An Ontology for CAD Data and Geometric Constraints as a Link Between Product Models and Semantic Robot Task Descriptions

Conference Paper · September 2015

DOI: 10.1109/IROS.2015.7353971

CITATIONS
34

READS
1,645

4 authors, including:

Alexander Perzylo
fortiss
34 PUBLICATIONS **783** CITATIONS
SEE PROFILE

Nikhil Somani
A*STAR Singapore
33 PUBLICATIONS **354** CITATIONS
SEE PROFILE

Markus Rickert
fortiss
71 PUBLICATIONS **998** CITATIONS
SEE PROFILE

An ontology for CAD data and geometric constraints as a link between product models and semantic robot task descriptions

Alexander Perzylo[1], Nikhil Somani[1], Markus Rickert[1] and Alois Knoll[2]

***Abstract*— In this paper, we introduce an approach for leveraging CAD description to a semantic level, in order to link additional knowledge to CAD models and to exploit resulting synergy effects.**
This has been achieved by designing a description language, based on the Web Ontology Language (OWL), that is used to define boundary representations (BREP) of objects. This involves representing geometric entities in a semantic meaningful way, e.g., a circle is defined by a coordinate frame and a radius instead of a set of polygons.
Furthermore, the scope of this semantic description language also covers geometric constraints between multiple objects. Constraints can be specified not only on the object level, but down to single edges or faces of an object. This semantic representation is used to improve a variety of applications, ranging from shape-based object recognition to constraint-based robot task descriptions.
Results from a quantitative evaluation are presented to assess the practicability of this approach.

I. Introduction

All aspects of industrial products are usually handled in a process called product lifecycle management (PLM). As part of this approach, the geometric product specifications are typically generated with Computer Aided Design (CAD) software. On the other side, a lot of applications are designed to use polygon-based geometry models, which only represent an approximation of the constructed model. This approach has many shortcomings, e.g., it only allows for a fixed level of detail and lacks the means to describe semantically meaningful geometries. For instance, a set of polygons which form a circular shape cannot easily be recognized as such, despite this information being already available at design time. If a circle and its parameters are described by the CAD model instead, the exact mathematical representation of the geometry is known and a triangulation can be carried out at any time using a level of detail that is most suitable to the current application's requirements. This separation of represented data and use-case dependent calculations is as important as the separation of data and the software that was used to generate it. These paradigms allow flexible sharing and re-use of information.

Apart from polygon based CAD formats, there are other formats that represent mathematical models of the contained geometries, e.g., STEP or IGES. STEP has been developed by the International Organization for Standardization as ISO 10303-21, whereas IGES was defined by the United States National Bureau of Standards as NBSIR 80-1978. Both formats have evolved to become widely accepted standards for CAD data exchange and storage. However, active development of IGES has been discontinued almost two decades ago. Still these formats lack the flexibility to be easily linked with other sources of information. Many pieces of CAD software feature the specification of materials for created geometries, but this information can only be exported to their proprietary data formats and gets lost when the CAD model is exported to STEP or IGES.

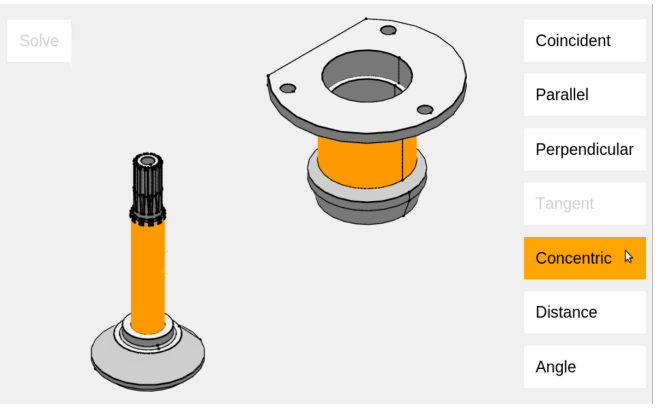


Fig. 1: Graphical user interface for defining geometric constraints between two objects. Available constraints are listed on the right side of the screen. After selecting either a point, curve, or surface on each of the two objects, compatible constraints can be specified. In the depicted example, a *Concentric* constraint is defined between the selected cylindrical surfaces.

