Building a Physical Feature Database for Integrated Modeling in Design

Takashi Kiriyama

Research into Artifacts, Center for Engineering
The University of Tokyo
Hongo 7-3-1, Bunkyo-ku, Tokyo 113, Japan
kiriyama@zzz.pe.u-tokyo.ac.jp

Tetsuo Tomiyama
Hiroyuki Yoshikawa

Department of Precision Machinery Engineering
Faculty of Engineering, The University of Tokyo
Hongo 7-3-1, Bunkyo-ku, Tokyo 113, Japan

Abstract

Building a large-scale knowledge base of engineering common sense is indispensable for the development of intelligent CAD systems. At The University of Tokyo, we have started a project to build a knowledge base of physical features in the domain of mechanical design. A physical feature is a qualitative representation of a physical phenomenon and related attributes. The physical feature database is intended to be used for model building, automatic model generation, consistency management, and qualitative behavioral reasoning. This paper presents the knowledge representation scheme for the physical feature database, the current state of development, and future research directions.

1 Introduction

Developing intelligent CAD systems is crucial within a future computer integrated manufacturing environment. There are already a considerable number of results, and most of them put an emphasis on the use of AI techniques to incorporate domain knowledge and intelligence that are missing from conventional CAD. Despite these efforts, there seems to exist no such truly intelligent CAD. One of the most persuasive explanations is that design knowledge is too huge and complex to be organized and dealt with by existing knowledge representation techniques.

To construct intelligent systems, the approach of large-scale knowledge bases is considered helpful. The Cyc Project [10] conducted at MCC and How Things Work Project [4] at Stanford University are its examples. These projects aim at building powerful AI systems by collecting a large number of knowledge chunks from common sense knowledge to domain specific knowledge, and by providing mechanisms for reusing and sharing the collected knowledge [13]. Intelligent CAD systems require such a large knowledge base containing design knowledge. At The University of Tokyo, we have started a project to build a large-scale database of physical features. A physical feature is a qualitative representation of physical phenomena and related attributes. The physical feature database is intended to be used for model building, model generation, model integration, and model-based reasoning in intelligent CAD.

In the rest of this paper, we present the fundamental idea for dealing with design object models and the knowledge representation scheme for the physical feature database. It also discusses the current state of development and future directions of research. Chapter 2 illustrates the use of the physical feature database for integrated design object modeling in intelligent CAD. Chapter 3 presents the representation scheme for the physical feature database. Chapter 4 discusses the current state of implementation and future research directions, and Chapter 5 concludes this paper.

2 Integrated Design Object Representation

2.1 The metamodel mechanism

A design object can be modeled in respect of various aspects such as geometry, kinematics, dynamics, materials, and assembly. Within one aspect there are abstraction levels of representation varying from purely qualitative to completely quantitative depending on the purpose of modeling. And representations of aspect models base

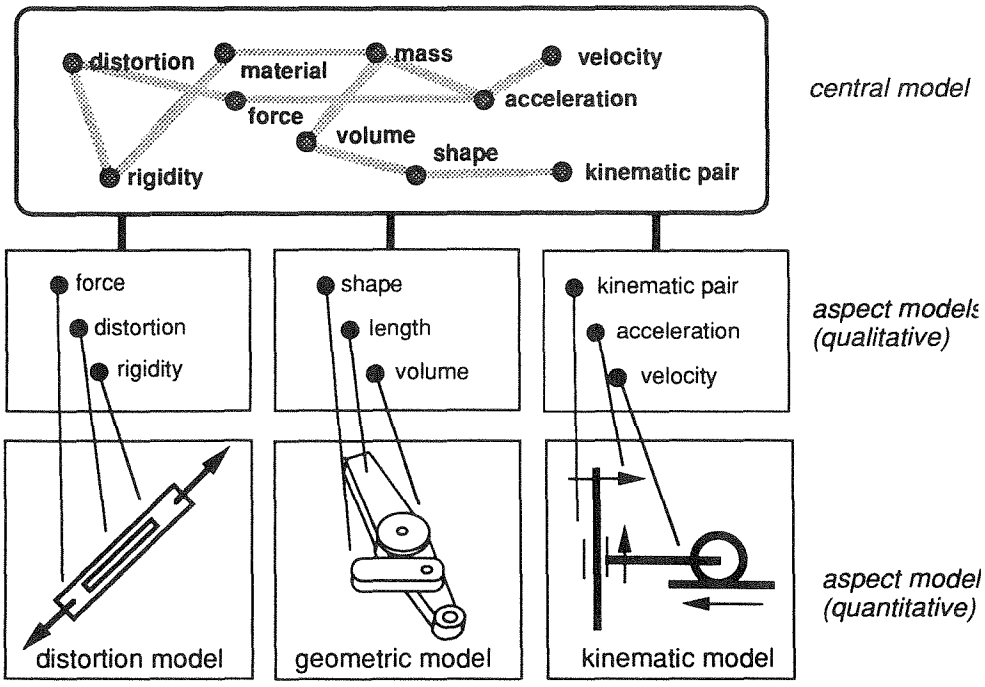

Figure 1: Integration of aspect models by a metamodel

on various ontologies. The designer chooses a suitable representation in accordance with a need that arise in a design process. In a conventional mechanical CAD system, however, models for analysis are centralized in a geometric model in an *ad hoc* manner. They are generated from a geometric model by putting additional information about conditions for analyses. It prevents a system from representing properties that do not match the data structure of a geometric model.

