Semantic Tagging Framework for Contextually Augmented Features

SungKu Kang[1], Lalit Patil[2], Michael E. Graham[3], Debasish Dutta[4]

[1]University of Illinois at Urbana-Champaign, skang47@illinois.edu
[2]University of Illinois at Urbana-Champaign, lpatil@illinois.edu
[3]GE Global Research, grahamme@ge.com
[4]Rutgers University, dutta@oq.rutgers.edu

Corresponding author: SungKu Kang, skang47@illinois.edu

Abstract. The value of (semi-)automatically recognizing the meaningful regions on a Computer Aided Design (CAD) model is well documented and has been traditionally studied as Geometric feature recognition. However, the effective tagging of the regions, especially tagging with contextually meaningful (i.e. semantic) keywords, has been elusive. In this paper, we focus on the non-geometric aspects of the tagging problem and present a semantic tagging framework. Specifically, given the contextually augmented instances created from the unidentified regions on a CAD model, the framework tags the instances with semantic keywords defined in the semantic model. This will provide the direct link between a CAD model and semantic models, and thus allow high-level reasoning to support design and manufacturing tasks. The feasibility of the approach has been verified by applying the developed framework to tag the regions on turbine blade models. This paper concludes with the future challenges of the approach.

Keywords: Feature identification, Feature augmentation, Semantic technology, Ontology, Computer-Aided Design, CAD
DOI: https://doi.org/10.14733/cadaps.2020.1-15

1 INTRODUCTION

Recognizing the meaningful features/regions on a 3D product model has been the key task in the integration of Computer-aided Design (CAD), Computer-aided Process Planning (CAPP), and Computer-aided manufacturing (CAM) [34]; typically referred to as Geometric feature recognition. Specifically, Geometric feature recognition minimizes the necessity of manual re-interpretation of design for process planning and manufacturing [31, 10], and streamlines the link between design and manufacturing processes.

Geometric feature recognition has been implemented with various methods. Rule-based methods including syntactic pattern recognition and graph-based methods are traditionally studied, and Artificial Neural Network

(ANN)-based and Ontology-based methods are recently studied [31, 5]. For the further improvement of Geometric feature recognition, the efforts have focused on the performance of geometric reasoning. However, the methods did not add significant additional values on Geometric feature recognition. In the meantime, even though the features/regions also represent their semantics in respect of design and manufacturing context, the semantic aspect of Geometric feature recognition has not been focused. Specifically, the semantics of the regions recognized by current Geometric feature recognition application is often overlooked or implicitly embedded in the application. Moreover, Geometric feature recognition has utilized solely geometry information and rarely considered the non-geometric relations between the features/regions and context to recognize individual features/regions. In short, to the best of knowledge, little (or no) research has focused on the problem of applying semantically meaningful tags on the features/regions considering non-geometric context.

In this paper, we present a semantic tagging framework to address the two issues, the lack of augmenting the features/regions with semantically meaningful tags and the lack of considering non-geometric relations in the tagging problem. It should be noted that the framework is designed to complement existing Geometric feature recognition systems, rather than replacing them, and enhance the recognized features/regions with semantically meaningful tags. The framework utilizes two components which are a semantic model and semantic reasoning engine using formal rules. First, we designed a semantic model that defines semantically meaningful tags, i.e., semantic tags for a product model. The semantic tags represent semantic aspect of features/regions along with the hierarchies/relations between them. Second, semantic reasoning engine utilizes formal rules to recognize the features/regions of interest and apply the semantic tags. Each formal rule defines a recognition rule that applies a semantic tag on a feature/region based on the non-geometric relation between the feature/region and adjacent ones. Specifically, given a feature/region with pre-defined semantic tag and the non-geometric relations between the feature/region and the others, the semantic reasoning engine iteratively applies semantic tags to all the other unidentified features/regions. To verify the feasibility of the framework, we also conducted case studies by applying the framework to the problem of tagging features/regions on several turbine blade models.

We expect that the semantic tagging framework will be able to complement existing Geometric feature recognition systems by contextually augmenting the features/regions with semantic tags defined in the semantic model. Specifically, semantic tags will provide a way to represent the semantics of a feature/region in respect of different design and manufacturing context. Therefore, performing high-level reasoning related to the combination of geometric and non-geometric context becomes feasible. Furthermore, the framework will benefit the adoption of semantic technology in design and manufacturing by providing an efficient way to directly link a 3D product model with semantic models.

The remainder of this paper is organized as follows. First, the related works are reviewed in Section 2. Then the problem definition and the overview of the semantic tagging framework is shown in Section 3 and Section 4 respectively. A detailed explanation of the framework including the semantic model and semantic reasoning engine will be shown from Section 5 to Section 6. The implementation of the framework and case studies are shown in Section 7, and the generalization of the proposed method is shown in Section 8. Lastly, the conclusion, the impact of the paper, and future challenges are provided in Section 9.

2 RELATED WORKS

2.1 Geometric feature recognition

Geometric feature recognition is the task of recognizing meaningful regions on a product model. In this section, the existing Geometric feature recognition methods are reviewed. Based on the existing literature reviews [31, 5], the methods can be divided into two categories. The first category is rule-based pattern recognition, which has been traditionally studied. As its name implies, the methods in this category utilize geometric rules to recognize the region of interest from the rest of the product model. Syntactic pattern recognition, state transition diagrams and automata, logic rules and expert systems, graph-based method, convex hull volumetric decomposition method, cell-based volumetric decomposition method, hint-based method, and hybrid method are included in this category. The second category is relatively recent methods, which do not belong to the rule-based pattern recognition. Neural network-based method and ontology-based method are included in this category. Each of these methods is explained in the rest of this section.

Syntactic pattern recognition method [18, 16, 17] is the earliest concept of Geometric feature recognition, introduced by Kyprianou [22]. In this method, a CAD model is projected into 2D in the form of a string, and the string is analyzed by a parser to recognize certain patterns matching to known features. While the method worked well for 2D prismatic features, 3D geometries should be projected into 2D to be processed. Due to this limitation, the method has not been studied recently.

Similar to syntactic pattern recognition method, state transition diagrams and automata-based method also translates a CAD model into the form of string. The method uses space sweeping operators for the representation of a CAD model and uses state transition diagrams and automata to identify the features. Sharing the same limitation of syntactic pattern recognition method, this method has not received attention except for the method developed in 1979 under the supervision of Prof. K. Iwata [36, 7].

Logic rules and expert systems-based method are introduced by Henderson and Anderson [14]. In this method, a set of production rules (i.e. IF-THEN rules) is used to implement Geometric feature recognition. The method can be regarded as a kind of generalization of syntactic methods, and it is more robust than other syntactic methods. However, the method is not flexible, and not able to handle some ambiguous representations. The method of Sadaiah et al. [26] also belongs to the method.

Graph-based method is probably the most popular method in this field, which translates a CAD model into a B-rep model, which can represent a topological and geometric information, including convex or concave adjacency. From the B-rep model, features are recognized by matched to pre-defined graphs representing known features. However, the extensive cost of constructing a B-rep model and the lack of capability to recognize the manufacturability of the recognized features are the limitation of the method. The limitations are usually relieved by adopting hybrid methods, and construct rich feature libraries. The methods of Gravankar and Henderson [11], Yuen and Venuvinod [35], and Ji and Marefat [19] belong to the method. The method of Stefano et al. [9] proposed a B-rep model-based method that also utilizes semantics of the features of a component from a geometric representation. However, their method does not incorporate non-geometric semantics so it shares the same limitation as the other methods.

Convex hull and cell-based decomposition method is introduced as the alternative to traditional B-rep models; the method differentiates itself from other B-rep model-based methods by decomposing a CAD model into a set of positive and negative intermediate volumes or cells. The subset of the set is recognized as a feature of interest when it matches to the representation of known features. Unlike other B-rep model-based methods, the method is history-independent and capable of storing additional information on features, including the list of faces not on a part boundary. The methods of Woo [29, 33] and Kim [20] belong to the method.

