recognition algorithm to support the automation of duplicate geometry recognition should be developed with user-controllable parameters to allow the designers flexibility in defining “duplicate” for their specific needs for the two motivating methods. The idea is to have single feature recognition system with user driven parameters that can provide the required extensibility. Additionally, it is also desired to have the feature recognition system that is independent of the geometry type. These two design methods serve as the motivation for this research and the newly developed design enabler, duplicate geometry feature recognition algorithm.

2 BACKGROUND: FEATURE RECOGNITION REVIEW

The current state of the art in the feature recognition technology focuses mainly on the integration of CAD and CAM, CNC visualization, process planning, and manufacturing [1,11,22,27]. Reviewing different feature recognition techniques, it is seen that the common challenges faced across all approaches are the recognition of interacting features, dealing with free-form surfaces, and having a general purpose algorithm for all feature types. The solid models’ topological entity relationships with certain geometric attributes are the preferred representation used in the graph-based, hint-based, and hybrid feature recognition approaches. Different kinds of representation used for the feature recognition purposes include the labeled graph, directed graph, bipartite graph, and undirected graph. The feature representation in convex hull decomposition and cell based decomposition techniques are volume based, and hence volumes of primitive shapes are used for feature representation. The comparison of previously discussed feature recognition techniques are shown in Tab. 1.

3 DEFINITION OF DUPLICATE GEOMETRY

There is no standard definition for features and the current definitions found in the literature depend on the downstream application where the model will be used [10]. Features can hold different meanings based on use context. The definition of features vary depending on whether the FR

FR Technique	Feature Representation	Reasoning	Geometry	Independent of feature type?	Complexity
Graph-based	Topology, Geometry	Graph matching [5,9]; Heuristic [28]; neural nets [4]; logic rules [10]	Planar, Cylindrical	No; includes pre-defined library of feature types	Exponential [6,8]
Hint-based	Topology, Geometry, Heuristics, Ray firing [25]	Graph matching, Rules	Planar, Cylindrical, Second-order curves	No; includes pre-defined library of feature types	Polynomial [8,20]
Convex hull	Delta volume of primitive shapes	Rules, Graph matching	Polyhedral, Cylindrical [26]	Independent of feature type	Exponential
Cell decomposition	Maximal volumes	Logic Rules, Heuristic [21], Graph matching [21]	Polyhedral	Independent of feature type	Exponential [8]
Hybrid	Topology, Geometry, Heuristics	Graph matching [5], Rules	Planar, Cylindrical, Second-order curves	No; includes library of feature hints	Polynomial [15]

Tab. 1: Comparison of feature recognition techniques.

algorithm is intended for identifying machining features, extruded features, polyhedral entities, or features for stress analysis. For example, extruded entities in the part are classified as a feature for the finite element modeling application for mesh generation. However, for machining purposes concave features are classified as features to calculate the tool path and the amount of material that needs to be removed to produce that feature. Also, presently there is no standard definition for features and it is argued that it may not be possible to have a single definition covering all feature types [8,22].

For the research in this paper, a feature recognition algorithm is needed to support the duplicate geometry identification and extraction of assembly relations from CAD assembly. An example for duplicate geometry is the vehicle underbody and cable guide as shown in Fig. 2 where the profile of the cable guide follows the profile of the vehicle underbody and both geometries lie close to each other.

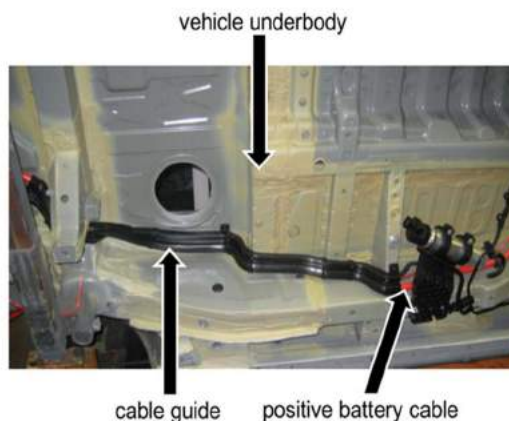


Fig. 2: Cable guide attached to the underside of the battery.

However, as discussed earlier this definition is not comprehensive and therefore identification depends on engineering judgment. To explain this further, some of the questions that need to be answered objectively for the identification of duplicate geometry are:

- Do the two geometries lie close to each other?
- What distance between the geometries can be regarded as close?

- Are the two geometries similar?
- If similar, what is the amount of similarity required?

In order to remove the ambiguity involved with identifying duplicate geometry from the current definition and also to make the definition objective for the purposes of automation, the following definition is proposed:

Geometries lying equal to or within a threshold distance (user defined) with the surface outward normals opposed to each other within a threshold tolerance (user defined) and the percentage of similarity between the two geometries is equal to or within a threshold value (user defined).

In this definition, there are three user defined variables that determine if the geometries are duplicate. The ambiguity involved in the earlier definition is removed by the use of these user defined variables that are quantitative in nature. Tab. 2 provides a comparison of subjective questions in the earlier definitions to the user defined variables in the new definition. There is also a threshold tolerance for the surface outward normal that is not shown in Tab. 2. This parameter is used to ensure that the profiles of two geometries are opposed to each other, which is discussed with an example in the next section.

The revised definition offers three conditions that need to be satisfied for the geometries to be evaluated as duplicate. The three conditions are the threshold distance condition, the orientation condition, and the percentage similarity condition. It should be noted that this effort is not a replication of similar part searches that have been implemented in software tools such as Cadenas or Siemen's Geolus. These tools look at the complete shape, rather than sub-regions of the parts. The next section presents the discussion on the three duplicate geometry conditions.

