# Form-Function Reasoning for Product Shape Ontology

Eric Wang and Yong Se Kim

Creative Design and Intelligent Tutoring Systems (CREDITS) Research Center
Sungkyunkwan University
Suwon, Korea 440-746
ewang@skku.edu, yskim@skku.edu

**Abstract.** We present an ontology of objects, functions, and generic shape representation that supports form-function reasoning. By reasoning from the mechanical and other functions of objects to their geometric shape requirements, we deduce the generic shape representation of objects, which we represent as a partial boundary representation composed of primitive geometric shape elements and their spatial and other relations. We use this ontology to model a knowledge base of everyday objects, including their generic shapes. This ontology can support applications such as product design and object recognition.

**Keywords:** Form-function reasoning, generic shape, object ontology.

## 1 Introduction

There is an emerging interest in automated reasoning support for product design applications, combining knowledge of functions, objects, generic shapes, and their interrelations, in a machine-understandable representation. In this paper, we present an ontology to represent everyday objects intended for interaction with humans or with other objects. An object's usage is achieved through functions conducted by the object's detailed shape. While objects may have many differences at the detailed shape level, their functions could be described at a more generic level, using a common set of generic functions. By reasoning about an object's generic function decomposition, we could deduce its geometric shape requirements. We embed these generic shape representations into the ontology, so that each object model carries its own generic shape data. This ontology supports diverse applications including product design and object recognition.

## 2 Related Works

To support function-based design, a function decomposition method and vocabularies for describing the functions of mechanical components has been developed [9][11]. In this method, the overall function of a product is recursively decomposed into sub-functions until a primitive level is reached, where all functions

have input and output in forms of *energy*, *material*, or *information*. Each primitive sub-function is then mapped to a concept or mechanical component to obtain a product design.

Through more systematic treatment, function-based taxonomies for design [6][10] have been proposed, including assembly-related and control-related functions of mechanical objects. These taxonomies define a hierarchical structure among groups of functions, and propose a generic phrase structure that describes many functions across different domains. However, they lack the completeness of an ontology in defining relations between the phrase elements used in function definitions, and their semantics. For example, Kirschman & Fadel [6] define a sentence form to represent a function's parameters, but forbid certain combinations of keywords from occurring together. Without a way to encode semantic meaning, this knowledge cannot be represented in the taxonomy itself, and must be stated as a meta-level comment.

Kitamura *et al.*'s Function and Behavior Representation Language (FBRL) includes an ontology of functions of artifacts. They use this to model a coffee maker's functionality and intended use, and anticipate unintended user behaviors [12]. They have also applied this ontology to the design of a power plant [7] and manufacturing processes of industrial products [8].

Other works have explored the process of obtaining form from function. Welch & Dixon describe the use of behavior graphs to map the function of subsystems of a part into specific forms [14]. Kim & Feng consider the synthesis of configuration shape of a mechanical part from its functional requirements at the early design stage [5]. Camelo *et al.* [1] proposes a knowledge representation model to support synthesis in design based on four levels of abstraction: purpose function for the designer's intent, action function as an abstraction of behavior, behavior as an abstraction of physical states, and structure as an abstraction of geometry.

## 3 Object Ontology Modeling

We have developed an ontology of objects with a generic shape representation [13], and used this to instantiate models for several dozen everyday objects, with an early emphasis on office furniture. We use UML and Protégé for ontology modeling. We have converted a subset of this ontology (related to chairs) to Jess to support a prototype reasoning application that performs object classification for chairs at a symbolic level.

### 3.1 Function Ontology

We incorporate a function ontology based on existing function-based design research, shown in **Fig. 1**. It is primarily based on Kirschman & Fadel's function taxonomy [6], with some contribution from Stone & Wood [10].