Hint-based method utilizes topological, geometrical and heuristic information about a part as the hints of the presence of a specific feature. Because of the nature of this method, it can be regarded as the combination of logic-based method and graph-based method. It should be noted that the duplicated hints for multiple presence rules sometimes make the method ineffective. The methods of Vandenbrande and Requicha [30], McCormack and Ibrahim [15], and Sommerville et al. [27] belong to this method.

Neural-network-based method recognizes a feature by training an artificial neural network (ANN); and the method has received attention since the use of ANN has been shown effective in many fields. The effectiveness comes from the fact that ANN can learn by examples. When enough number of examples are given, ANN-based method can provide extremely high accuracy. In spite of its geometric reasoning power, ANN is not the most effective method since ANN cannot perform logical operations. Due to the limitation, successful recognition rate is obtained only when the input features of the ANN are well chosen. The methods of Prabhakar et al. [24], Nezis et al. [21], Charkraborty and Basu [6], and Sunil and Pande [28] belong to the method.

In addition to the above methods, some of the methods adopted ontologies in Geometric feature recognition.

Garcia et al. [10] designed an ontology that includes the classes and properties corresponding to primitive geometries, and wrote SQWRL [23] rules to implement Geometric feature recognition. Wang et al. [32] designed an ontology that includes the classes and properties corresponding to STEP schema, and then their system read STEP file of a CAD model and perform reasoner on the model to implement Geometric feature recognition. Zhang et al. [37] demonstrated the ontology-based concept model for representing the machining faces and features, annotated geometric faces using domain ontology, and recognized the faces based on the annotation. While the methods tried introducing ontology in Geometric feature recognition, the methods focused on introducing the ontology-based versions of the other methods by exclusively considering geometric context, rather than considering non-geometric context with an ontology.

2.2 Review of the Relevant Works

As mentioned, the previous efforts have introduced various methods to improve geometric reasoning for the Geometric feature recognition. However, augmenting the features/regions with semantic tags and complementing recognition process by non-geometric relations have not received enough focus. As a result, while the performance of geometric reasoning has been continuously improved, the advance of Geometric feature recognition did not add significant additional values, such as providing the combination of geometric and non-geometric context for high-level reasoning to support design and manufacturing.

To address the challenge, we demonstrate the semantic tagging framework that utilizes a semantic model and a semantic reasoning engine. First, the semantic model provides semantic tags, which incorporate non-geometric information, to augment features/regions of a CAD model. Second, the semantic reasoning engine utilizes a set of formal rules to perform the tagging process based on non-geometric information.

3 PROBLEM DEFINITION

3.1 The motivating example

The overall problems are the lack of semantic tags for features/regions of a product model and the lack of considering non-geometric information in recognizing process. Figure 1 illustrates the problems through the motivating example of semantic tagging framework. The input (the left side of Figure 1) represents a turbine blade model with its meaningful features/regions recognized by a Geometric feature recognition system. It should be noted that the recognized features/regions do not provide semantically meaningful context as they do not have semantic tags on them (marked as "?"). The main function of the semantic tagging framework is deriving the desired output, which is the same turbine blade model with the features/regions tagged with semantic tags (the right side of Figure 1). Developing the desired framework requires addressing the following requirements:

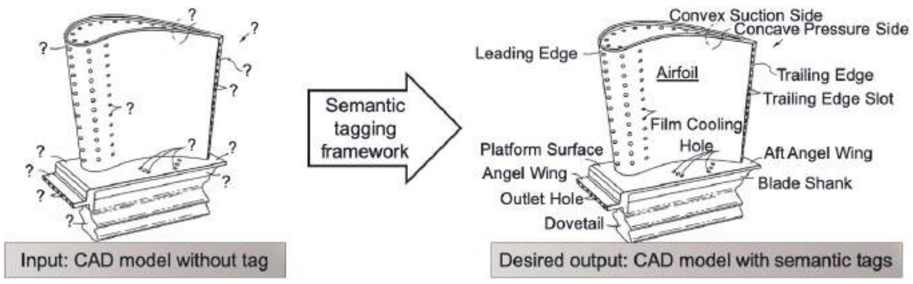


Figure 1: The motivating example of semantic tagging framework.

The semantic model to represent the semantic tags and non-geometric relations CAD model is not a good way to represent non-geometric information including semantic tags and inter-region relations. Therefore, an additional model is required to represent them. In this paper, we designed an ontology-based semantic model to fulfill the requirement.

The mechanism to perform the tagging process Once the semantic model is available, a mechanism to perform the actual tagging process is required. In this paper, we designed a set of tagging rules, so that a semantic reasoner can execute the rules to perform the automated tagging process.

4 OVERVIEW OF SEMANTIC TAGGING FRAMEWORK

Figure 2 shows the overview of the semantic tagging framework and what the input and output of the framework look like. The input (the left side of Figure 2) of the framework is a product model and disjointed regions on it. In this paper, we assume that a Geometric feature recognition system recognizes the disjointed regions from the CAD model. It should be noted that the regions are unidentified, i.e., do not have semantic tags on them. On the other hand, The output (the right side of Figure 2) is the product model with the semantic tags on the regions. In the example, the three regions are tagged as "Airfoil", "Platform", and "Dovetail", respectively.

The middle side of Figure 2 shows the primary focus of this paper, the semantic tagging framework. The framework includes the semantic model of turbine blade models and the semantic reasoning engine. Specifically, the framework augments the input based on the semantic model and put semantic tags on the regions with the semantic reasoning engine. In the following sections, each component of the framework is explained.

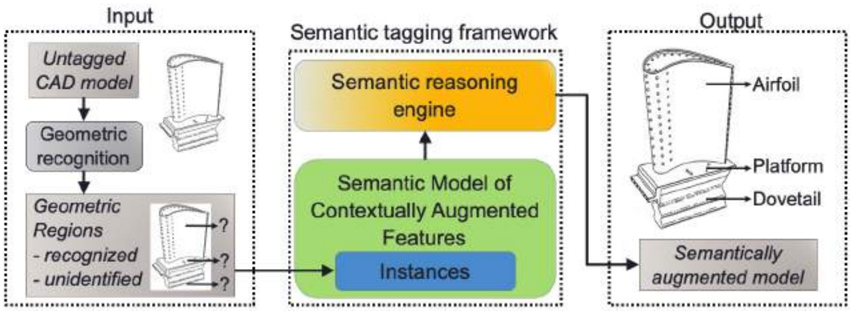

Input
Untagged CAD model
Geometric recognition
Geometric Regions
- recognized
- unidentified
?
?
?
Semantic tagging framework
Semantic reasoning engine
Semantic Model of Contextually Augmented Features
Instances
Output
Airfoil
Platform
Dovetail
Semantically augmented model

Figure 2: The overview of the semantic tagging framework.

5 THE SEMANTIC MODEL OF A PRODUCT MODEL

As stated in the previous section, developing the representation for semantic tags and inter-region relations is the first prerequisite to develop the semantic tagging framework. In this paper, we designed an ontology-based semantic model to address the need. Figure 3 shows the overview of the semantic model, which consists of a domain ontology and augmented instances. The representative descriptions of the domain ontology and augmented instances are provided in the rest of this section.

5.1 Domain ontology

Ontology is typically referred as "an explicit specification of a conceptualization" [12], and domain ontology is an ontology specifically designed to represent a domain of interest. In other words, a domain ontology

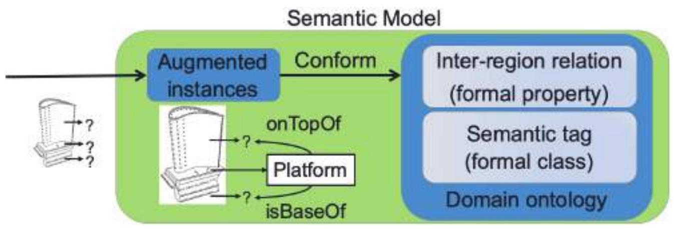


Figure 3: The components of the semantic model including the domain ontology and augmented instances.

introduces domain-specific concepts and their properties (attribute of a concept and inter-concept relationship) to represent the domain knowledge in a machine-understandable form. In this paper, we created a domain ontology representing semantically meaningful keywords to augment features/regions on a turbine blade. Specifically, each concept of the turbine blade domain ontology represents a semantic tag for features/regions on a turbine blade, and each property of the domain ontology represents a possible inter-region relation or an attribute of features/regions.