Questions in original definition	Addressed in revised definition
Do the two geometries lie close to each other? What distance between the geometries can be regarded as close?	Threshold distance
Are the two geometries similar? If similar, what is the amount of similarity required?	Percentage value of similarity

Tab. 2: Subjectivity in the old definition addressed in revised definition.

3.1 Threshold Distance

Threshold distance is the first condition in the definition of duplicate geometry. As per the definition, only those geometries that are lying within or equal to the threshold distance should be considered for duplicate geometry analysis. This condition is derived from the original definition of duplicate geometry from lazy parts mass reduction method that requires geometries to be in close proximity. By defining a threshold distance, the ambiguity involved with what distance can be considered close is removed. The example in Fig. 3 shows two instances of same curve pairs but with different distances between them. In Fig. 3 (a), the curves are considered for duplicate geometry analysis as the distance between them is equal to the threshold distance. However, in the Fig. 3 (b) the same two curves cannot be considered for duplicate geometry analysis as the distance between them is greater than the threshold distance.

3.2 Orientation Angle and Tolerance

The orientation angle and tolerance is a user-defined input value for the algorithm that determines the angle between the duplicate geometries. From the definition of the duplicate geometries, it is required for the duplicate geometries to satisfy the angle condition. Typically in the assemblies the angle between the geometries is not always a single value, especially in the case of freeform and cylindrical surfaces. Moreover, the intent of identifying duplicate geometry is more of satisficing problem than an optimization problem [24]. Because of this reason a tolerance is used to compensate the variation of the angle along the surface. For the example shown in Fig. 4, the angle between the two opposing topologically correct normal need to be within $180^\circ \pm a$ tolerance band. Here the angle (-) needs to be within the tolerance. If the angle is equal to the angle , then the angle would be 180° . Therefore the

difference between angle α and β should be less than the orientation tolerance if the two geometries need to be considered for the duplicate geometry analysis.

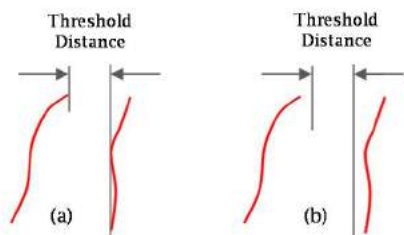


Fig. 3: Geometries that are lying within or equal to threshold distance are considered for duplicate geometry analysis.

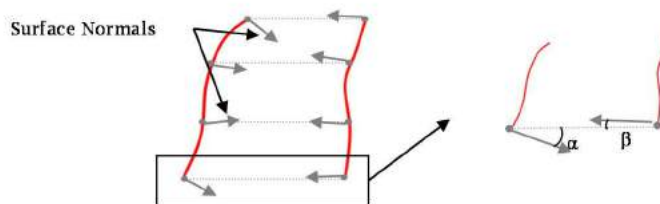


Fig. 4: The angle between the outward normals from opposing geometries need to be within the threshold tolerance.

3.3 Percentage Similarity

Percentage similarity is the third, and final, condition in the revised definition of duplicate geometry. The percentage similarity is a user defined parameter used to address the ambiguity involved with the amount of similarity in the original definition. The original definition stated that the two geometries need to be similar in order to be considered duplicate, but did not mention the amount of similarity that was required. The revised definition provides control to the user to determine how much of similarity is required for the intended application. The similarity between the two geometries, upon satisfying the first two conditions, is calculated by measuring distance between sampling points on the two surfaces. The distances d_1 , d_2 , d_3 , and d_4 in the Fig. 5 show the distance measurements between the corresponding sampling points. The two geometries are considered duplicate if the number of measurements between the sampling points from the two geometries meets the user defined percentage similarity value. In this example, if d_1 , d_2 , and d_3 were all equal to each other and the percentage of similarity defined was 75% or above, then the two geometries are duplicate. However, in the actual assembly it may not always be feasible to have all measurements equal to one another based on the number of sampling points used. For this reason, bounds are considered instead of a single value. That is if d_1 , d_2 , and d_3 are all equal to each other within a certain tolerance then the two geometries are duplicate. The bounds can be adjusted by the user based on the intended application.

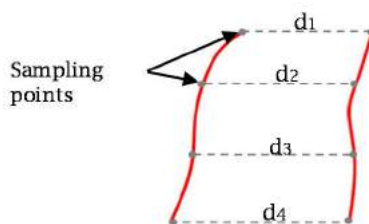


Fig. 5: The distance measurements between the sampling points.

3.4 Recognition Requirements

Based on the definition, a set of duplicate geometry recognition tool requirements are elicited (Tab. 3). These requirements are defined for both the functionality of the algorithm, but also based on constraints within the development environment, such as access to commercial CAD API developer tools. A complete discussion on the requirements and how they are satisfied is found in [23].