In this paper we present a BREP-based (see Section III) semantic representation, i.e., a BREP ontology, of points, curves, surfaces and volumes built using the Web Ontology Language (OWL). A complete specification of BREP's geometric and topological representation can be found in ISO 10303-42. Apart from the already mentioned benefits of having an open and application independent exchange format for CAD data, there are other major advantages. By having a semantic description language that is based on a logical formalism, it is possible to link to, and to automatically combine knowledge from different sources. For instance, a robot's CAD model can be extended by linking it with its kinematic model. Automatic reasoning can be used, e.g., to determine the right tools or parameters for applying

[1]Alexander Perzylo, Nikhil Somani, and Markus Rickert are with fortiss GmbH, Guerickestr. 25, 80805 München, Germany {perzylo|somani|rickert}@fortiss.org
[2]Alois Knoll is with the Department of Informatics VI, Technische Universiät München, Boltzmannstr. 3, 85748 Garching, Germany knoll@in.tum.de

certain production steps on a work piece by considering the workpiece's shape, material, or mass properties.
As the second contribution of this paper, we introduce an extension to the BREP ontology (in Section IV), which adds geometric interrelation constraints between points, curves, and surfaces. When composing a robot task description, e.g., an assembly plan, the BREP-ontology-based representation can be used to parameterize the task with a set of geometric constraints between sub-elements of the involved assembly parts. Figure 1 shows a graphical user interface, which allows the definition of such constraints in an intuitive way. This is one of the example applications explained in Section V.
The third contribution is a software component to automatically convert standard CAD models, e.g., STEP or IGES, to the proposed ontological representation. In order to use this approach in a real system, certain performance criteria are important. A quantitative evaluation of these aspects are explained in Section VI.

II. Related Work

There have been different approaches for enriching CAD models with semantics. OntoCAD [1] is an ontological annotation approach that is based on labeling geometry elements of CAD models using concepts from a CAD ontology. [2] created an ontology for describing CAD models in the Drawing Exchange Format (DXF). In [3], the authors presented an ontology-based approach of semantically describing CAD data and features. They use a rule system to automatically classify CAD features in order to provide a compatible mapping between different CAD systems. [4] showed how not only CAD models but the complete Product Lifecycle Management (PLM) approach can be semantically modeled. The RoboEarth project [5] followed a holistic approach to knowledge representation and sharing for robots. The RoboEarth language [6], which was built using the Web Ontology Language (OWL), was used to describe many aspects of robot tasks, e.g., task structures, semantic environment maps and models of interaction objects. RoboEarth object models are, however, only meta-descriptions for linked industry-standard CAD models.
[7], [8], [9] presented an approach for describing coordinate frames and geometric constraints between the frames for applications in the robotics domain.
In this paper, we introduce a semantic description language to describe not only meta-information for a given CAD model [6], but also the content of the model. Given the complete ontological representation of CAD models, we can define geometric constraints not only on the object level [6] or only on a frame level [8], but also on all semantically meaningful intermediate levels, e.g., points, edges, and faces of CAD models.

III. Boundary Representation (BREP)

A Boundary Representation (BREP) of CAD data describes the geometric properties of points, curves, surfaces and volumes using mathematical models as its basis. CAD models are created by defining boundary limits to given base geometries. The BREP specification distinguishes geometric and topological entities, as illustrated in Figure 2. Geometric entities hold the numerical data, while the topological entities group them and arrange them in a hierarchical fashion.

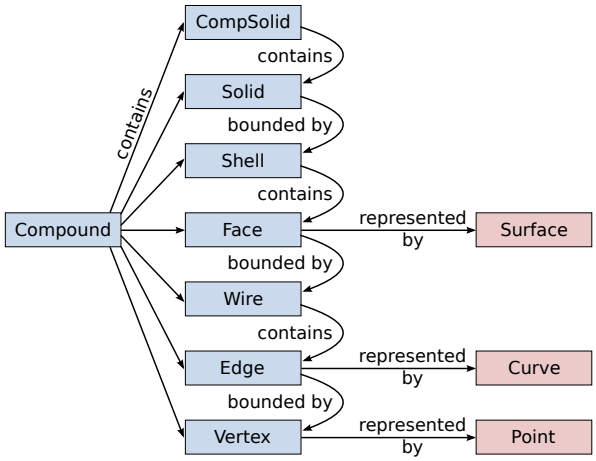


Fig. 2: Overview of the BREP structure. The data model contains topological entities, highlighted in blue, and geometric entities, highlighted in red.