Intelligent CAD is expected to integrate design object models varying over aspects, abstraction levels, and ontologies. We proposed the metamodel mechanism as a framework of design object representation in intelligent CAD [9]. The key idea of the metamodel mechanism is the use of a qualitative central model called a metamodel to represent relationships among aspect models. Data in aspect models are symbolically represented their meanings by uniformly defined concepts, and a metamodel represents dependencies among the concepts [1]. Figure 1 depicts a metamodel and aspect models in the metamodel mechanism. If an aspect model is modified, the change is reported to the metamodel mechanism, which is then used to qualitatively reason out the new behavior of the design object and update related aspect models. If a value of a quantitative aspect model is changed, the change is propagated to the relevant aspect models

[1] *Metamodel* is used twofold; *the metamodel mechanism* denotes the framework of design object modeling, whereas *a metamodel* is a qualitative model of dependency among aspect models.

through the dependency network of the metamodel.

A metamodel can be regarded as a mental model [7] of the design object. Although the designer evaluates the design object concerning one of the aspects at a time, he must always take their dependency into consideration. This can be achieved by using knowledge about dependency among attributes and physical phenomena objects, which are not necessarily mathematically strict equations but naive knowledge. A metamodel qualitatively models conceptual relationships among properties represented by aspect models. In this sense, it represents a designer's mental model of the design object.

2.2 Physical features

In order to achieve model integration by the metamodel mechanism, intelligent CAD must have a knowledge base with which behaviors of the design object is reasoned out. In qualitative physics, various reasoning techniques that can be used for the metamodel mechanism have been developed. On the other hand, it is not well understood how to build a knowledge base containing a large amount of concepts about physical phenomena that appear in engineering design. Thus, the aim of the research is to explore a methodology for building a large-scale knowledge base for design. We call a piece of knowledge of the database a *physical feature* [15]. A physical feature is a representation of a physical phenomenon and related attributes. For instance, a wedge physical feature represents magnification of a force with a wedge-shaped object. Figure 2 illustrates examples of physical features.

The physical feature database plays two roles in intelligent CAD. Firstly, it provides the designer with a library of physical phenomena to build a qualitative behavioral model in conceptual design. The designer models the desired behavior of the design object using physical features as building blocks. We call the qualitative model a primary model. Secondly, it provides the metamodel mechanism with knowledge for handling aspect models. From a primary model built by the designer, the metamodel mechanism generates a metamodel. In generating a metamodel, prerequisites for each physical feature are checked against the primary model to derive all instances of physical features that may arise in the design object. Thus a metamodel includes all potentially possible physical phenomena. The metamodel is used to generate qualitative aspect models by selecting physical phenomena relevant to the aspects of modeling. In an aspect model, state transitions are derived by envisioning techniques.