	Requirement	Explanation
1	<i>The system allows users to define the threshold distance between duplicate geometries.</i>	The user gets to decide the proximity between geometries based on the application and experience for the duplicate geometry analysis. The proximity between duplicate geometries could be different for different mechanical systems
2	<i>The system allows users to define the tolerance for surface outward normal</i>	The user can set the orientation that is required between the geometries with certain tolerance value for the duplicate geometry analysis. Depending on the geometric type of parts in the assembly, the user can choose the angle that is appropriate for the given assembly
3	<i>The system allows users to define the percentage of similarity</i>	The degree of similarity that is desired between geometries is decided by the user. This parameter indicates the extent of similarity between the geometries being compared. For example, the percentage of similarity value of 100% would mean identical geometries and the value zero would mean completely dissimilar
4	<i>The system allows users to adjust the bounds for the distance measurements between the sampling points</i>	The similarity between geometries is calculated by measuring the distance between sampling points on the two geometries. This list of distances between sampling points are analyzed to check if they are equal to each other within a certain tolerance. The tolerance value can be varied depending on the application and is decided by the user.
5	<i>The system needs to highlight instances of duplicate geometry</i>	The system need to display the result of the duplicate geometry analysis to the user, so that the user is able to visualize the instances of duplicate geometries in the assembly. The highlighted geometries in the assembly would inform the user about regions where lazy parts mass reduction method could be applied.
6	<i>The system needs to work with different geometric types</i>	The parts in the assembly may be composed of different geometric types. The examples of some of the geometric types are planar, cylindrical, spherical, conical, freeform, and toroidal. The algorithm needs to function with such geometric types.
7	<i>The system offers extensibility to extract assembly relations with weight</i>	The algorithm need to extract the weighted bipartite graph of assembly relations to support the part connectivity based assembly time estimation method
8	<i>The system supports the assembly models from SolidWorks (licensed CAD software in the university) for design analysis</i>	The SolidWorks is the licensed CAD software in Clemson University that provides easy-to-use GUI for creating parts and assemblies.
9	<i>The users are able to access the duplicate geometry program from within the SolidWorks software:</i>	Presently, the duplicate geometries and part connectivity information are manually evaluated by loading the CAD assemblies in SolidWorks. For the automation of duplicate geometry identification and extraction of assembly relations, the system need to provide access to duplicate geometry program upon opening the CAD assembly file.
10	<i>The users are able to start the duplicate geometry analysis by the click of a mouse button:</i>	The system need to reduce the time required for the user to start the duplicate geometry analysis

Tab. 3: Recognition system requirements.

4 DESIGN AND IMPLEMENTATION

In this section, the software architecture and the implementation details of the duplicate geometry feature recognition algorithm are discussed.

4.1 Design

The software system consists of three components: (1) a CAD modeling tool to support the solid model data, (2) a duplicate geometry feature recognition algorithm for reasoning, and (3) a GUI for the user input parameter definition and output (highlighting duplicate geometries and generating a

connectivity graph). The SolidWorks 2010 software is used as a CAD tool and is integrated with the feature recognition algorithm. SolidWorks CAD software offers an API (Application Programming Interface) for the exchange of geometric and topological data between the SolidWorks CAD software and an external application. Visual studio C++ programming language was used for the implementation of the feature recognition algorithm.

The algorithm extracts the necessary geometric and topological information pertaining to the CAD assembly file through the API function calls. After extracting the required CAD data, the FR algorithm performs the duplicate geometry analysis. The results of the analysis are communicated back to the SolidWorks CAD software using the API function calls. This communication of data between SolidWorks and the feature recognition algorithm is represented as a loop in the Fig. 6. An add-in is created in the SolidWorks GUI for the user to start the duplicate geometry analysis as shown in the Fig. 7. Before starting the analysis, the user is required to import the CAD assembly into SolidWorks. The results of the analysis are displayed in SolidWorks by highlighting the instances of duplicate geometry.

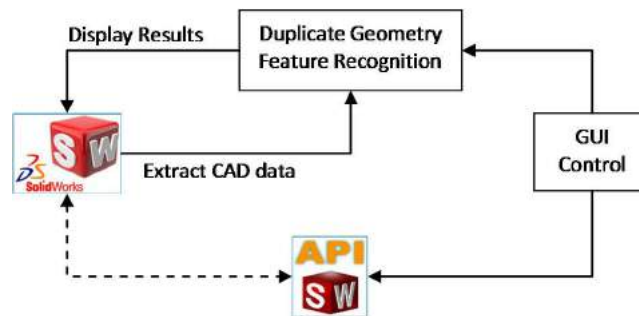


Fig. 6: Simplified system architecture.



Fig. 7: SolidWorks GUI with "Find Duplicate Geometries" add-in.

4.2 Implementation

The flowchart (Fig. 8) illustrates the general approach of the feature recognition algorithm for the identification of duplicate geometry. The algorithm is categorized into three stages for the ease of discussion. The first stage involves reading the assembly file and filtering parts satisfying the threshold distance condition. The second stage checks for the orientation condition and the third stage deals with the analysis for percentage similarity between faces.

4.3 Stage One: Filter Parts within the Threshold-distance

The feature recognition algorithm reads the assembly file loaded in the current session of SolidWorks. The program extracts the total number of visible parts from the active assembly document. If the assembly consists of sub-assemblies, the parts in the sub-assemblies are considered towards the total part count.