A. Topological Entities

The BREP standard specifies eight kinds of topological entities: *Vertex*, *Edge*, *Face*, *Wire*, *Shell*, *Solid*, *CompSolid*, and *Compound*. Only *Vertices*, *Edges*, and *Faces* have direct links to geometric entities. A *Vertex* is represented by a point. An *Edge* is represented by a curve and bounded by up to two *Vertices*. A *Wire* is a set of adjacent *Edges*. When the *Edges* of a *Wire* form a loop, the *Wire* is considered to be closed. A *Face* is represented by a surface and bounded by a closed *Wire*. A *Shell* is a set of adjacent *Faces*. When the *Faces* of a *Shell* form a closed volume, the *Shell* can be used to define a *Solid*. *Solids* that share common *Faces* can be grouped further into *CompSolids*. *Compounds* are top-level containers and may contain any other topological entity.

B. Geometric Entities

The topological entities may link to three types of geometric entities, which are *Points*, *Curves*, and *Surfaces*. They represent 0-, 1-, and 2-dimensional geometries respectively. *Curves* and *Surfaces* are defined through parameterizable mathematical models. Supported curve types can be categorized as unbounded curves, e.g., lines, parabolas, or hyperbolas, and bounded curves, e.g., Bezier curves, B-spline curves, circles, or ellipses. Offset curves represent a translated version of a given base curve along a certain vector, whereas trimmed curves bound a given base curve by limiting the minimum and maximum parameters of their mathematical model. In case the exact model is unknown, a curve might also be approximated by a polygon on triangulated data. The geometric representation of an *Edge* might be specified by a 3D curve, or a 2D curve in the parameter space of each surface that the *Edge* belongs to.

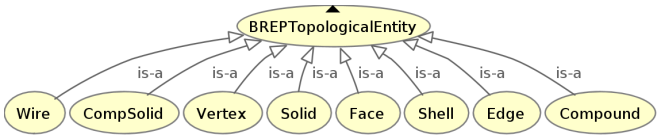


Fig. 3: Taxonomy of topological BREP entities

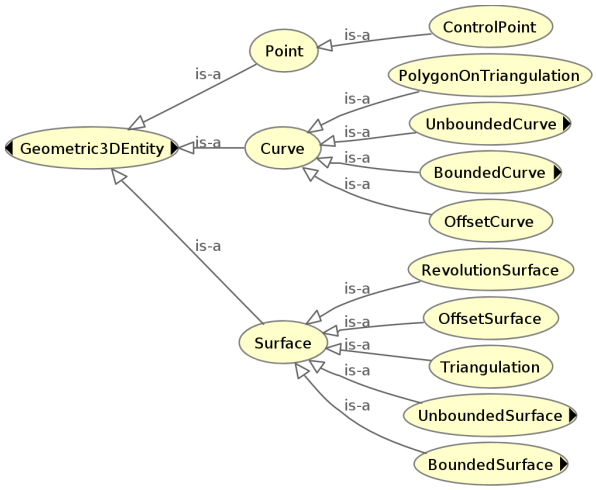


Fig. 4: Upper taxonomy of geometric entities specified in 3D space

Surfaces rely on unbounded mathematical models, e.g., planes, cones, or cylindrical surfaces, and bounded models, e.g., Bezier surfaces, B-spline surfaces, spheres, or toruses. Surfaces can also be defined as linearly extruded curves. An offset surface translates a base surface along a given vector, and a revoluted surface is created by rotating a given base curve around a given direction vector. Again, if the exact mathematical model of a surface is unknown, an approximation based on triangulation might be specified.

IV. CAD ONTOLOGY

The proposed semantic description language for representing CAD models has been implemented using the Web Ontology Language (OWL). OWL is an open web standard developed by the World Wide Web Consortium (W3C) [1] and is based on a formal specification, i.e. a description logic, which facilitates logical inference. OWL distinguishes between classes, individuals (instances of classes), and properties defined for classes or individuals. Classes and properties are arranged in a hierarchical manner and may be derived from multiple super classes and properties respectively. In this section, the semantic BREP representation and the taxonomy of geometric constraints will be explained. OWL listings will be given in Manchester OWL syntax.