In creating the turbine blade model domain ontology, we referred to various sources including the inputs from domain experts, patents, and research literature about turbine blades. First, the names of the turbine blade features/regions are collected from the sources. Some of them (e.g. "Airfoil", "Platform", and "Dovetail") are regarded as the meaningful tags, and chosen as the concepts of the turbine blade domain ontology. On the other hand, domain-specific terms representing the spatial relations between the turbine blade features/regions (e.g. "Aft", "Top", and "Starboard") are also collected from the sources, and some of them are chosen as the properties of the turbine blade domain ontology representing the inter-region relations. More details on the concepts and properties are discussed in the following subsections.

5.1.1 Concepts representing semantic tags

As mentioned, the names of turbine blade features/regions are collected to define the concepts representing semantic tags. Figure 4 illustrates the nomenclature of a turbine blade, and the some of the corresponding concepts of the turbine blade domain ontology. It should be noted that there are two different levels of concepts, low-level and high-level concepts. First, low-level concepts represent each disjointed feature/region/cavity on a turbine blade. For example, each name attached on the turbine blade model of Figure 4, such as "Leading Edge", "Platform Surface", and "Trailing Edge Slot", represents a low-level concept. These low-level features can be used to represent the semantics of specific features/regions of interest. On the other hand, there are three high-level concepts, "Airfoil", "Platform", and "Dovetail", which represent major sections of a turbine blade respectively. Rather than representing the semantics of a specific feature/region, these high-level concepts can represent the overall semantics of a set of features/regions belong to one of the sections. For example, a high-level concept "Airfoil" can represent the overall semantics of multiple low-level concepts including "Leading Edge", "Convex Suction Side", and "Trailing Edge". By providing two different levels of concepts, the turbine blade domain ontology provides more flexibility in representing the semantics of turbine blade features/regions.

In this paper, we used Semantic Application Design Language (SADL) [8] to create the semantic model. Figure 5 shows the SADL representation of concepts in the turbine blade domain ontology. It should be noted that the attributes of the features/regions are also defined in the turbine blade domain ontology. Specifically, the bottom side of Figure 5 shows that the concept *VolumeProperty* represents the attribute if a feature/region has positive volume or negative volume (i.e., cavity). For example, the concept such as "AirfoilWall" will have "PositiveVolume" as its attribute, while the concept such as "TrailingEdgeSlot" will have "NegativeVolume" as its attribute.

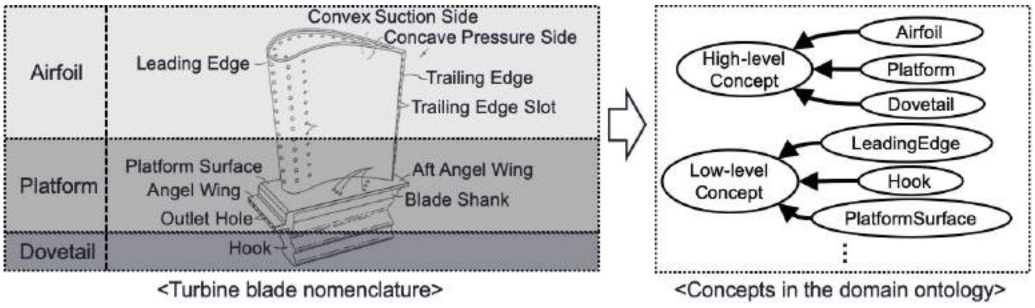
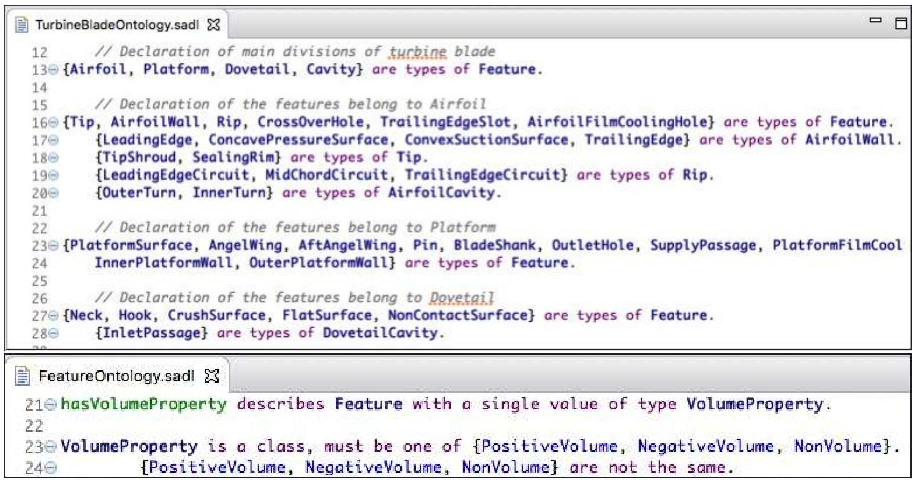


Figure 4: The nomenclature of a turbine blade (left) and the creation of concepts in the turbine blade domain ontology (right). Each section or disjoint feature represents a high or lower-level concept respectively.



```
    // Declaration of main divisions of turbine blade
{Airfoil, Platform, Dovetail, Cavity} are types of Feature.

    // Declaration of the features belong to Airfoil
{Tip, AirfoilWall, Rip, CrossOverHole, TrailingEdgeSlot, AirfoilFilmCoolingHole} are types of Feature.
    {LeadingEdge, ConcavePressureSurface, ConvexSuctionSurface, TrailingEdge} are types of AirfoilWall.
    {TipShroud, SealingRim} are types of Tip.
    {LeadingEdgeCircuit, MidChordCircuit, TrailingEdgeCircuit} are types of Rip.
    {OuterTurn, InnerTurn} are types of AirfoilCavity.

    // Declaration of the features belong to Platform
{PlatformSurface, AngelWing, AftAngelWing, Pin, BladeShank, OutletHole, SupplyPassage, PlatformFilmCool
    InnerPlatformWall, OuterPlatformWall} are types of Feature.

    // Declaration of the features belong to Dovetail
{Neck, Hook, CrushSurface, FlatSurface, NonContactSurface} are types of Feature.
    {InletPassage} are types of DovetailCavity.
```

```
hasVolumeProperty describes Feature with a single value of type VolumeProperty.

VolumeProperty is a class, must be one of {PositiveVolume, NegativeVolume, NonVolume}.
        {PositiveVolume, NegativeVolume, NonVolume} are not the same.
```

Figure 5: The SADL representation of turbine blade concepts (top), and their attributes (bottom).

5.1.2 Properties representing spatial relationships

Similar to the creation of the concepts in the domain ontology, the terms representing spatial relationships between the turbine blade features/regions are collected from the sources to define the properties of the turbine blade domain ontology. Figure 4 illustrates the orientation of the spatial relationships of a turbine blade, and the corresponding properties of the turbine blade domain ontology. Specifically, "Fore", "Aft", "Port", "Starboard", "Top", "Base" represent the relative position (front, back, left, right, top, bottom respectively) of a feature from another feature. For example, based on Figure 4, it is shown that the position of "Angel Wing" is "Fore" of "Platform Surface", while the position of "Blade Shank" is "Base" of "Platform Surface". It should be noted that an additional property "AdhereTo" is created as the super-property of all the spatial relationships for representing adjacency of features/regions. In other words, if any of spatial relationships are specified between two features/regions, the features/regions are automatically "AdhereTo" each other at the same time.