The next step in this stage is to filter pairs of parts satisfying the threshold distance condition. The axis aligned bounding box representation of the part is used for this purpose due to its simple

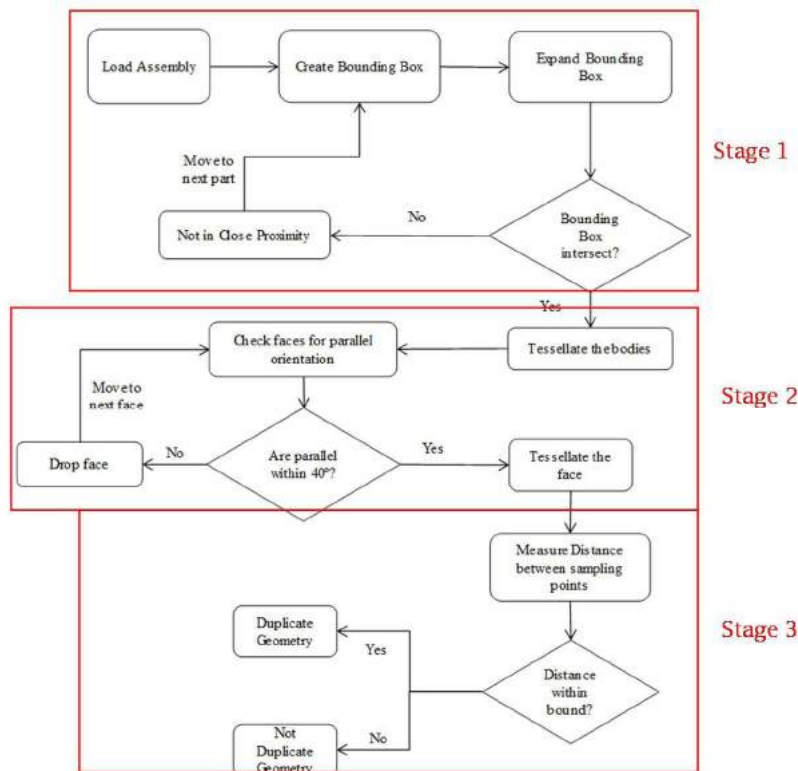


Fig. 8: Three stages of duplicate geometry recognition.

geometric representation [3]. The axis aligned bounding box for a sample 2d shape is shown in Fig. 9. Since, the bounding box represents the outer enclosure of the part, absence of intersection between bounding boxes ensures absence of intersection between the parts. This rule is used in this research to determine proximity between parts.

The bounding box is obtained in the form of x, y, and z coordinates for the upper and lower diagonal points of the bounding box. After retrieving bounding box for all visible parts in the assembly, the bounding box is expanded. The bounding box size is expanded in the three main Cartesian coordinate directions by the amount equal to half the user-defined threshold distance value. To explain this further, if the user-defined threshold distance value is 'x' unit then the bounding box around all parts are expanded by the amount equal to 'x/2' unit in the three main Cartesian coordinate directions.

As shown in the Fig. 10, expanded bounding box are used to check for intersection between parts. Intersection between the expanded bounding boxes ensures that the distance between the original bounding boxes meets the threshold distance condition.

The bounding box method is an approximate and quick check to filter only those parts that satisfy the threshold distance value. Therefore, the actual minimum distance between the two parts may be greater than the threshold distance value. Although bounding box method includes false positives, false negatives are eliminated. The false positives are eliminated in the future stages of this algorithm.

If the assembly contains 'n' parts, then each part is checked for interference with (n-1) parts. The Big O complexity for this algorithm is $O(N^2)$, where N is the number of parts as there is a for-loop nested within another for-loop.

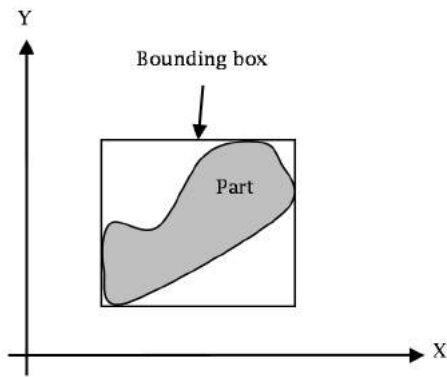


Fig. 9: Example of an axis-aligned bounding box.

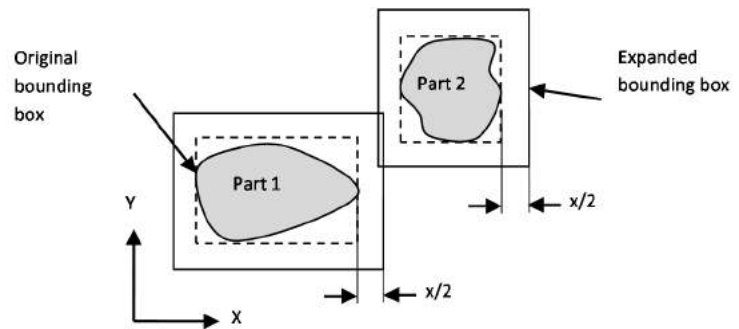


Fig. 10: Bounding box expanded to check for threshold distance conditions.

Once the intersection between the two bounding boxes is determined, the parts associated with the two bounding boxes are stored as pairs in a container (Tab. 4). This forms the end of stage One, where part pairs satisfying the threshold distance condition are grouped. The next stage presents the discussion on the check for orientation condition.

Array	Components Names
[0]	("P1", "P2")
[1]	("P1", "P3")
[2]	("P2", "P3")

Tab. 4: Parts stored as array of pairs.

4.4 Stage Two: Check for Orientation between Faces

The part pairs that have satisfied threshold distance condition are analyzed in this stage for the presence of faces whose surface outward normals are opposed to each other within a threshold tolerance (from the definition of duplicate geometry). The first step in this stage is to access each pair of parts from the array (Tab. 4) for the extraction of bodies and faces. The bodies and faces are the topological entities of the solid models' B-Rep data structure of the two parts.