A. Boundary Representation of objects

Following the structure of the BREP representation, as explained in Section III, a taxonomy of topological (see Figure 3) and geometric (see Figures 4) entities has been implemented. The topological entities are connected through properties which resemble the relations depicted in Figure 2. The remainder of this section explains details of the representation following illustrative excerpts of an examplary CAD model. Figure 5 shows the ontological representation of a CAD model loaded in the ontology editor Protégé [2]. The selected individual *Solid9* is highlighted by a custom rendering plug-in, which we have implemented to provide model inspection directly in Protégé.
The solid *Solid9* is bounded by shell *Shell9*.

```
Individual: cad:Solid9
  Types:
    cad:Solid
  Facts:
    cad:boundedBy  cad:Shell9
```

Shell9 consists of a set of 41 faces that form a closed volume, which corresponds to the highlighted part of the CAD model in Figure 5.

```
Individual: cad:Shell9
  Types:
    cad:Shell
  Facts:
    cad:contains  cad:Face85,
    cad:contains  cad:Face97,
    ...
    cad:contains  cad:Face74
```

Face97 is one of the faces that are part of *Shell9*. It is *locatedAt* at a certain position with a certain orientation. This pose is given by *TransformationMatrix739*. The face is *representedBy* the geometrical entity *CylindricalSurface34* and *boundedBy Wire99*. The orientation of the face is set as reversed. Setting the orientation to be forward or reversed on the topological level allows for sharing the same geometric entities among multiple topological entities, e.g., when the same surface is part of two solids.

```
Individual: cad:Face97
  Types:
    cad:Face
  Facts:
    cad:representedBy  cad:CylindricalSurface34,
    cad:boundedBy      cad:Wire99,
    cad:locatedAt      cad:TransformationMatrix739,
    cad:isReversed     true
```

The geometric entity *CylindricalSurface34* represents a parameterizable cylindrical surface. The three direction vectors *Vector879*, *Vector880*, and *Vector881* define the cylindrical surface's coordinate system at *Position469*. The cylindrical surface also has a *radius*.

```
Individual: cad:CylindricalSurface34
  Types:
    CylindricalSurface
  Facts:
    cad:directionX   Vector880,
    cad:directionY   Vector881,
    cad:directionZ   Vector879,
    cad:hasPosition  Position469,
    cad:radius       "67.5"^^xsd:double
```

[1] http://www.w3.org/

[2] http://protege.stanford.edu/

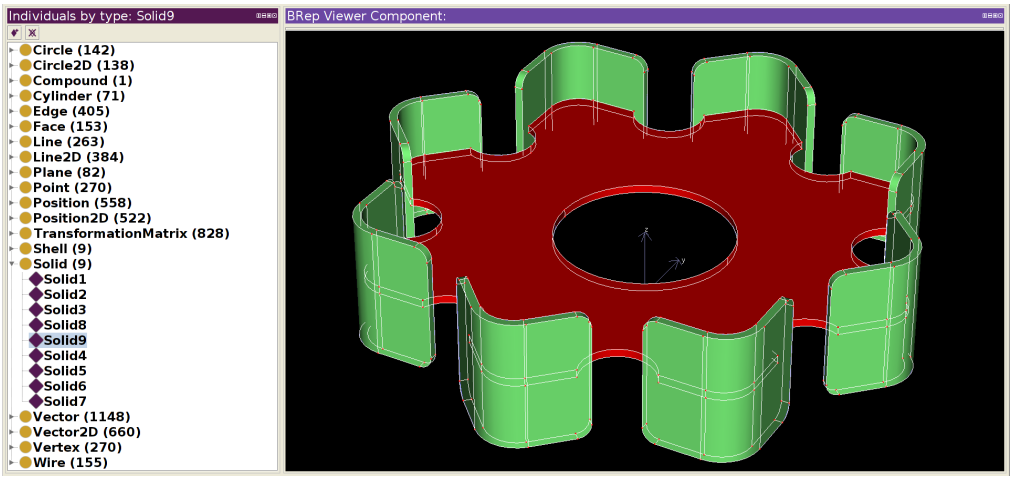


Fig. 5: OWL representation of a CAD model loaded into the ontology editor Protégé. On the left, a list of contained OWL individuals is shown. On the right, a custom-made rendering plug-in is used to visualize the described model. As a specific solid was selected in the list of individuals, the corresponding part of the 3D rendering is highlighted with red color.