Figure 7 shows the SADL representation of the properties representing spatial relationships between turbine blade features/regions. As mentioned, all the properties representing spatial relationships are declared as the sub-properties of "AdhereTo". In addition, the inverse relations between the properties are also declared. For example, "onTopOf" is declared as the inverse property of "isBaseOf", and "isPortOf" is declared as the inverse

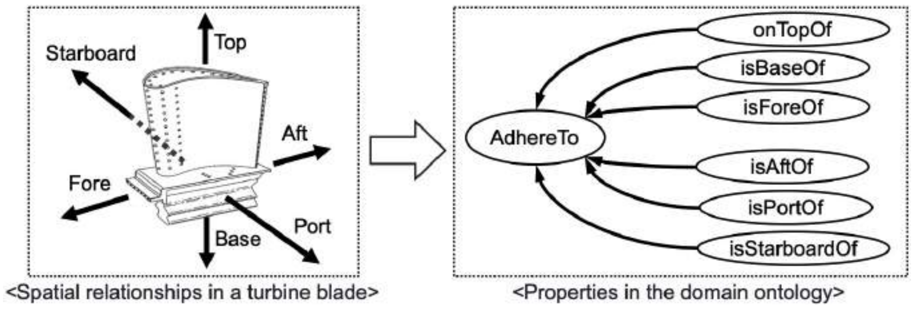
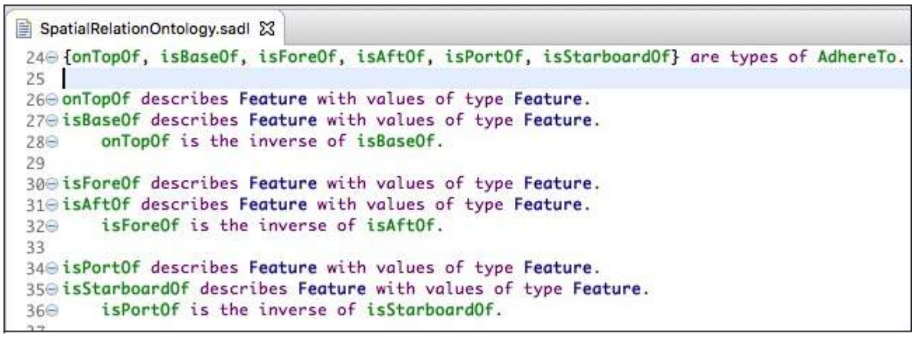


Figure 6: The spatial relationships between turbine blade features (left) and corresponding properties in the domain ontology (right).

property of "isPortOf". In other words, the statement "Region_A onTopOf Region_B" also implies "Region_B isBaseOf Region_A".



```
{onTopOf, isBaseOf, isForeOf, isAftOf, isPortOf, isStarboardOf} are types of AdhereTo.

onTopOf describes Feature with values of type Feature.
isBaseOf describes Feature with values of type Feature.
    onTopOf is the inverse of isBaseOf.

isForeOf describes Feature with values of type Feature.
isAftOf describes Feature with values of type Feature.
    isForeOf is the inverse of isAftOf.

isPortOf describes Feature with values of type Feature.
isStarboardOf describes Feature with values of type Feature.
    isPortOf is the inverse of isStarboardOf.
```

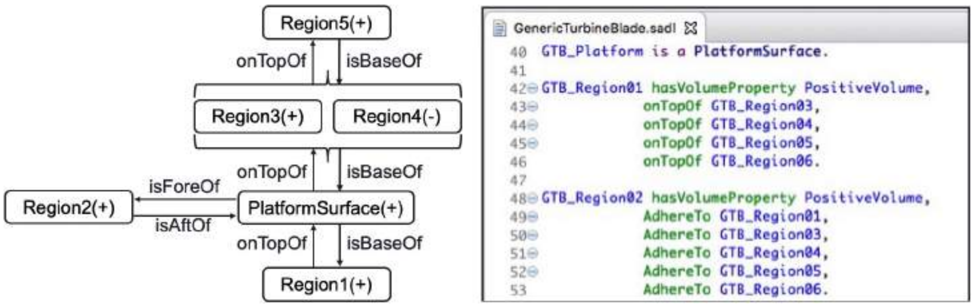
Figure 7: The SADL representation of the properties representing spatial relationships.

5.2 Augmented instances

Given the turbine blade domain ontology, an arbitrary turbine blade model with unidentified regions can be represented as a set of augmented instances conforming to the domain ontology. Figure 8 illustrates the Entity-Relation diagram (left) and the SADL representation (right) of the augmented instances created from unidentified features/regions of a simplified turbine blade model. In the Entity-Relation diagram, Each instance corresponds to each unidentified region (shown as the rounded box) and is augmented with the spatial relations (e.g. *onTopOf* and *isForeOf* shown as the arrows). The (+) mark and (-) mark mean if the region has positive or negative volume. It should be noted that during the tagging process, we manually put a semantic tag on one of the regions to provide the foundation of the following automated tagging. For example, in Figure 8, it is shown that *PlatformSurface* is regarded as the region with a known semantic tag.

6 SEMANTIC REASONING ENGINE

While the semantic model can represent the semantic tags and the inter-region relations, the model is static in nature. The second component of the semantic tagging framework is the semantic reasoning engine, which updates the augmented instances in the static model into identified instances. Figure 9 shows the components of the semantic tagging engine, which are formal tagging rule and semantic reasoner.



```
GTB_Platform is a PlatformSurface.

GTB_Region01 hasVolumeProperty PositiveVolume,
             onTopOf GTB_Region03,
             onTopOf GTB_Region04,
             onTopOf GTB_Region05,
             onTopOf GTB_Region06.

GTB_Region02 hasVolumeProperty PositiveVolume,
             AdhereTo GTB_Region01,
             AdhereTo GTB_Region03,
             AdhereTo GTB_Region04,
             AdhereTo GTB_Region05,
             AdhereTo GTB_Region06.
```

Figure 8: The Entity-Relation diagram (left) and The SADL representation (right) of the augmented instances created from a simplified turbine blade model with a known feature (*PlatformSurface*).

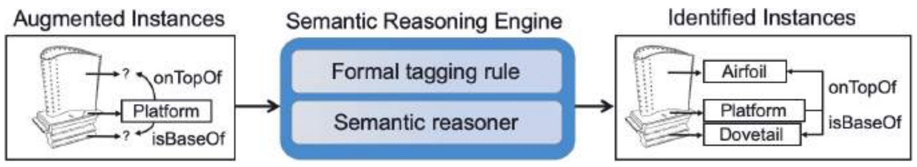
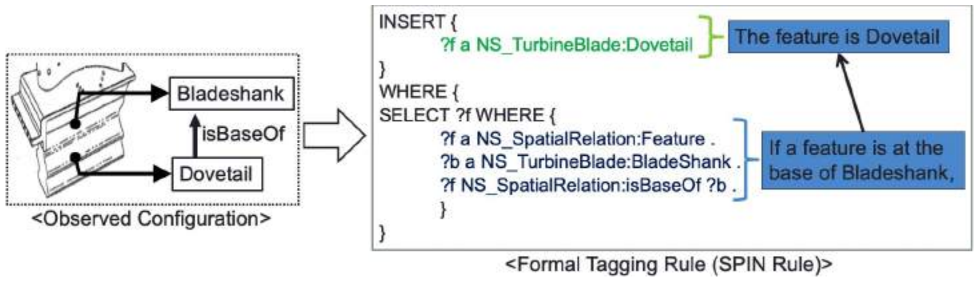


Figure 9: The overview of the semantic reasoning engine.