As shown in Fig. 11, part P1 and P2 represent a part pair from the array. B1 and B2 represent the bodies extracted from the parts P1 and P2 respectively. Faces f11 through f15 represent a total of five faces extracted from the body B1. Similarly, faces f21 through f2n represent a total of 'n' faces extracted from the body B2. Once the faces from both parts are extracted, the program tessellates the faces to generate sampling points.

Tessellation is the process of representing the face in terms of triangles much like the representation seen in STL format. The FR algorithm performs coarse tessellation on all the faces of the part pairs by specifying no limits on the maximum edge length for the triangles. The sampling points generated from tessellation are used for the checking orientation between faces. The program compares the orientation of each face of part P1 with all the faces from part P2. The complexity of this algorithm is less than or equal to $O(N^2)$.

The program retrieves the unit surface outward normals at every sampling point for the two faces of interest. The surface outward normals for the two faces are stored in two separate lists. The first unit normal vector is selected from list one and the angle between this and all the unit normal vectors from the list two is calculated. If 'n1' is the unit normal vector from list one and 'n2' is the unit normal vector from list two, then the angle between them is calculated using the dot product.

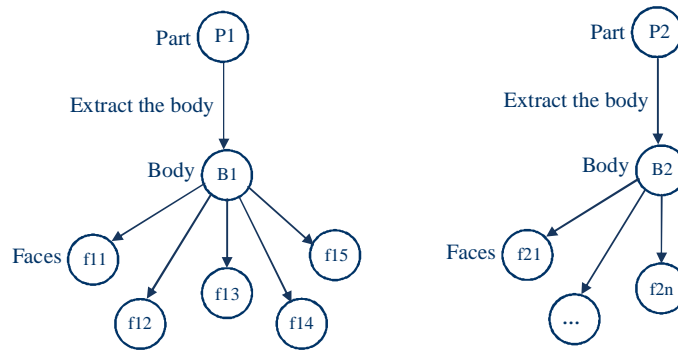


Fig. 11: Faces extracted from the bodies in the part pair.

If the angle between the two vectors is within the user-defined tolerance then a counter is incremented by one unit. Next, the iteration is repeated with the second unit normal from list one and all the unit normals from list two until all the unit normals in the list one are exhausted. If at least three unit normals from list one forms the angle with any unit normals from list two that is within the tolerance, then the two faces are considered to be meeting the orientation condition. The three unit normals satisfying the orientation condition ensures that there may lie atleast one facet (or triangle) on face one that satisfies the orientation condition with face two. The face pairs from the two parts that satisfy the orientation condition are stored in an array (Tab. 5). The face pairs in this list are analyzed for the percentage similarity in stage three.

Pair of faces that have parallel orientation
f11 - f23
f15 - f24
f16 - f25

Tab. 5: Example list showing faces stored as pairs that have orientation within the user-defined angle and tolerance.

4.5 Stage Three: Analyze Percentage Similarity between Faces

The face pairs are re-tessellated in stage three to produce finer mesh by limiting the maximum edge length of the triangle. Presently, the maximum edge length of the triangle is limited to the length of the shortest edge in the assembly. The sampling points thus generated for the two faces are stored in two separate lists. Starting with the shorter (or any list if the size is equal) of the two lists, the distances are measured from each sampling point from the list one to all the sampling points in list two (Fig. 12). The distance formula is used to measure the distance between sampling points.

After completing the calculation of distances from first sampling point from list one to all the sampling points in list two, the shortest distance is saved and the procedure is repeated now starting with sampling point two in list one. After exhausting all sampling points in list one, we get a list of shortest distances between sampling points from list one and sampling points from list two. The average is computed for the list of shortest distances. The user-defined upper and lower tolerance is added to the newly computed average.

To determine the percentage similarity between the two faces, the ratio of the number of shortest distances that fall within the tolerance bounds to the size of the shortest distance list is calculated. The two faces are evaluated to be duplicate, if the ratio is equal to or greater than the user-defined percentage similarity value. The two faces are evaluated to be duplicate if the user-defined percentage similarity value is lesser than or equal to the ratio (0.66). The number signifies that there is 66%

similarity between the two faces. The faces are highlighted by the algorithm if they are evaluated to be duplicate geometries.

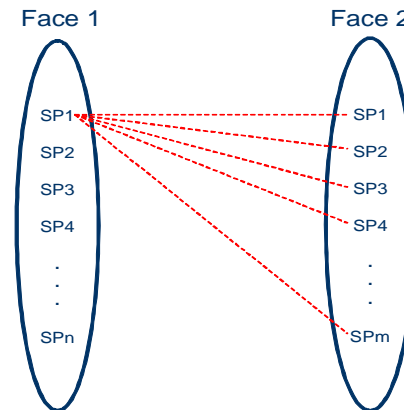


Fig. 12: Measurement of distance between sampling points.

5 DEMONSTRATION OF SYSTEM REQUIREMENT ADDRESSMENT

The functioning of the algorithm is evaluated against the system requirements (Tab. 3) using test cases [23]. Different test cases are designed to study the performance of the algorithm for each of the first seven requirements. The first seven requirements refer to the system requirements necessary for duplicate geometry analysis and displaying the results culminating in the requirement that the system highlight duplicate geometry of assembly models. A demonstration of this functionality is presented in the next section. Requirements 8-10 are relevant to usability and relate to the implementation. The current system supports assembly models from SolidWorks CAD software for the analysis (requirement no. 8). Additionally, the users are able to access the duplicate geometry program from inside SolidWorks by using the duplicate geometry tool built in SolidWorks (requirement no. 9 and 10). The requirement for supporting bi-partite generation of part connectivity is demonstrated in the following section.