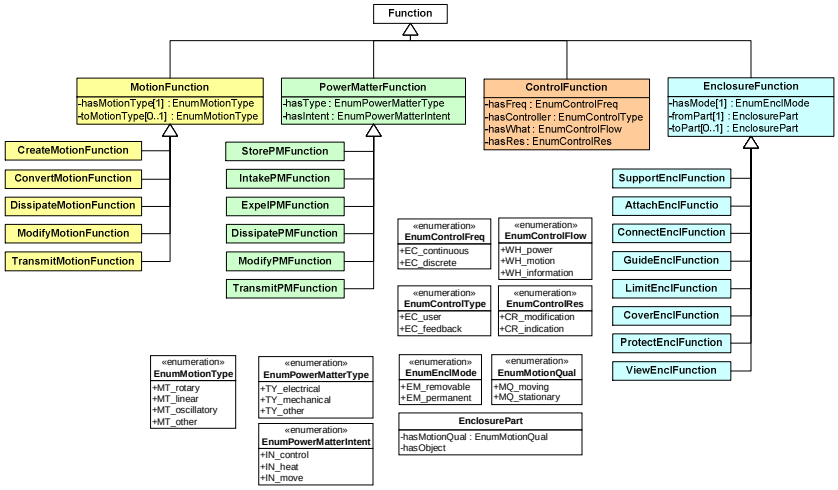


**Fig. 1.** Function ontology (after Kirschman & Fadel 98, Stone & Wood 99)

### 3.2 Part-Whole Representation of Objects and Features

Manufactured objects are typically assembled from multiple components, where each component contributes some specific functionality. We adopt a part-whole representation based on decomposing an object into a set of *features* and their spatial relationships, where a feature is a functionally significant subset of an object or another feature. Each feature is characterized by its intended functions and usage information. This feature-based decomposition can be carried out to any level of detail, although for real objects it necessarily halts at a finite depth.

Some features could themselves be objects if considered separately. Hence, we use a recursive data structure in which Object and Feature both derive from a Descriptor base class, and inherit the same data attributes from it, as shown in **Fig. 2**.