First, based on the sources and the actual configuration of features/regions of turbine blade models, formal tagging rules are designed to define the semantic tagging tasks. Figure 10 shows the description of the creation of a formal tagging rule used in this paper. Specifically, the configuration between "Bladeshank" and "Dovetail" is generalized to create the formal tagging rule, which identifies "Dovetail" given adjacent "Bladeshank" and their spatial relations. Similar to the example, each formal tagging rule is written to apply a specific semantic tag on an unidentified feature/region based on the type of adjacent features/regions and spatial relations between them. In our work, we used SPARQL Protocol and RDF Query Language (SPARQL) [13] and SPARQL Inferencing Notation (SPIN) [2] to write the formal tagging rules. Figure 10 also shows the formal tagging rule written in SPIN Rule (right). Specifically, the rule checks if there is an unidentified feature/region at the base of "Bladeshank", and apply the semantic tag "Dovetail" if such feature exists.


Bladeshank
isBaseOf
Dovetail
<Observed Configuration>
The feature is Dovetail
If a feature is at the base of Bladeshank,
<Formal Tagging Rule (SPIN Rule)>

```
INSERT {
        ?f a NS_TurbineBlade:Dovetail
}
WHERE {
SELECT ?f WHERE {
        ?f a NS_SpatialRelation:Feature .
        ?b a NS_TurbineBlade:BladeShank .
        ?f NS_SpatialRelation:isBaseOf ?b .
        }
}
```

Figure 10: The observed configuration between "Bladeshank" and "Dovetail" (left), and the formal tagging rule (SPIN Rule) created by generalizing their inter-region configuration (right).

Once the tagging procedure is specified by the formal tagging rules, the semantic reasoner performs the actual tagging process. Specifically, the reasoner reads the formal tagging rules and iteratively executes each rule to apply a tag on each unidentified region until there is nothing to update. Figure 11 illustrates the tagging process. In this paper, the semantic reasoner provided by Apache Jena [1] is used.

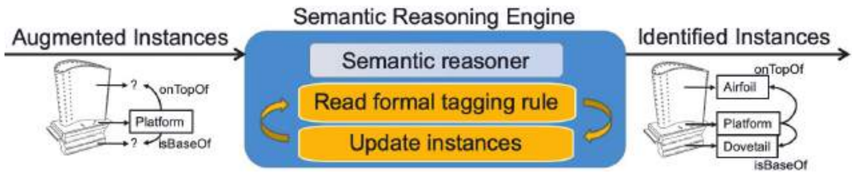


Figure 11: The tagging process performed by the semantic reasoner. The semantic reasoner iteratively executes the formal tagging rules until there is nothing to update.

7 CASE STUDIES

To demonstrate the feasibility of the semantic tagging framework, we conducted several case studies using publicly available turbine blade models (from patents and research literature) and conversed with subject matter experts from the industry to ensure the validity of the case studies.

7.1 Case 1: Publicly available turbine blade models, which are used to develop the framework

The first case study is performed by applying the semantic tagging framework to two turbine blade models which are used in the framework development. Figure 12 illustrates the simplified version of the input and the output of the first case study. As shown in the left side of Figure 12, it is assumed that the input represents geometric regions recognized, but the tags for the regions are unidentified except for the "PlatformSurface" because it should provide the foundation of the following tagging process. The right side of Figure 12 shows the output, which indicates that the framework was able to tag all the regions correctly.

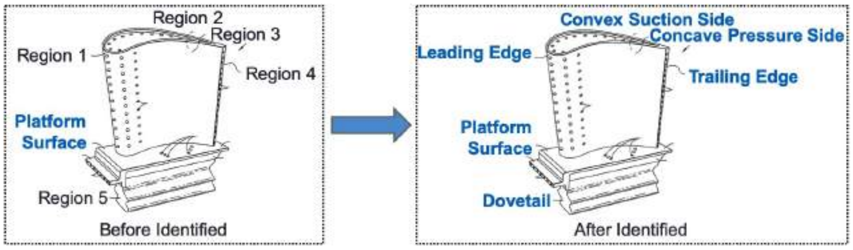
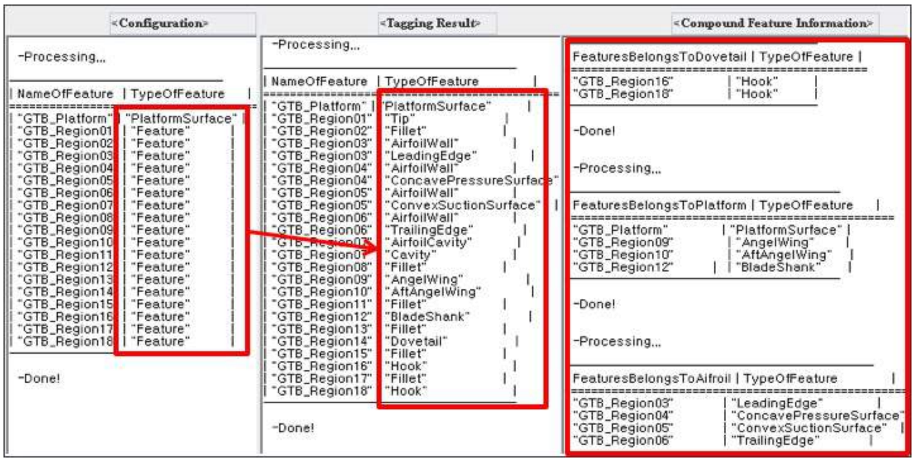


Figure 12: The illustration of the first case study. Given "PlatformSurface", the semantic tagging framework was able to tag the other regions correctly.

Figure 13 shows the actual output of the framework. It should be noted that both low-level concepts and high-level concepts are specified in the result. Specifically, the middle side of Figure 13 shows how low-level concepts are assigned to each of the input regions, while the right side of Figure 13 shows the list of regions that belong to each of high-level concepts. For example, it is shown that the instance *GTB_ Region03* has the semantic tag "Leading Edge" and "Airfoil Wall" representing low-level concepts, and belongs to the group of regions consisting the high-level concept "Airfoil".

7.2 Case 2: The turbine blade model, which is not used for the framework development

For the second case study, the semantic tagging framework was applied to a new turbine blade model. The input turbine blade model conforms to the semantic model we created and the other conditions are the same