5.1 Requirement Met: Duplicate Geometry Highlighting

The fifth requirement of the algorithm states that it is required to highlight the instances of duplicate geometries for the user to visualize the results on screen. This requirement is met by changing the color of duplicate geometries to red. To illustrate further, for a given CAD assembly all the instances of duplicate geometry pair are highlighted by changing the color of the face to red (Fig. 13). Due to this reason, it required not to have any parts in the assembly whose color is already set to red.

5.2 Demonstration: Connectivity Graph Generation

The ability of the algorithm to use the duplicate geometry approach to extract the part connectivity graph for CAD assemblies is presented in this section. For brevity, only one demonstration example is provided, that of a caster assembly; others may be found in [23]. The test case used is the caster assembly from SolidWorks library. Bias is mitigated by selecting an externally developed assembly model of a typical real world system. The caster is an assembly of the wheel and supporting parts that is attached to the bottom of mechanical structure for the purpose of moving. The caster assembly consists of seven parts (Fig. 14).

The input parameters for the analysis of caster assembly are shown in Tab. 6. The parameters are similar to others used in different demonstration cases [23], with the sole difference the face edge size was doubled to 10mm.

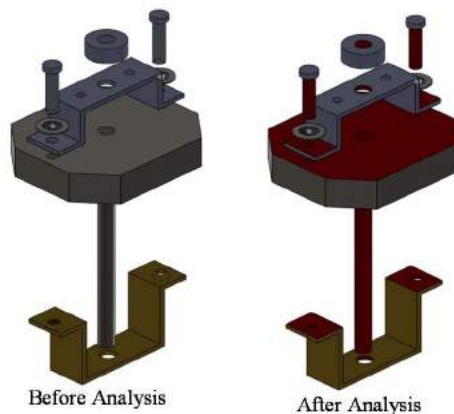


Fig. 13: Instances of duplicate geometry highlighted by the algorithm.

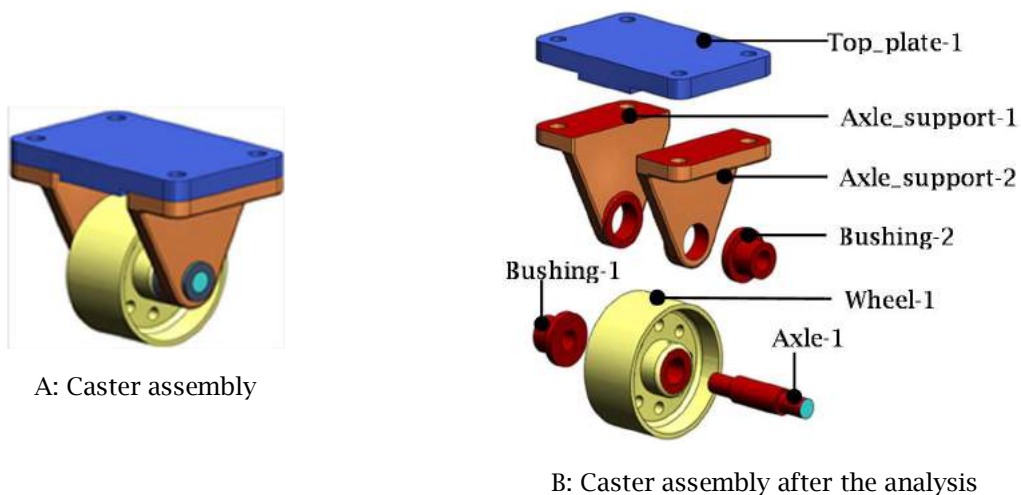


Fig. 14: The caster assembly from SolidWorks library.

Name	caster.sldasm
Max Facet Size	10 mm
Threshold Distance	1 mm
Orientation	$180^{\circ} \pm 2^{\circ}$
Percentage Similarity	Not applicable for assembly relations extraction
Bound	2 mm

Tab. 6: Input parameters for the caster assembly.

A total of eleven part connections are identified for the caster assembly. Through manual visual inspection of the assembly, a total of eleven part pairs are identified as containing duplicate geometry (Tab. 7). The feature recognition system should be able to recognize these part pairs, replicating the

manual approach that is defined within the lazy part identification method [15]. These pairs will be used to evaluate the ability of the feature recognition algorithm to identify duplicate parts.

Relation #	Part Name	Part Name
1	Top_plate-1	Axle_Support-1
2	Top_plate-1	Axle_Support-2
3	Axle_Support-1	Bushing-1
4	Axle_Support-1	Bushing-1
5	Axle_Support-2	Bushing-2
6	Axle_Support-2	Bushing-2
7	Bushing-1	Wheel-1
8	Bushing-2	Wheel-1
9	Axle-1	Wheel-1
10	Axle-1	Bushing-1
11	Axle-1	Bushing-2

Tab. 7: Anticipated part connections for caster assembly.

The algorithm was able to identify twenty five part connections in the assembly. This is fourteen part connections more than the anticipated part connections. The algorithm has identified other duplicate geometric pairs that have satisfied the 1mm threshold condition. The algorithm consumed 36.6 minutes to complete the analysis. The weighted bipartite graph of part connections for the caster assembly retrieved by the algorithm is shown in Tab. 8. The predicted relations are highlighted. Additional relations are discovered between the parts' other faces. All part pairs predicted in Tab. 7 are found in Tab. 8. This demonstrates that the algorithm extracts the predicted relations and can reliably automate the manual process.