Having the unbounded *CylindricalSurface34* as the base geometry for the topological entity *Face97*, a boundary limit on the surface is required and is given through *Wire99*. The wire consists of the four edges *Edge234*, *Edge201*, *Edge147*, and *Edge145*. The *firstElement* of the wire is explicitly specified. Further information on the connectivity of edges is defined in the edge entities.

```
Individual: cad:Wire99
  Types:
    cad:Wire
  Facts:
    cad:firstElement  cad:Edge234,
    cad:contains      cad:Edge234
    cad:contains      cad:Edge201,
    cad:contains      cad:Edge147,
    cad:contains      cad:Edge145,
```

Looking at *Edge234* reveals that it is *representedBy Circle97* and *boundedBy Vertex98* and *Vertex99*. Is has exactly one *adjacentEdge* connected to it. *Edge234*'s pose is given by *TransformationMatrix538*. By asserting all contained edges for wires and all adjacent edges for edges, it is possible to share edges between multiple wires. The specification of *Circle97* is done in a manner analogous to the cylindrical surface.

```
Individual: cad:Edge234
  Types:
    cad:Edge
  Facts:
    cad:representedBy  cad:Circle97,
    cad:boundedBy      cad:Vertex98,
    cad:boundedBy      cad:Vertex99,
    cad:adjacentEdge   cad:Edge147,
    cad:locatedAt      TransformationMatrix538
```

Vertex98 is *representedBy Point52* and has a pose defined by *TransformationMatrix52*. Setting the orientation to be forward or reversed on the vertex level allows sharing the same point among multiple vertices, e.g., when two edges share a common end point.

```
Individual: cad:Vertex98
  Types:
    cad:Vertex
  Facts:
    cad:locatedAt      TransformationMatrix52,
    cad:representedBy  Point52,
    cad:isReversed     false
```

Point52 has asserted x, y, and z coordinates. Vectors are defined in the same way.

```
Individual: cad:Point52
  Types:
    cad:Point
  Facts:
    cad:x  "67.5"^^xsd:double,
    cad:y  "-6.162975822039715E-32"^^xsd:double
    cad:z  "47.0"^^xsd:double,
```

Apart from the explained geometric entities *CylindricalSurface*, *Circle*, and *Point*, there are many more supported types. In Section III most of them have been mentioned. Listing all of their properties is out of the scope of this paper, but they can be investigated in the related ontology file, which is available online [3].

B. Geometric constraints

Given the rich semantic description of CAD models, as explained in Section IV-A, it is possible to refer to arbitrary parts of a CAD model and to link it with additional information. In this paper we want to focus on how to add geometric constraints. Figure 6 shows the upper taxonomy of the geometric constraints that have been designed. Those constraints are meant to be specified between points, curves, and surfaces of objects. We explain the constraints formulation using a couple of examples, which provide a fair understanding of our approach. A complete formal description can be found in the aforementioned ontology file. In our representation a geometric constraint refers to two

[3] https://github.com/OntoBREP/ontobrep

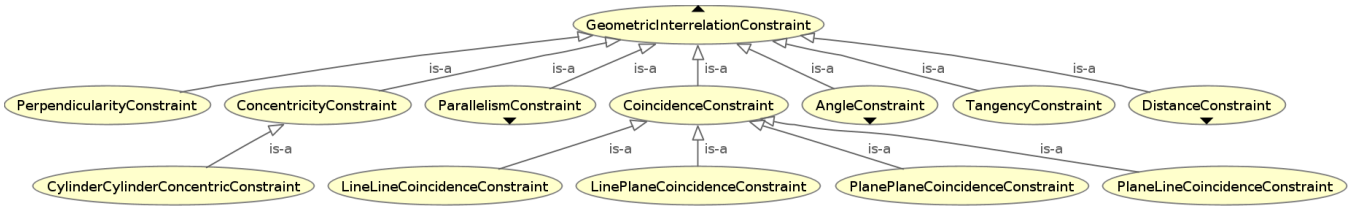


Fig. 6: Visualisation of upper taxonomy of supported geometric interrelation constraints

geometric entities: a base entity with a defined pose and a constrained entity whose pose depends on the fixed entity and the constraint itself. The null-space of a geometric constraint describes a set of relative transformations between the involved geometries that satisfy the constraint.