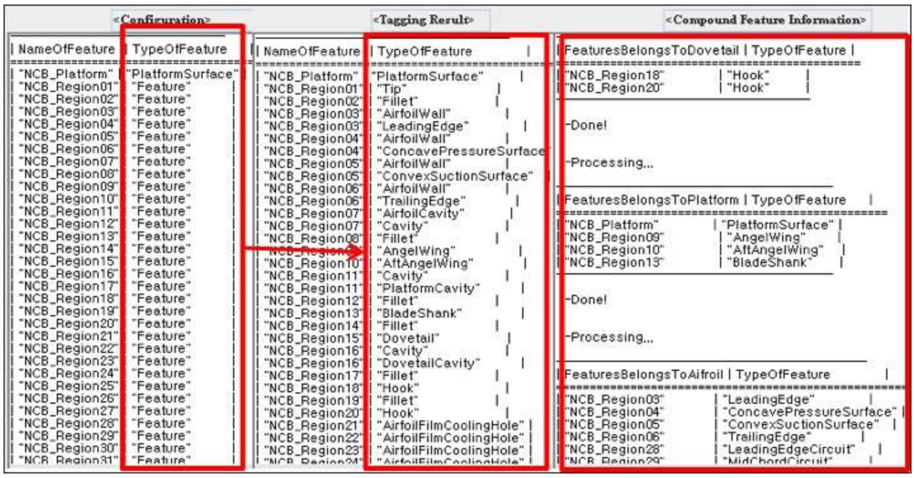
```
-Processing...

| NameOfFeature  | TypeOfFeature     |
======================================
| "GTB_Platform" | "PlatformSurface" |
| "GTB_Region01  | "Feature"         |
| "GTB_Region02  | "Feature"         |
| "GTB_Region03  | "Feature"         |
| "GTB_Region04  | "Feature"         |
| "GTB_Region05  | "Feature"         |
| "GTB_Region06  | "Feature"         |
| "GTB_Region07  | "Feature"         |
| "GTB_Region08  | "Feature"         |
| "GTB_Region09  | "Feature"         |
| "GTB_Region10  | "Feature"         |
| "GTB_Region11  | "Feature"         |
| "GTB_Region12  | "Feature"         |
| "GTB_Region13  | "Feature"         |
| "GTB_Region14  | "Feature"         |
| "GTB_Region15  | "Feature"         |
| "GTB_Region16  | "Feature"         |
| "GTB_Region17  | "Feature"         |
| "GTB_Region18  | "Feature"         |

-Done!
```

```
-Processing...

| NameOfFeature  | TypeOfFeature              |
===============================================
| "GTB_Platform" | "PlatformSurface"          |
| "GTB_Region01"  "Tip"                    |
| "GTB_Region02"  "Fillet"                 |
| "GTB_Region03"  "AirfoilWall"            |
| "GTB_Region03"  "LeadingEdge"            |
| "GTB_Region04"  "AirfoilWall"            |
| "GTB_Region04"  "ConcavePressureSurface"
| "GTB_Region05"  "AirfoilWall"            |
| "GTB_Region05"  "ConvexSuctionSurface"   |
| "GTB_Region06"  "AirfoilWall"            |
| "GTB_Region06"  "TrailingEdge"           |
| "GTB_Region07"  "AirfoilCavity"          |
| "GTB_Region07   "Cavity"                 |
| "GTB_Region08"  "Fillet"                 |
| "GTB_Region09"  "AngelWing"              |
| "GTB_Region10"  "AftAngelWing"           |
| "GTB_Region11"  "Fillet"                 |
| "GTB_Region12"  "BladeShank"             |
| "GTB_Region13"  "Fillet"                 |
| "GTB_Region14"  "Dovetail"               |
| "GTB_Region15"  "Fillet"                 |
| "GTB_Region16"  "Hook"                   |
| "GTB_Region17"  "Fillet"                 |
| "GTB_Region18"  "Hook"                   |

-Done!
```

```
FeaturesBelongsToDovetail | TypeOfFeature |
============================================
"GTB_Region16"            | "Hook"        |
"GTB_Region18"            | "Hook"        |

-Done!

-Processing...

FeaturesBelongsToPlatform | TypeOfFeature     |
================================================
"GTB_Platform"           | "PlatformSurface" |
"GTB_Region09"            | "AngelWing"       |
"GTB_Region10"            | "AftAngelWing"    |
"GTB_Region12"         |  | "BladeShank"      |

-Done!

-Processing...

FeaturesBelongsToAifroil | TypeOfFeature            |
=====================================================
"GTB_Region03"           | "LeadingEdge"            |
"GTB_Region04"           | "ConcavePressureSurface"
"GTB_Region05"           | "ConvexSuctionSurface"   |
"GTB_Region06"           | "TrailingEdge"           |
```

Figure 13: The actual output from the first case study. The regions are correctly labeled with semantic tags.

as the previous case. Figure 14 shows the actual output from the framework. We observed that all the regions were tagged correctly. The result indicates that the framework performs well if the input conforms to the underlying turbine blade domain ontology.



```
| NameOfFeature  | TypeOfFeature     |
======================================
| "NCB_Platform" | "PlatformSurface" |
| "NCB_Region01"   "Feature"         |
| "NCB_Region02"   "Feature"         |
| "NCB_Region03"   "Feature"         |
| "NCB_Region04"   "Feature"         |
| "NCB_Region05"   "Feature"         |
| "NCB_Region06"   "Feature"         |
| "NCB_Region07"   "Feature"         |
| "NCB_Region08"   "Feature"         |
| "NCB_Region09"   "Feature"         |
| "NCB_Region10"   "Feature"         |
| "NCB_Region11"   "Feature"         |
| "NCB_Region12"   "Feature"         |
| "NCB_Region13"   "Feature"         |
| "NCB_Region14"   "Feature"         |
| "NCB_Region15"   "Feature"         |
| "NCB_Region16"   "Feature"         |
| "NCB_Region17"   "Feature"         |
| "NCB_Region18"   "Feature"         |
| "NCB_Region19"   "Feature"         |
| "NCB_Region20"   "Feature"         |
| "NCB_Region21"   "Feature"         |
| "NCB_Region22"   "Feature"         |
| "NCB_Region23"   "Feature"         |
| "NCB_Region24"   "Feature"         |
| "NCB_Region25"   "Feature"         |
| "NCB_Region26"   "Feature"         |
| "NCB_Region27"   "Feature"         |
| "NCB_Region28"   "Feature"         |
| "NCB_Region29"   "Feature"         |
| "NCB_Region30"   "Feature"         |
| "NCB_Region31"   "Feature"         |
```

```
| NameOfFeature  | TypeOfFeature              |
===============================================
| "NCB_Platform"   "PlatformSurface"          |
| "NCB_Region01" | "Tip"                    |
| "NCB_Region02" | "Fillet"                 |
| "NCB_Region03" | "AirfoilWall"            |
| "NCB_Region03" | "LeadingEdge"            |
| "NCB_Region04" | "AirfoilWall"            |
| "NCB_Region04" | "ConcavePressureSurface"
| "NCB_Region05" | "AirfoilWall"            |
| "NCB_Region05" | "ConvexSuctionSurface"   |
| "NCB_Region06" | "AirfoilWall"            |
| "NCB_Region06" | "TrailingEdge"           |
| "NCB_Region07" | "AirfoilCavity"          |
| "NCB_Region07" | "Cavity"                 |
| "NCB_Region08" | "Fillet"                 |
| "NCB_Region09"   "AngelWing"              |
| "NCB_Region10" | "AftAngelWing"           |
| "NCB_Region11" | "Cavity"                 |
| "NCB_Region11" | "PlatformCavity"         |
| "NCB_Region12" | "Fillet"                 |
| "NCB_Region13" | "BladeShank"             |
| "NCB_Region14" | "Fillet"                 |
| "NCB_Region15" | "Dovetail"               |
| "NCB_Region16" | "Cavity"                 |
| "NCB_Region16" | "DovetailCavity"         |
| "NCB_Region17" | "Fillet"                 |
| "NCB_Region18" | "Hook"                   |
| "NCB_Region19" | "Fillet"                 |
| "NCB_Region20" | "Hook"                   |
| "NCB_Region21" | "AirfoilFilmCoolingHole" |
| "NCB_Region22" | "AirfoilFilmCoolingHole" |
| "NCB_Region23" | "AirfoilFilmCoolingHole" |
| "NCB_Region24" | "AirfoilFilmCoolingHole" |
```

```
| FeaturesBelongsToDovetail | TypeOfFeature |
==============================================
| "NCB_Region18"            | "Hook"        |
| "NCB_Region20"            | "Hook"        |

-Done!

-Processing...

| FeaturesBelongsToPlatform | TypeOfFeature     |
==================================================
| "NCB_Platform"           | "PlatformSurface" |
| "NCB_Region09"            | "AngelWing"       |
| "NCB_Region10"            | "AftAngelWing"    |
| "NCB_Region13"            | "BladeShank"      |

-Done!

-Processing...

| FeaturesBelongsToAifroil | TypeOfFeature            |
=======================================================
| "NCB_Region03"           | "LeadingEdge"            |
| "NCB_Region04"           | "ConcavePressureSurface" |
| "NCB_Region05"           | "ConvexSuctionSurface"   |
| "NCB_Region06"           | "TrailingEdge"           |
| "NCB_Region28"           | "LeadingEdgeCircuit"     |
| "NCB_Region29"           | "MidChordCircuit"        |
```

Figure 14: The actual output from the second case study. The regions are correctly labeled as well.