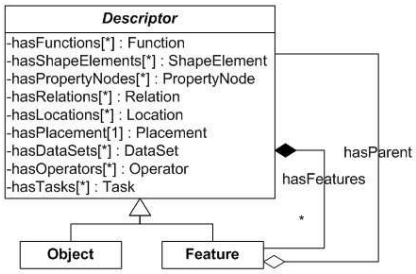


**Fig. 2.** Part-whole representation of Object and Feature classes

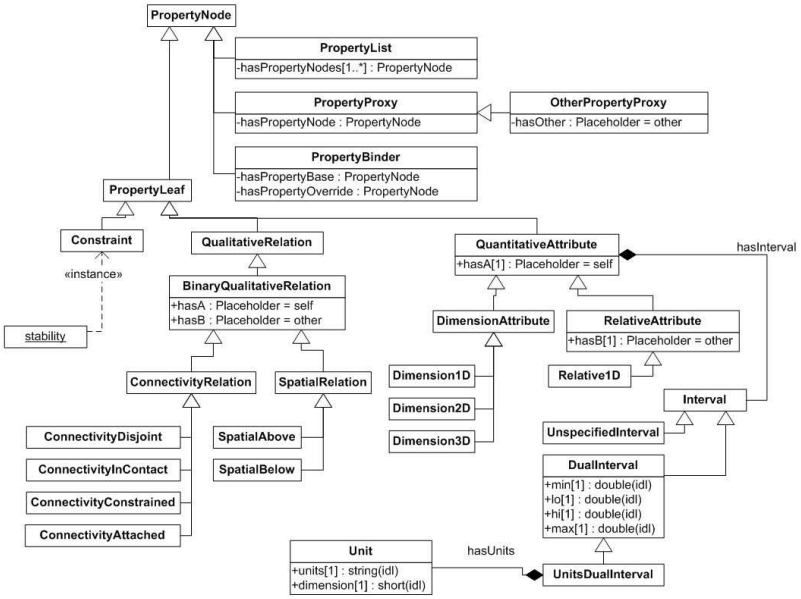


**Fig. 3.** PropertyNode hierarchy for part-whole data attributes

A primary goal of this object ontology is to instantiate a hierarchical knowledge base of object models, representing classes of real everyday objects. This presents us with a challenge in organizing objects' data attributes. We have identified that it is useful to (a) manage a set of properties as if they were a single individual, (b) specify the existence of a property separately from its value, (c) compose sets of properties from other sets of properties, and (d) override property values in other features of the same object, based on the part-whole containment hierarchy within a single object model, rather than on the object-feature class hierarchy in the ontology. The traditional ontological approach of modeling data using properties (binary relations) proves to be too limiting, as it provides attribute inheritance only within the ontology class hierarchy, and doesn't support composition. We reify the notion of *data attribute* as a concrete PropertyNode class, shown in **Fig. 3**, with support for unspecified (deferred) values, and data value overriding. Data value overriding is useful both within the object class hierarchy (e.g. a Table base class may establish the existence of an *area* attribute, but leave its value range unspecified, while each subclass of Table provides its own override), and within the part-whole model of a single object class (e.g. a StandardTable class of typical business desks could impose specific value constraints on the height and angle of its Supporter leg features).

### 3.3 Generic Shape Representation

A key consideration in our object ontology is a generic representation of shape, which can flexibly describe a family of objects. We first model primitive geometric

shape elements as shown in **Fig. 4**; an example of a geometric shape element is *horizontal planar surface.* Each shape element comprises a geometric datum element that specifies the relevant subset of the object or feature's geometry, geometric constraint such as *horizontal* or *planar*, and zero or more modifiers, which provide qualitative (discretized) measures of variations from the nominal constraint.

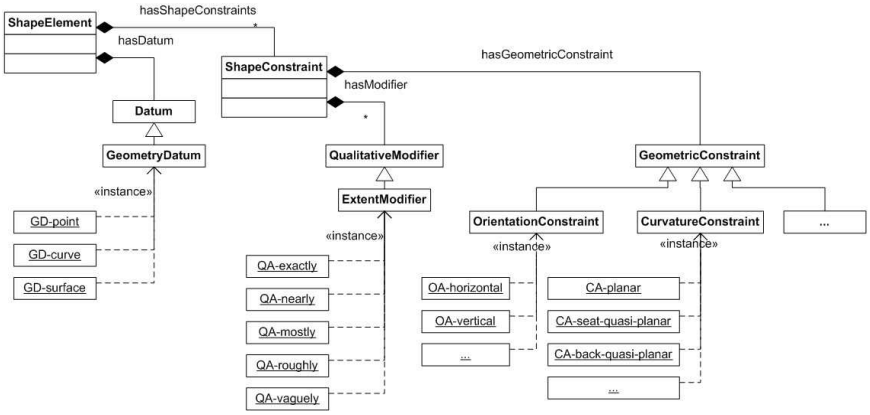


**Fig. 4.** ShapeElement hierarchy for generic shape representation

Shape elements are one of the data attributes associated to every object and feature (as can be seen in **Fig. 2**). We then represent generic shapes of objects by composing shape elements and their spatial and other constraints, including constraints between features of the same object.

To tolerate wide variations in specific geometry, we adopt a partial boundary representation (B-rep) interpretation, in which only the relevant subset of an object or feature's boundary is fully specified, representing the critical geometric and topological relations only. That is, the set of all of a feature's shape elements are together taken to comprise a partial B-rep. Unspecified portions of the boundary are abstracted away. In their place, we provide a generalized bounding volume, e.g. bounding box or sphere, to enforce the principle that all real solid objects are bounded.

## 4 Form-Function Reasoning

We apply form-function reasoning, from the functions of objects to their generic shapes, to deduce the functional elements, called *organs* [3], which are the active elements that carry the functions of the features, and their geometric shape requirements, as well as any geometric relations and constraints that exist between features. The result of form-function reasoning is a set of geometric shape elements, which describe the minimum necessary elements for an object or feature to achieve the desired function.

This is a complex kind of reasoning, involving many different reasoning techniques, and intelligent (human-like) understanding and insight. Currently, the authors perform this reasoning manually, and only the results thereof are embedded into the object ontology. In this section, we present three cases of function-to-shape reasoning, and document the techniques used.

### 4.1 Geometric Concepts

We make use of the following geometric notions.

**Gravity.** All objects are affected by the force of gravity. Over typical distance scales, we assume that gravity exerts a vertical downward force everywhere.