1) CylinderCylinderConcentricityConstraint: This constraint can be defined between two cylindrical surfaces. It constrains the symmetry axis of the *constrainedGeometry* to align with the symmetry axis of the *baseGeometry*. The resulting null-space of this constraint is specified by a Line, which is identical to the symmetry axis of the *baseGeometry*.

```
Class: CylinderCylinderConcentricConstraint
  SubClassOf:
    ConcentricityConstraint,
    baseGeometry exactly 1 CylindricalSurface,
    constrainedGeometry exactly 1
        CylindricalSurface,
    hasNullSpace exactly 1 Line
```

2) PlanePlaneCoincidenceConstraint: This constraint can be defined between two planar surfaces. It constrains the distance of the *constrainedGeometry* from the *baseGeometry* and also the rotation of the *constrainedGeometry* along the two directions orthogonal to the normal vector of the *baseGeometry*. The resulting null-space of this constraint is specified by a Plane, which is identical to the *baseGeometry*.

```
Class: PlanePlaneCoincidenceConstraint
  SubClassOf:
    CoincidenceConstraint,
    baseGeometry exactly 1 Plane,
    constrainedGeometry exactly 1 Plane
    hasNullSpace exactly 1 Plane
```

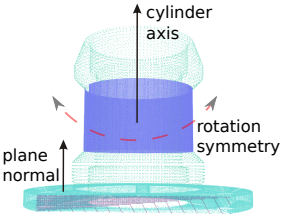
Practical applications of the semantic BREP representation and the geometric constraints are given in Section V.

V. Applications

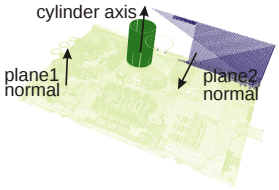
A. Object Recognition

Object recognition and pose estimation using CAD models is a classical research area in the field of computer vision. There are several approaches that rely on object detection models composed of primitive shapes. We extended our previous work [10] to directly obtain primitive shapes from object models that are described using the BREP ontology presented in this paper. This improves the performance by eliminating errors due to incorrect detections of shape primitives from point cloud models.

In [11], we presented an approach for detecting symmetrical objects using geometric constraints. This was extended to use semantic BREP models of primitive shapes and geometric

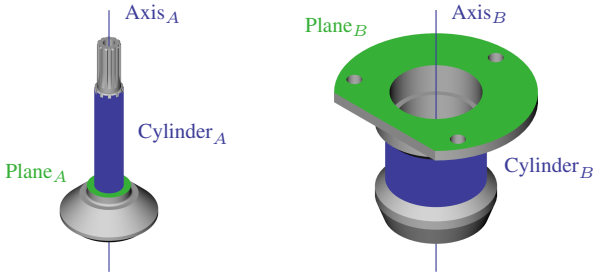


(a) Recognition of underspecified pose for a symmetrical object



(b) Estimation of a completely specified pose using primitive shapes

Fig. 7: Estimation of object poses (fully-specified or underspecified) using primitive shape constraints based on CAD semantics



CylinderCylinderConcentricConstraint(Cylinder$_A$, Cylinder$_B$)
PlanePlaneCoincidentConstraint(Plane$_A$, Plane$_B$)

Fig. 8: Geometric constraints that specify an assembly task involving two objects

constraints between them (see Figure 7). The resulting underspecified object poses, controlled-spaces, and null-spaces can now be represented as geometric entities themselves.

B. Task specification based on geometric constraints

Intuitive interfaces for robot programming [12] are an important target application of this work. Semantically rich descriptions of CAD models allow users to refer not only to manipulation objects, but to the geometric entities that they are comprised of. Robot tasks can then be defined using interrelation constraints between the relevant geometric entities, e.g., to specify an assembly pose [13]. Figure 1 shows a dialog of a graphical user interface used to visualize object models and to define a set of geometric constraints between them. Figure 8 presents a constraint-based definition of an assembly task involving these workpieces, which is parameterized by constraints between the geometrical entities

that compose the workpieces.
Using the object poses obtained from the vision system (see Section V-A), the geometric constraints are solved to generate target poses for task execution.

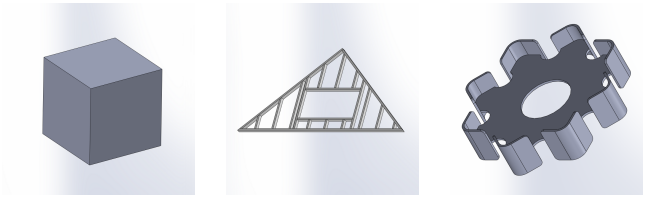
C. Constraint-based robot control

In this application, the robot task specifications that are created using geometric constraints (see Section V-B) are combined with semantic descriptions of the robotic workcell. A semantic workcell description contains the workcell layout, sensors, tools, robots and their kinematic structure, etc. The task's geometric constraints are solved to generate corresponding constraints on the robot's pose. The resulting robot-dependent null-space of the solved geometric constraints can be linked to the task instance. The constrained robot poses are then executed on the low-level robot controller, which may exploit the information about null-spaces to optimize the robot's motion without violating the given constraints. Details of this application can be found in [14].