The algorithm was tested using the caster assembly. The results indicate that the algorithm is able to identify suppressed and hidden parts and consider only the active parts for the analysis. The assembly relations are exported to a *.csv file with the part names and the corresponding overlapping weights.

6 CONCLUSION

The motivation for this research was to develop a feature recognition system that could automate the identification of duplicate geometries in CAD assemblies to support the lazy-parts lightweight engineering method. Also, a need was identified for the development of a feature recognition system for the automated extraction of assembly relations from CAD assembly file to support the part connectivity-based assembly time estimation method. Based on the identified needs, the objective of this research was to develop a feature recognition algorithm that could both identify duplicate geometries and retrieve assembly relations.

6.1 Research Contribution

The repeatability issue associated with the manual identification of duplicate geometry is addressed by this research. The original definition for duplicate geometry was subjective and therefore provided opportunity for subjectivity in the decision making. The formal definition of duplicate geometry proposed in this research removes the subjectivity in identifying duplicate geometries. In addition to addressing the issues of repeatability and subjectivity, the automated identification of duplicate geometry by the feature recognition algorithm removes the tediousness involved with the manual identification.

The part connectivity based assembly time estimation is a semi-automated method for the assembly time estimation that required manual construction of the assembly relationship for the input. The construction of the part connectivity graph manually was a tedious process that required time and effort both for the construction and quality check for errors. The algorithm developed in this research allows for the automated extraction of part connectivity graph from an assembly file that

reduces human effort required to study the assembly and prepare the graph. The algorithm eliminates the need for checking the graph for manual construction error and consistency. The automated retrieval of the assembly relations would allow designers more time on the data analysis by the reduction in time and effort required for data collection. The algorithm provides the way for complete automation of part connectivity-based assembly time estimation.

Relation # (Tab. 7)	Part Name	Part Name	Overlap Weight
9	Axle-1	Wheel-1	0.865269
	Axle-1	Wheel-1	0.357143
	Axle-1	Wheel-1	0.357143
	Axle-1	Bushing-2	0.469880
11	Axle-1	Bushing-2	1.000000
	Axle-1	Bushing-2	0.406780
10	Axle-1	Bushing-1	1.000000
	Axle-1	Bushing-1	0.406780
	Axle-1	Bushing-1	0.469880
8	Wheel-1	Bushing-2	0.0714268
7	Wheel-1	Bushing-1	0.0357143
5	Bushing-2	Axle_support-2	0.513274
	Bushing-2	Axle_support-2	0.0617284
	Bushing-2	Axle_support-2	0.269767
	Bushing-2	Axle_support-2	0.0617284
6	Bushing-2	Axle_support-2	0.974522
	Bushing-2	Axle_support-2	0.451327
	Bushing-2	Axle_support-2	0.383459
2	Axle_support-2	Top_plate-1	0.7343
1	Top_plate-1	Axle_support-1	0.7343
3	Axle_support-1	Bushing-1	0.898089
4	Axle_support-1	Bushing-1	0.452229
	Axle_support-1	Bushing-1	0.383459
	Axle_support-1	Bushing-1	0.0740741
	Axle_support-1	Bushing-1	0.0864198

Tab. 8: Part connections retrieved for the caster assembly.

The research in [16,18,19] focused on the development of a tool for the complete automation of the assembly time estimation for CAD assemblies using the user-defined mates information. However, the limitation of this research was the inability to extract connections in the case of part patterns. The duplicate geometry algorithm presented in this paper can extract connectivity information from the part patterns. The limitations of using user-defined mates for the assembly time prediction can be overcome by using the duplicate geometry algorithm that can extract the part connections which is objective.

The feature recognition algorithm developed in this research is independent of the geometric types. The test cases made of different geometric types demonstrate the ability of the algorithm to evaluate different kinds of geometries. It is observed that the analysis time was the only parameter affected by the different geometric types because of the change in the number of sampling points for orientation and percentage similarity calculation. The geometric type did not have any effect on the threshold distance calculation.

6.2 Future Work

The algorithm can retrieve weighted bipartite graph of part connections that is used as input for the part connectivity based assembly time estimation. The method is semi-automated except for the process of data collection for the input. With this algorithm, automation of collecting part connectivity information is achieved. There is a need for the integration of the algorithm presented in this research with the semi-automated part connectivity based assembly time estimation method in order to make the assembly time estimation a completely automated tool. The current part connectivity method for the assembly time estimation uses a Matlab program for performing computations on the bipartite graph. It is required to integrate the SolidWorks add-in developed for this research with the Matlab code so that when the duplicate geometry algorithm is initiated from the SolidWorks the part connectivity graph is exported to the Matlab code for computations and the estimated assembly time is presented back in the SolidWorks software.