**Static behavior of objects.** We will consider objects whose intended usage is mostly static, i.e. they do not change their shape over time in the course of normal usage. We ignore specialized or transient physical effects such as acceleration, friction, texture, surface tension, vibrations, etc.

**Degrees of freedom (DOF).** All motions in 3D space can be characterized by 3 translational and 3 rotational DOFs. We consider two kinds of restrictions on each DOF. A *half-open* restriction limits a DOF to a half-open interval. That is, it blocks motion along one half-axis, but doesn't restrict motion in the opposite direction. (This applies even to rotational DOFs, e.g. in the case of a ratchet component.) A *finite* restriction limits a DOF to a finite closed interval, i.e. it blocks motion in both directions along a single axis.

**Polar set.** The polar set [1] of a 3D point set P is another point set Q such that for all pairs of points $p \in P$ and $q \in Q$, $(p \cdot q) \leq 1$. Since the dot product operation is symmetric, it follows that P is also the polar set of Q, i.e. P and Q are geometric duals of each other. This provides a natural way to convert from a direction of motion to a point set that would block that motion, and vice versa.

**Normal cone and accessibility cone.** The normal cone of an object is the convex hull of the normal vectors of its faces [4]. The accessibility cone is then defined as the polar dual of the normal cone. This succinctly characterizes the set of accessibility directions of an object.

**Interaction with human.** Objects, particularly furniture, that are intended to contact or interact closely with a human, require sufficient clearance for the human's body parts, as well as accessibility directions that allow the human to approach and leave. We model clearance as negative (empty) volumes bounded by the faces of one or more objects, and represent accessibility directions using accessibility cones derived from sets of relevant faces of the object. Modeling of the human's body itself is currently handled implicitly on a case-by-case basis, rather than explicitly representing human body shape in the ontology.

### 4.2 Form-Function Reasoning for Container

We consider a generic Container class that contains liquids. We elaborate its functions as follows.

**Limit all motions in the lower halfspace.** Freely flowing liquid moves in every direction that has any downward component and any horizontal component. Hence, a Container must limit all such motions simultaneously. More specifically, it suffices to limit vertical downward motion to a half-open interval, but all horizontal motions orthogonal to gravity must be limited to finite intervals. The set of limited motions thus comprises the normal cone of a *pocket* form feature [4], oriented upward with respect to gravity, as shown in **Fig. 5**. Hence, we deduce a geometric shape requirement of an *upward pocket* with respect to gravity.

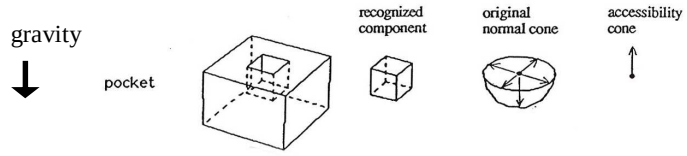


**Fig. 5.** Upward pocket feature

**Contain a liquid.** The material properties of a liquid are that its molecules flow freely, but maintain a constant volume. It follows that a liquid can escape by flowing through a hole of any size. Hence, we deduce a geometric shape requirement of *no through holes*.

Hence, we deduce that a Container must be an upward pocket without holes. By observation, the converse also holds: *any* upward pocket without holes can function as a Container of liquids. This can be observed after any rainfall by seeing rain water collecting in every depression in rocks and other surfaces.

### 4.3 Form-Function Reasoning for Table

We identify a generic table's function as follows: to support multiple general solid objects without motion, at some constant elevation (height above the ground), so as to make them conveniently accessible to a human. We decompose the Table class into two feature types, (1) a Counter feature that contacts the objects, and (2) one or more Supporter features that fulfill the role of maintaining the Counter's constant elevation.

Focusing our attention on the Counter, we elaborate its functions as follows:

**Contact multiple general solid objects.** To support another object without motion implies that the supported object is stable. Hence, to stably support many objects of arbitrary shapes, with arbitrary positioning, implies a multitude of contact points. From this, we deduce a geometric requirement of a *surface*. Note that this is a fairly universal line of reasoning, which applies to most contact relations between solids. It is known that most mechanical assemblies and contact-related functions are characterized by their mating surfaces.