7.3 Case 3: The turbine blade model with a wrong initial tag

For the third case study, the input turbine blade model with a wrong initial tag was used, so that we can test if we obtain any false positives. Figure 15 illustrates the simplified version of the input and the output of the third case study. As shown in the left side of Figure 15, it is assumed that the initial tag *PlatformSurface* is not correctly given. The right side of Figure 15 shows the output, which indicates that some of the regions are not correctly tagged or not even tagged due to the wrongly given tag. The result shows that the semantic tagging framework does not generate false positive in case of the incorrect input. Figure 16 shows the actual output of this case.

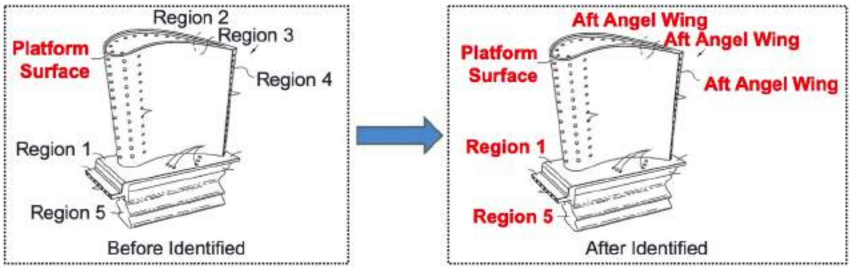
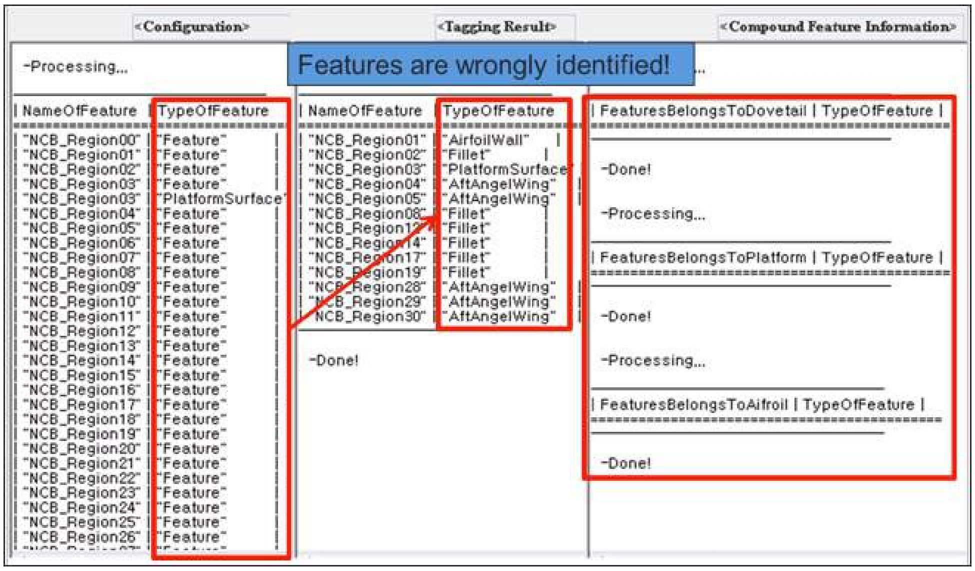


Figure 15: The illustration of the third case study, which assumes a wrongly provided initial tag. In this case, the semantic tagging framework was not able to tag the regions correctly. this.


<Configuration>
<Tagging Result>
<Compound Feature Information>
Features are wrongly identified!

```
-Processing...

| NameOfFeature  | TypeOfFeature
===================================
| "NCB_Region00" | "Feature"         |
| "NCB_Region01" | "Feature"         |
| "NCB_Region02" | "Feature"         |
| "NCB_Region03" | "Feature"         |
| "NCB_Region03" | "PlatformSurface" |
| "NCB_Region04" | "Feature"         |
| "NCB_Region05" | "Feature"         |
| "NCB_Region06" | "Feature"         |
| "NCB_Region07" | "Feature"         |
| "NCB_Region08" | "Feature"         |
| "NCB_Region09" | "Feature"         |
| "NCB_Region10" | "Feature"         |
| "NCB_Region11" | "Feature"         |
| "NCB_Region12" | "Feature"         |
| "NCB_Region13" | "Feature"         |
| "NCB_Region14" | "Feature"         |
| "NCB_Region15" | "Feature"         |
| "NCB_Region16" | "Feature"         |
| "NCB_Region17" | "Feature"         |
| "NCB_Region18" | "Feature"         |
| "NCB_Region19" | "Feature"         |
| "NCB_Region20" | "Feature"         |
| "NCB_Region21" | "Feature"         |
| "NCB_Region22" | "Feature"         |
| "NCB_Region23" | "Feature"         |
| "NCB_Region24" | "Feature"         |
| "NCB_Region25" | "Feature"         |
| "NCB_Region26" | "Feature"         |
```

```
| NameOfFeature  | TypeOfFeature
===================================
| "NCB_Region01" | "AirfoilWall"     |
| "NCB_Region02" | "Fillet"          |
| "NCB_Region03" | "PlatformSurface" |
| "NCB_Region04" | "AftAngelWing"    |
| "NCB_Region05" | "AftAngelWing"    |
| "NCB_Region08" | "Fillet"          |
| "NCB_Region12" | "Fillet"          |
| "NCB_Region14" | "Fillet"          |
| "NCB_Region17" | "Fillet"          |
| "NCB_Region19" | "Fillet"          |
| "NCB_Region28" | "AftAngelWing"    |
| "NCB_Region29" | "AftAngelWing"    |
| "NCB_Region30" | "AftAngelWing"    |

-Done!
```

```
| FeaturesBelongsToDovetail | TypeOfFeature |
=============================================

-Done!

-Processing...

| FeaturesBelongsToPlatform | TypeOfFeature |
=============================================

-Done!

-Processing...

| FeaturesBelongsToAifroil | TypeOfFeature |
=============================================

-Done!
```

Figure 16: The actual output of the third case study. Some of the regions are not correctly tagged or not tagged due to the wrongly given initial tag.

8 EXTENSION OF THE PROPOSED METHOD

The proposed method has been demonstrated on the example of turbine blade features. However, it is important to ensure its extensibility to be applied to different domains for a broader impact. It should be noted that the method can be easily extended as it adopts the ontology-based strategy. A user can follow the procedure in this paper to develop a semantic tagging framework for a specific purpose. In short, the value of this paper lies in providing the methodology to develop a semantic tagging framework that would be customizable as per its purpose, which aligns with the industry expectations. The method can be extended via the following steps:

Development of a domain ontology First, a domain ontology of the application domain is necessary to provide the relevant features and their relations. A user can adopt/extend an existing ontology, or create one by following the procedure presented in Section 5.1. It should be noted that if the domain can be described with an upper ontology such as MSDL [4], this will ensure the interoperability to a similar domain which relies on the same upper ontology.

Creation of augmented instances with a known feature Based on the domain ontology, disjointed features and their relations can be used to augment a CAD model (preferably with a known feature) as presented in Section 5.2, to provide the foundation of the semantic tagging process.

Development of tagging rules Once the domain ontology is created, the configurations between the features are observed to design semantic tagging rules as presented in Section 6. It should be noted that while spatial relationships are used in the turbine blade example, any other semantic relations can be used if the relations can be generalized to develop a tagging rule.

For example, suppose a medical organization would like to develop the semantic tagging framework on human anatomy, they first develop a domain ontology about human anatomy (an ontology such as The Foundational Model of Anatomy Ontology (FMA) [25] can be used as the foundation of the domain ontology for a better interoperability), augment human CAD models with a known feature, and then develop semantic tagging rules (e.g. FMA provides domain-specific spatial relationships between esophagus, trachea, and other organs based on Visible Human Project [3], similar to the spatial relationships between the turbine blade features) to tag the regions of interest.

9 CONCLUSION AND FUTURE WORK

In this paper, we demonstrated the semantic tagging framework that can apply semantic tags on the features/regions provided by existing Geometric feature recognition system. The framework grounded in semantics-based approach with a longer-term view of developing other semantic applications to support the design and analysis of a product model. We verified the feasibility of the framework through the case studies on the turbine blade model and found that the framework works as desired and did not produce a false positive output.