VI. Quantitative Evaluation

In order to mitigate the required efforts for describing a CAD model using our semantic description language, we implemented a software component for converting STEP and IGES files to the proposed ontological representation. As most widely used CAD software support these interchange formats, our converter enables us to continue relying on these mature and highly specialised software tools for designing the CAD models. The converter has been implemented in Java and uses the OWL API [4] for creating OWL axioms and a Java Native Interface (JNI) to the community edition of Open CASCADE [5] for parsing STEP and IGES files.
We investigated the quantitative aspects of importing an existing STEP or IGES model and creating corresponding ontological entities. This includes the time required to parse the given STEP or IGES files, to initialize the OWL API's manager and factory classes, and to translate the in-memory data structure of the source model to OWL individuals, classes, and object property and data property assertions. The resulting in-memory OWL representation was serialized again to a file using different formats, i.e., the Manchester OWL syntax and the RDF/XML syntax. Figure 9 shows the models used in this evaluation. In Table I, file sizes of the different CAD formats, including the two OWL-based serializations, are compared. As OWL representations feature unique text identifiers for all geometric and topological entities, they are bigger in size. Due to the redundancy in identifiers that share common prefixes, the discrepancy in size, e.g., between STEP and the RDF/XML syntax shrinks from a factor of 7 to 1.7 when both files are compressed. Table II lists the types and counts of topological entities for the models used in this evaluation and also the types and counts of OWL axioms that were created to describe the same objects.

[4] http://owlapi.sourceforge.net/
[5] http://www.opencascade.org/



(a) Cube (b) Frame (c) Rotor

Fig. 9: Models used for the quantitative evaluation

TABLE III: Required time for converting a STEP representation of CAD data to an OWL-based description, and for loading the OWL-based models in the RDF triple store Sesame.

Model	*Converting STEP*	*Loading OWL in Sesame*	
	time in ms	*time in ms*	*axioms per ms*
CUBE	365	25	57.2
FRAME	805	343	77.0
ROTOR	1018	704	60.9

There are various ways to store and query the created OWL-based CAD models. One common option is to use the Sesame [6] triple store. In a second experiment, the OWL representations of models have been loaded into Sesame in order to benchmark loading time. Required times for importing STEP models and loading the OWL representation in Sesame are listed in Table III. All experiments have been carried out on an Intel Xeon quad-core CPU running at 2.80 GHz with 12 GB of RAM and a mechanical harddisk. Listed times have been calculated as an average score based on ten trials each, with only minor deviations among single results, i.e. less then 10 %.

VII. Conclusion

In this paper, we present a semantic description language for CAD models, which is based on a boundary representation of points, curves, surfaces, and volumes. This language also offers features to specify geometric constraints between arbitrary parts of CAD models. They may not only refer to the coordinate frame of an object, but also directly to its vertices, edges, or faces. The benefits of this approach have been showcased through a set of applications, that range from object recognition to intuitive parameterization of task descriptions and constraint-based robot control. A quantitative evaluation has been carried out, in order to assess the complexity of representing complete CAD models in a semantic way. The results of the evaluation show that the approach is not only feasible but practically useful.

Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 287787 in the project SMErobotics.

[6] http://rdf4j.org/

TABLE I: Comparison of file sizes for given CAD models represented in different formats. OWL-based formats are more verbose than the industry standards STEP and IGES, but compressing the models drastically decreases the discrepancy to an acceptable level.

Model	*File Size in kB*									
	BREP		*STEP*		*IGES*		*OWL Manchester*		*OWL RDF/XML*	
	plain	*zipped*	*plain*	*zipped*	*plain*	*zipped*	*plain*	*zipped*	*plain*	*zipped*
CUBE	4.0	0.9	15.9	2.9	21.2	1.6	51.6	3.1	150.0	4.7
FRAME	143.5	15.9	353.7	38.9	444.7	27.4	1010.3	49.6	2764.3	69.6
ROTOR	170.8	20.0	650.8	63.3	896.0	54.6	1636.9	79.6	4455.3	115.8

TABLE II: Number of topological BREP entities for the given CAD models and how they translate into the proposed OWL representation.