**Limit vertical downward motion.** Gravity induces a vertical downward force on all objects; hence, the Counter feature must limit the resulting motion to a half-open interval. The polar dual of a vertical downward vector is a horizontal planar halfspace that faces upward with respect to gravity. From this, we deduce a geometric requirement of *planarity*, i.e. a planar surface.

While the polar dual technique also suggests a horizontalness property, this alone isn't sufficient to establish it. For example, the union of many small Counters at different elevations could still satisfy both of the above functions, but isn't a typical table. We introduce additional factors to rule out this possibility.

**Minimize forces and energy.** A secondary characteristic of a table's usage is to minimize the forces acting on its supported objects. This rules out the case of an inclined Counter, as this would cause objects to tend to slide off (ignoring friction). Also, a table should minimize the energy cost of repositioning objects on it, which argues against having an inclination, or many sub-Counters with different elevations. From these considerations, we deduce stronger support for a geometric requirement of *horizontalness*.

**Accessible from upper halfspace.** The purpose of a table is to make objects conveniently accessible to a human, i.e. graspable at a moment's notice. This implies that a Counter's elevation shall be attuned to the human's expected posture, and its area shall be appropriate for a human's arm's reach. It also implies that an object on the table should be accessible from any direction in the upper halfspace induced by the Counter's top surface, with respect to the gravity direction. This further supports having a uniform elevation everywhere on the Counter, which also supports the geometric requirement of *horizontalness.*

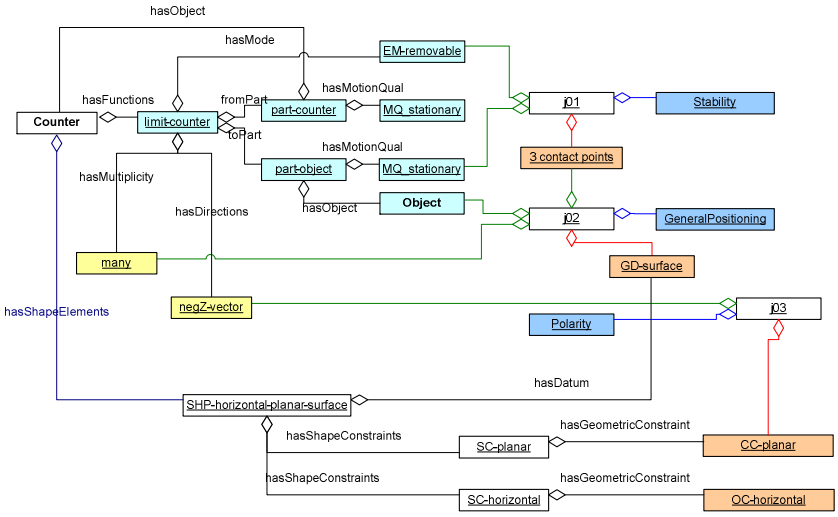


**Fig. 6.** Trace of form-function reasoning for a Table's Counter feature

Hence, from the functions of the Counter feature, we deduce a geometric shape element of a *horizontal planar surface*. This agrees with the intuitive notion of a countertop or tabletop. We explicitly represent the above chain of reasoning in our ontology, as shown in **Fig. 6**.

### 4.4 Form-Function Reasoning for Chair

A chair's primary function is to support one human in a seated posture, at a constant elevation. We decompose the Chair class into two feature types, (1) a Seat feature that contacts the human, and (2) one or more Supporter features that maintain the Seat's constant elevation. We note that a chair's Seat feature has similar functions as a table's Counter feature, namely to contact another object, and to limit its vertical downward motion. Hence, by applying similar reasoning, the geometric shape element of a *horizontal planar surface* can also be applicable to a Seat. By observation, this is, indeed, a valid shape for some real chairs.

However, since the Seat is meant to directly contact a human, additional issues such as ergonomics must be taken into account.

**Ergonomics.** When a rigid object is intended to contact a human for an extended period of time, the human's comfort becomes a significant consideration. One solution is to add padding to soften the contact, but this entails some shape deformation during usage (in fact, this deformation is precisely the function of the padding!), which we do not yet model in our ontology. An alternative mechanism that maintains rigidity is to contour the surface to better fit the intended body part. Hence, a Seat could have various non-planar deviations, so long as it remains *approximately* planar to fulfill its primary function.