REFERENCES

- [1] Babic, B.; Nestic, N.; Miljkovic, Z.: A review of automated feature recognition with rule-based pattern recognition, *Computers in Industry*, 59(4), 2008, 321-337.
- [2] Caldwell, B.; Namouz, E.; Richardson, J.; Sen, C.; Rotenburg, T.; Summers, J.; Mocko, G.; Obieglo, A.: *Automotive Lightweight Engineering: A Method for Identifying Lazy Parts*, *International Journal of Vehicle Design (IJVD)*, in press, 2013.
- [3] Ding, S.; Mannan, M. A.; Poo, A. N.: Oriented bounding box and octree based global interference detection in 5-axis machining of free-form surfaces, *Computer-Aided Design*, 36(13), 2004, 1281-1294.
- [4] Ding, L.; Yue, Y.: Novel ANN-based feature recognition incorporating design by features, *Computers in Industry*, 55(2), 2004, 197-222.
- [5] Gao S.; Shah, J. J.: Automatic recognition of interacting machining features based on minimal condition subgraph, *Computer-Aided Design*, 30(9), 1998, 727-739.
- [6] Garey, M. R.; Johnson, D. S.: *Computers and Intractability: A Guide to the Theory of NP-Completeness*, 1979.
- [7] Griese, D.; Namouz, E.; Shankar, P.; Summers, J. D.; Mocko, G. M.: Application of a Lightweight Engineering Tool: Lazy Parts Analysis and Redesign of a Remote Controlled Car. ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. ASME, Washington, DC, 2011, DETC2011-47544.
- [8] Han, J.; Pratt, M.; Regli, W. C.: Manufacturing feature recognition from solid models: a status report, *Robotics and Automation*, 16(6), 2000, 782-796.
- [9] Hilaga, M.; Shinagawa, Y.; Kohmura, T.; Kunii, T.: Topology Matching for Fully Automatic Similarity Estimation of 3D Shapes, 28th Annual Conference on Computer Graphics and Interactive Techniques. ACM, Los Angeles, CA, 2001, 203-212.
- [10] Joshi, S.; Chang, T. C.: Graph-based Heuristics for Recognition of Machined Features from a 3D Solid Model, *Computer-Aided Design*, 20(2), 1988, 58-66.
- [11] Kao, C. Y.; Kumara, S. R. T.; Kasturi R.: Extraction of 3D object features from CAD boundary representation using the super relation graph method, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 17(12), 1995, 1228-1233.
- [12] Mathieson, J. L.; Summers, J. D.: Complexity Metrics for Directional Node-Link System Representations: Theory and Applications. ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, ASME, Montreal, Canada, 2010, DETC2010-28561.
- [13] Mathieson, J. L.; Wallace, B. A.; Summers, J. D.: Assembly time modelling through connective complexity metrics, *International Journal of Computer Integrated Manufacturing*, on-line(in press), 2013 1-13.
- [14] Miller, M.; Mathieson, J.; Summers, J. D.; Mocko, G. M.: Representation: Structural Complexity of Assemblies to Create Neural Network Based Assembly Time Estimation Models. ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. ASME, Chicago, IL. 2012, DETC2012-71337.

- [15] Namouz, E. Z.; Mears L.; Summers, J.: Lazy Parts Indication Method: Application to Automotive Components, SAE 2011 World Congress & Exhibition, 2011, 2001-2011.
- [16] Namouz, E. Z.; Summers, J. D.: Complexity Connectivity Metrics-Predicting Assembly Times with Abstract Assembly Models, Smart Product Engineering, M. Abramovici, and R. Stark, eds. Springer Berlin Heidelberg, Bochum, Germany, 2013, 77-786.
- [17] Namouz, E. Z.; Summers, J. D.; Mocko, G. M.; Obieglo, A.: Workshop for Identifying Assembly Time Savings: An OEM Empirical Study. Manufacturing Automation (ICMA), 2010 International Conference on, 2010, 178-185.
- [18] Owensby, J. E.; Namouz, E. Z.; Shanthakumar, A.; Summers, J. D.: Representation: Extracting Mate Complexity from Assembly Models to Automatically Predict Assembly Times. ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, ASME, Chicago, IL. 2012, DETC2012-70995.
- [19] Owensby, E.; Shanthakumar, A.; Rayate, V.; Namouz, E. Z.; Summers, J. D. D.; Owensby J. E.: Evaluation and Comparison of Two Design for Assembly Methods: Subjectivity of Information. ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. ASME, Washington, DC, 2011, DETC2011-47530.
- [20] Regli, W. C.; Gupta S. K.; Nau, D. S.: Extracting alternative machining features: An algorithmic approach, Research in Engineering Design, 7(3), 1995, 173-192.
- [21] Sakurai, H.: Volume decomposition and feature recognition: Part 1 - polyhedral objects, Computer-Aided Design, 27(11), 1995, 833-843.
- [22] Shah, J. J.; Anderson D.; Kim, Y. S.; Joshi S.: A Discourse on Geometric Feature Recognition From CAD Models, Journal of Computing and Information Science in Engineering, 1(1), 2001, 41.
- [23] Shanthakumar, A.: Development of a Feature Recognition Algorithm for Automated Identification of Duplicate Geometries in CAD Models, Clemson University, 2012.
- [24] Simon H. A.: The Sciences of the Artificial, M.I.T Press, Massachusetts, 1970.
- [25] Sommerville M. G. L.; Clark D. E. R.; Corney J. R.: Viewer-centered geometric feature recognition, Journal of Intelligent Manufacturing, 12(4), 2001, 359-375.
- [26] Wang, E.; Kim, Y. S.: Form feature recognition using convex decomposition: results presented at the 1997 ASME CIE Feature Panel Session, Computer-Aided Design, 30(13), 1998, 983-989.
- [27] Wu, M. C.; Lit C. R.: Analysis on machined feature recognition techniques based on B-rep, Computer-Aided Design, 28(8), 1996, 603-616.
- [28] Zheng, Y.; Taib, J. M.; Tap M. M.: Implementation of Heuristic Reasoning to Recognize Orthogonal and Non-Orthogonal Inner Loop Features From Boundary Representation (B-Reps) Parts, Jurnal Mekanikal, 33, 2011, 1-14.