Model	*Number of topological BREP entities*									*Number of OWL axioms*							
	Ve[a]	*Ed*[b]	*Fa*[c]	*Wi*[d]	*Sh*[e]	*So*[f]	*CS*[g]	*Co*[h]	*Total*	*C*[i]	*OP*[j]	*DP*[k]	*I*[l]	*CA*[m]	*OPA*[n]	*DPA*[o]	*Total*
CUBE	8	12	6	6	1	1	0	1	35	15	12	17	206	206	281	694	1431
FRAME	152	228	114	114	19	19	0	1	647	16	12	17	3915	3915	5358	13186	26419
ROTOR	270	405	153	155	9	9	0	1	1002	19	12	18	6068	6068	8342	22314	42841

[a]Vertex [b]Edge [c]Face [d]Wire [e]Shell [f]Solid [g]CompSolid [h]Compound

[i]Class [j]Object property [k]Data property [l]Individual [m]Class assertion [n]Object property assertion [o]Data property assertion

REFERENCES

[1] C. Li, C. McMahon, and L. Newnes, "Progress with OntoCAD: A standardised ontological annotation approach to CAD systems," in *International Conference on Product Lifecycle Management (PLM)*, Eindhoven, Netherlands, July 2011.

[2] L. E. R. Garcia, "Ontological CAD data interoperability framework," in *4th International Conference on Advances in Semantic Processing (SEMAPRO)*, Florence, Italy, Oct. 2010.

[3] S. Tessier and Y. Wang, "Ontology-based feature mapping and verification between CAD systems." *Advanced Engineering Informatics*, vol. 27, no. 1, pp. 76–92, 2013.

[4] A. Matsokis and D. Kiritsis, "An ontology-based approach for product lifecycle management," *Computers in Industry*, vol. 61, no. 8, pp. 787–797, Oct. 2010.

[5] M. Waibel, M. Beetz, J. Civera, R. D'Andrea, J. Elfring, D. Gálvez-López, K. Häussermann, R. Janssen, J. Montiel, A. Perzylo, B. Schießle, M. Tenorth, O. Zweigle, and R. van de Molengraft, "Roboearth - a world wide web for robots," *IEEE Robotics and Automation Magazine (Special Issue Towards a WWW for Robots)*, vol. 18, no. 2, pp. 69–82, 2011.

[6] M. Tenorth, A. Perzylo, R. Lafrenz, and M. Beetz, "Representation and exchange of knowledge about actions, objects, and environments in the roboearth framework," *IEEE Transactions on Automation Science and Engineering (T-ASE)*, vol. 10, no. 3, pp. 643–651, 2013.

[7] J. De Schutter, T. De Laet, J. Rutgeerts, W. Decré, R. Smits, E. Aertbeliën, K. Claes, and H. Bruyninckx, "Constraint-based task specification and estimation for sensor-based robot systems in the presence of geometric uncertainty," *The International Journal of Robotics Research*, vol. 26, no. 5, pp. 433–455, 2007.

[8] T. De Laet, S. Bellens, R. Smits, E. Aertbelien, H. Bruyninckx, and J. De Schutter, "Geometric relations between rigid bodies (part 1): Semantics for standardization," *IEEE Robotics Automation Magazine*, vol. 20, no. 1, pp. 84–93, Mar. 2013.

[9] T. De Laet, S. Bellens, H. Bruyninckx, and J. De Schutter, "Geometric relations between rigid bodies (part 2): From semantics to software," *IEEE Robotics Automation Magazine*, vol. 20, no. 2, pp. 91–102, June 2013.

[10] N. Somani, E. Dean-Leon, C. Cai, and A. Knoll, "Scene perception and recognition in industrial environments," in *International Symposium on Visual Computing (ISVC)*, July 2013.

[11] N. Somani, C. Cai, A. Perzylo, M. Rickert, and A. Knoll, "Object recognition using constraints from primitive shape matching," in *International Symposium on Visual Computing (ISVC)*, Dec. 2014.

[12] S. Profanter, A. Perzylo, N. Somani, M. Rickert, and A. Knoll, "Analysis and semantic modeling of modality preferences in industrial human-robot interaction," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept. 2015.

[13] A. Perzylo, N. Somani, S. Profanter, M. Rickert, and A. Knoll, "Toward efficient robot teach-in and semantic process descriptions for small lot sizes," in *Robotics: Science and Systems (RSS), Workshop on Combining AI Reasoning and Cognitive Science with Robotics*, Rome, Italy, July 2015.

[14] N. Somani, A. Gaschler, M. Rickert, A. Perzylo, and A. Knoll, "Constraint-based task programming with CAD semantics: from intuitive specification to real-time control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept. 2015.