Thus, the *horizontal planar* property is taken not as a firm requirement, but as one allowed extreme within some range of variations. We abstract away these variations in a seat's shape by defining a qualitative condition of *seat_quasi_planar*, which spans a range of surface curvatures from perfectly planar to contoured so as to fit a human's bottom. At this level of abstraction, we do not commit to any analytic characterization of such contouring

#### 4.4.1 Form-Function Reasoning for BackedChair

As the Chair superclass represents all possible chairs, it does not commit to any other, more specialized, features. Such commitments are deferred to the numerous subclasses of Chair in our object ontology. BackedChair is a subclass of Chair that includes a Back feature that also contacts the human, whose function is to limit the human's reclining motion (rotation of the torso about the hip joint). The human contact function is similar to that of a Seat, so we deduce an analogous geometric shape requirement of a *quasi-planar surface*, which may be contoured to fit a human's back. We expand its other functions as follows:

**Limit approximately horizontal motion.** A human's reclining motion can be decomposed into a rotational force, or torque, around the human's hip joint. As the human's torso is initially upright, the tangential component of this torque is approximately horizontal. Hence, a Back feature must limit an approximately horizontal motion. From the polar set technique, we deduce an *approximately vertical* halfspace. While some real chairs do exhibit a perfectly vertical back, ergonomic considerations allow for a slight inclination for added comfort, such that the external dihedral angle between the seat and the back is slightly greater than 90°.

**Accessibility.** We deduce that the key characteristic of a BackedChair is that the seat and the back together shall define a *step* form feature [4], represented by an

accessibility cone consisting of a 2D sector spanned by the normal vectors of the seat's top face and the back's inner face.

**Geometric variations.** Note that the Seat and Back features can be disjoint, and may be separated by horizontal or vertical gaps, without violating the step form feature requirement. We characterize these allowable variations in geometry by two numeric parameters, *extent* and *separation*. The *extent* is the length of the usable portion of the seat's top face, i.e. in the positive halfspace induced by the back's inner face. The extent must be within a finite interval – if it is too short, then the seat cannot function as a seat, and if it is too long, then the back no longer functions as a back. The *separation* is the horizontal gap, if any, between the seat and the back, and this must be below a threshold value, else the object can't function as a backed chair.

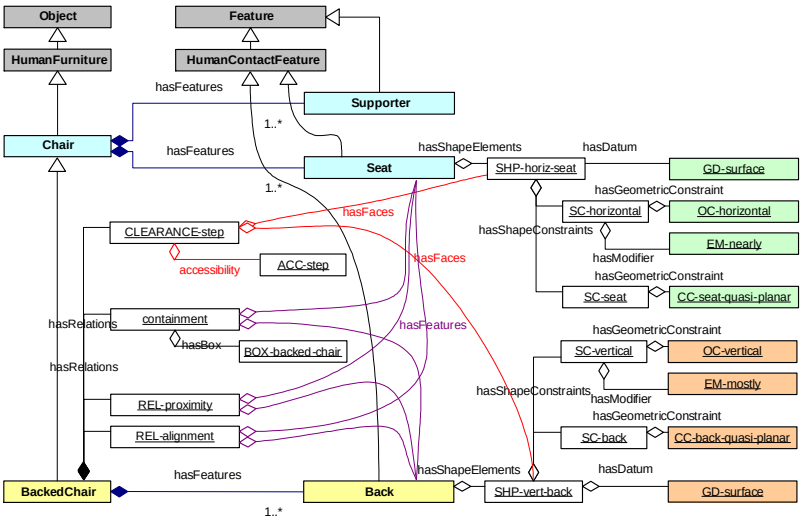


**Fig. 7.** Shape model for BackedChair

Based on the above reasoning, we instantiate a shape model for the BackedChair subclass, shown in **Fig. 7**. The Chair superclass (in light blue) defines a Seat feature having a shape element of *horizontal planar surface*. The BackedChair subclass (in yellow) inherits the Seat feature, and adds a Back feature with a shape element of *vertical planar surface*.