We believe that the semantic tagging framework will considerably increase the value of Geometric feature recognition by augmenting geometric regions with semantically meaningful tags. Specifically, the additional value will benefit the entire design and manufacturing process by providing richer information defined in the semantic model of the product. We expect it will facilitate the adoption of semantic technology in manufacturing field and the employment of the powerful computer reasoning process to complement conventional design and manufacturing tasks.

Although we have made contributions to the research area of semantic tagging of a product, much work needs to be done. Following lists are the challenges we found out, and the future directions of the research:

Interacting features We have currently considered disjointed geometric areas for tagging. However, interacting features, such as a single cavity extending from dovetail to airfoil, pose challenges, and newer ways of capturing these semantics and their reasoning are needed.

Multiple contexts We have considered a single context, i.e., spatial relations between features. It is important to consider additional contexts, e.g. functional relations. This requires the development of ontologies that capture multiple contexts.

Scalability Currently, we have tested the proposed framework with several models of simplified turbine blades. We need to study the impact of using significantly larger product models with an extensive list of tags for regions and contexts.

ORCID

SungKu Kang http://orcid.org/0000-0001-8146-4103
Lalit Patil http://orcid.org/0000-0002-6063-0034
Michael E. Graham http://orcid.org/0000-0001-5869-5026
Debasish Dutta http://orcid.org/0000-0002-3056-5825

REFERENCES

[1] Apache Jena. Available at https://jena.apache.org/.

[2] SPIN - SPARQL Inferencing Notation. Available at http://spinrdf.org/.

[3] Ackerman, M.J.: The visible human project. Proceedings of the IEEE, 86(3), 504–511, 1998.

[4] Ameri, F.; Dutta, D.: An Upper Ontology for Manufacturing Service Description. In ASME 2006 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, vol. 3, 651–661, 2006.

[5] Babić, B.R.; Nešić, N.; Miljković, Z.: Automatic feature recognition using artificial neural networks to integrate design and manufacturing: Review of automatic feature recognition systems. AI EDAM, 25(3), 289–304, 2011.

[6] Chakraborty, S.; Basu, A.: Retrieval of machining information from feature patterns using artificial neural networks. The International Journal of Advanced Manufacturing Technology, 27(7-8), 781–787, 2006.

[7] Chang, T.C.: Expert process planning for manufacturing. Addison-Wesley Longman, 1990.

[8] Crapo, A.; Moitra, A.: Toward a unified english-like representation of semantic models, data, and graph patterns for subject matter experts. International Journal of Semantic Computing, 7(03), 215–236, 2013.

[9] Di Stefano, P.; Bianconi, F.; Di Angelo, L.: An approach for feature semantics recognition in geometric models. Computer-Aided Design, 36(10), 993–1009, 2004.

[10] García, L.E.R.; Garcia, A.; Bateman, J.: An ontology-based feature recognition and design rule checker for engineering. In Workshop "Ontologies come of Age in the Semantic Web"(OCAS2011) 10 th International Semantic Web Conference Bonn, Germany, October 24, 2011, 48–59, 2011.

[11] Gavankar, P.S.; Henderson, M.R.: Graph-based extraction of two-connected morphological features from boundary representations. Journal of Intelligent Manufacturing, 6(6), 401–413, 1995.

[12] Gruber, T.R.: Toward principles for the design of ontologies used for knowledge sharing? International journal of human-computer studies, 43(5-6), 907–928, 1995.

[13] Harris, S.; Seaborne, A.; Prud'hommeaux, E.: SPARQL 1.1 Query Language. W3C Recommendation, 2013. Available at https://www.w3.org/TR/sparql11-query/.

[14] Henderson, M.R.; Anderson, D.C.: Computer recognition and extraction of form features: a cad/cam link. Computers in industry, 5(4), 329–339, 1984.

[15] Ibrhim, R.; McCormack, A.: Process planning using adjacency-based feature extraction. The International Journal of Advanced Manufacturing Technology, 20(11), 817–823, 2002.

[16] Ismail, N.; Abu Bakar, N.; Juri, A.: Feature recognition patterns for form features using boundary representation models. The International Journal of Advanced Manufacturing Technology, 20(8), 553–556, 2002.

[17] Ismail, N.; Bakar, N.A.; Juri, A.: Recognition of cylindrical and conical features using edge boundary classification. International Journal of Machine Tools and Manufacture, 45(6), 649–655, 2005.

[18] Jain, P.; Kumar, S.: Automatic feature extraction in prizcapp. International Journal of Computer Integrated Manufacturing, 11(6), 500–512, 1998.

[19] Ji, Q.; Marefat, M.M.: A dempster–shafer approach for recognizing machine features from cad models. Pattern recognition, 36(6), 1355–1368, 2003.

[20] Kim, Y.S.: Recognition of form features using convex decomposition. Computer-Aided Design, 24(9), 461–476, 1992.

[21] Konstantinos, N.; George, V.: Recognizing shape features using a neural network and heuristics, vol. 29. Computer-Aided Design, 523–539, 1997.

[22] Kyprianou, L.K.: Shape classification in computer-aided design. Ph.D. thesis, University of Cambridge, 1980.

[23] O'Connor, M.J.; Das, A.K.: Sqwrl: A query language for owl. In OWLED, vol. 529, 208–215, 2009.

[24] Prabhakar, S.; Henderson, M.R.: Automatic form-feature recognition using neural-network-based techniques on boundary representations of solid models. Computer-Aided Design, 24(7), 381–393, 1992.

[25] Rosse, C.; Mejino, J.L.: The foundational model of anatomy ontology. In Anatomy Ontologies for Bioinformatics, 59–117. Springer, 2008.

[26] Sadaiah, M.; Yadav, D.; Mohanram, P.; Radhakrishnan, P.: A generative computer-aided process planning system for prismatic components. The International Journal of Advanced Manufacturing Technology, 20(10), 709–719, 2002.

[27] Sommerville, M.; Clark, D.E.; Corney, J.R.: Viewer-centered geometric feature recognition. Journal of Intelligent Manufacturing, 12(4), 359–375, 2001.

[28] Sunil, V.; Pande, S.: Automatic recognition of machining features using artificial neural networks. The International Journal of Advanced Manufacturing Technology, 41(9-10), 932–947, 2009.

[29] Tseng, Y.J.; Joshi, S.: Recognition of interacting rotational and prismatic machining features from 3-d mill-turn parts. International Journal of Production Research, 36(11), 3147–3165, 1998.

[30] Vandenbrande, J.H.; Requicha, A.A.: Spatial reasoning for the automatic recognition of machinable features in solid models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 15(12), 1269–1285, 1993.

[31] Verma, A.K.; Rajotia, S.: A review of machining feature recognition methodologies. International Journal of Computer Integrated Manufacturing, 23(4), 353–368, 2010.

[32] Wang, Q.; Yu, X.: Ontology based automatic feature recognition framework. Computers in Industry, 65(7), 1041–1052, 2014.

[33] Woo, Y.: Fast cell-based decomposition and applications to solid modeling. Computer-Aided Design, 35(11), 969–977, 2003.

[34] Wu, M.C.; Liu, C.R.: Analysis on machined feature recognition techniques based on b-rep. Computer-Aided Design, 28(8), 603–616, 1996.

[35] Yuen, C.; Venuvinod, P.: Geometric feature recognition: coping with the complexity and infinite variety of features. International Journal of Computer Integrated Manufacturing, 12(5), 439–452, 1999.

[36] Zhang, H.C.; Alting, L.: Computerized manufacturing process planning systems, vol. 81. Chapman & Hall London, 1994.

[37] Zhang, Y.; Luo, X.; Zhang, B.; Zhang, S.: Semantic approach to the automatic recognition of machining features. The International Journal of Advanced Manufacturing Technology, 89(1-4), 417–437, 2017.