## 5 Discussion and Future Work

We have developed an object ontology that includes a generic representation of shape, and generic functions of objects from established function taxonomies. Our ontology supports a kind of form-function reasoning, where we first identify the key functions that characterize an everyday object, identify attributes and values that parameterize these functions, and then deduce geometric shape elements implied by

these functions. We have found that this reasoning process is complex and challenging, and draws on a vast array of different knowledge sources: physics, mechanics, material properties, ergonomics, etc. In this paper, we have presented several techniques that we have found useful in performing form-function reasoning., but our list is by no means complete. Presently, the form-function reasoning can't be automated, so we perform it manually, relying on our human expertise. The outcomes of this reasoning is represented in our ontology, both in the form of explicit justification graphs, and in the set of geometric shape requirements for a given set of functions, which are encoded into the relevant object class in the ontology.

This ontology could support designers by serving as a library of cases that link functions to forms. For a given parameterized function that matches a known case, it could quickly return the associated shape information. For a given function or feature of an object, it could enumerate successful previous designs that achieved those functions or incorporated those features, to broaden a human designer's horizons.

A major future extension of this ontology is to support automated form-function reasoning. This generally entails that we extend the ontology to represent knowledge sources, means of justifications, and proof steps, and combine it with a reasoning engine, possibly using an approach similar to theorem-proving. In addition, it would require a substantial knowledge base that covers a wide range of "common-sense" knowledge.

## References


[1] Camelo, D., Mulet, E., and Vidal, R., "Function and Behaviour Representation for Supporting Flexible Exploration and Generation in a Functional Model for Conceptual Design", *Proc. Int'l. Conf. on Engineering Design (ICED)*, Paris, Aug. 2007.

[2] Grünbaum, B., *Convex Polytopes*, John Wiley & Sons, Ltd., 1967.

[3] Haudrum, J., *Creating the Basis for Process Selection in the Design Stage*, Ph.D. Thesis, Institute of Manufacturing Engineering, Technical University of Denmark, 1994.

[4] Kim, Y. S., "Recognition of Form Features Using Convex Decomposition", *Computer-Aided Design*, Vol. 24, No. 9, pp. 461–476, Sep. 1992.

[5] Kim, Y. S., and Feng, S. C., "Case Studies to Understand the Relations among Function, Form and Manufacturing Process for Integration of Process Planning into Early Design Stage", *Proc. ASME Conf. on Computers and Information in Engineering*, DETC99/CIE-9121, Las Vegas, 1999.

[6] Kirschman, C. F. and Fadel, G. M., "Classifying Functions for Mechanical Design", *Journal of Mechanical Design*, Vol. 120, pp. 475–482, 1998.

[7] Kitamura, Y. and Mizoguchi, R., "Towards redesign based on ontologies of functional concepts and redesign strategies", Proc. 2nd Int'l Workshop on Strategic Knowledge and Concept Formation, pp. 181–192, 1999.

[8] Kitamura, Y. and Mizoguchi, R., "Ontology–based Systematization of Functional Knowledge", *Journal of Engineering Design*, **15**(4), 327–352, 2004.

[9] Pahl, G., and Beitz, W., *Engineering Design*, Design Council, London, 1988.

[10] Stone, R. B., and Wood, K. L., "Development of a Functional Basis for Design", *Proc. ASME Conf. on Design Theory and Methodology*, Las Vegas, 1999.

[11] Ullman, D. G., *The Mechanical Design Process*, McGraw Hill, 1992.

[12] van der Vegte, W., Kitamura, N., Koji, Y., and Mizoguchi, R., "Coping with Unintended Behavior of Users and Products: Ontological Modelling of Product Functionality and Use", *Proc. ASME Conf. on Computers and Information in Engineering*, DETC04-57720, Salt Lake, 2004.

[13] Wang, E., Kim, Y. S., and Kim, S. A., "An Object Ontology Using Form-Function Reasoning to Support Robot Context Understanding", *Computer-Aided Design & Applications*, **2**(1-4), 2005.

[14] Welch, R., and Dixon, J., "Representing functions, behavior and structure during conceptual design", *Proc. ASME Conf. on Design Theory and Methodology*, Scottsdale, Sep. 1992.