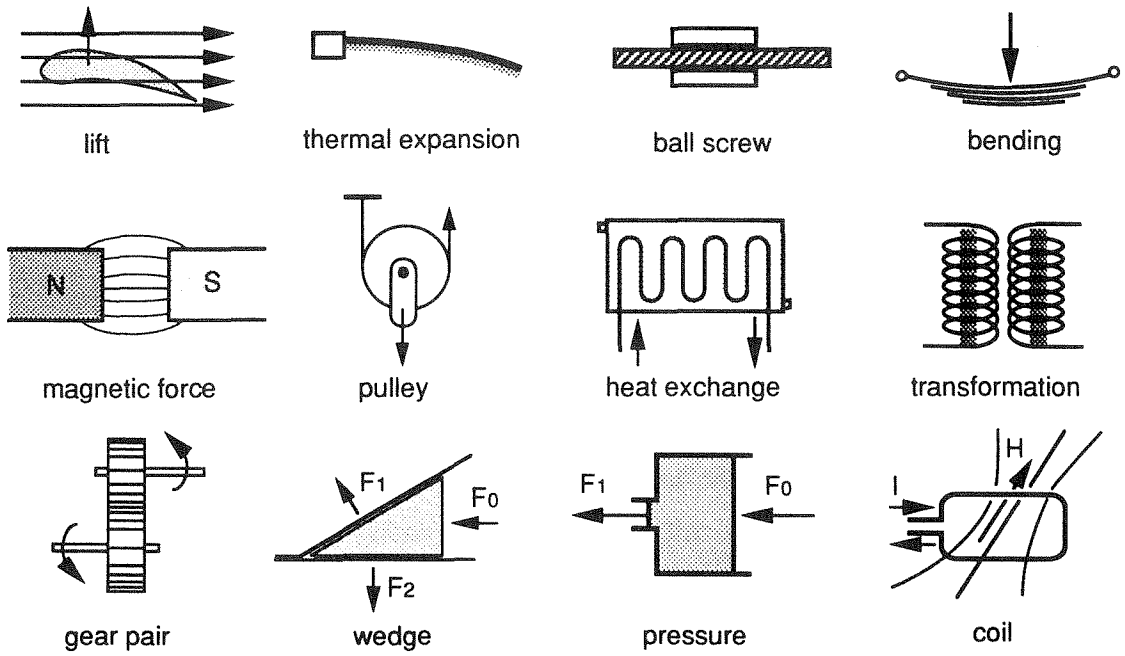

Figure 2: Physical features

3 The Physical Feature Database

3.1 The framework

As the basis of modeling the physical world, we employ the framework of Qualitative Process Theory (QPT) [3]. In QPT, there are three categories, *i.e.*, individuals, views, and processes. Instead of individuals of QPT, we use **objects** that have structures, attributes, and internal states. We use views in the same sense in QPT. A **view** is an abstraction of an object or a collection of objects from a specific viewpoint. It defines quantities for describing qualitative states of the objects. Through a set of views, the design object is modeled as an aspect model. In QPT, a precondition of a view is a condition determined by the external scope of the aspect the view belongs to, and the meaning of the precondition is not defined within the aspect. Instead of preconditions, we use **relations** to represent relationships among objects, so the meanings of them are hierarchically defined in the physical feature database. A **physical feature** is represented by an extended scheme for processes. Processes of QPT influence only on quantities, whereas physical features can influence on internal states and existences of objects.

The physical feature database is implemented in Smalltalk. As the physical feature

database incorporates new kinds of knowledge, its ontology for knowledge representation needs to be extended. For instance, if we adopt an ontology such that an existing object never disappears (which is an appropriate assumption in dynamics), we may confront a difficulty in representing phenomena like evaporation. It means we must adopt an extendible database scheme. Since the advantage of incremental programming of object-oriented languages serves for this purpose, we use Smalltalk to implement the physical feature database.

3.2 Objects

Objects are organized in an abstract-concrete hierarchy. The hierarchy has multiple abstract-concrete relationships, so that objects can be categorized in more than one ways. Object has two types, *i.e.*, entity and stuff. An entity has mass and a boundary, such as a gear, a spring, a bolt, a bearing, and a shaft. Stuff is material of which an entity is made, such as water, oil, metal, and plastics. An entity has parts and structure as attributes. Parts are elements of which the entity consists, and structure represents how the parts are combined. Entities can have internal states, such as on or off for a switch.

In addition to inheritance, delegation [11] gives an object properties of other objects. In designing with physical features, delegation is used to combine existing objects into a new objects. The new object is treated as subclasses of the delegating objects. For instance, one can make a box to be delegated by an electric-conductor, so that the box can become an electric path. The properties of a delegated object is a union of those of the original objects, and an abstract property is eliminated by a concrete property. Thus, by making components delegate their subclasses, the designer can refine the structural model of the design object.

3.3 Views

A view represents how an object can be modeled at an abstract level. A view creates quantities relevant to the viewpoint of abstraction. For instance, an electric coil is modeled with a conductor view, which creates a conductivity for the conductor. A view has conditions for quantities to be satisfied when it is used. For instance, water can be modeled by a solid view when its temperature is below freezing point. When its temperature is above freezing point, it is modeled by a liquid view or a gas view.

The general scheme of views has the slots below;

- name of the view,
- abstract views,
- prerequisites for objects and other views,
- prerequisites for relations among objects,
- prerequisites for quantities,
- quantities created by the view, and
- functional relations between quantities.

Context-dependent Behaviors (CDBs) proposed by Nayak and Joskowicz [12] also provide representations of devices on specific aspects. CDBs are automatically selected to generate an appropriate model for explaining the given question. In the metamodel mechanism, on the other hand, an aspect is defined by a collection of views and physical features, and an aspect model is generated by selecting the views and physical features relevant to the aspect.

3.4 Physical features

A physical feature is described by the slots below;

- name of the physical feature,
- abstract physical features,
- prerequisites for views and other physical features,
- prerequisites for relations among objects,
- prerequisites for relations among physical features,
- prerequisites for quantities,
- quantities created by the physical feature,
- functional relations between quantities, and
- influences on quantities.

A prerequisite condition for relations among physical features is used to avoid correlating irrelevant physical features. For instance, as illustrated in Figure 3, the physical feature amplification must not be instantiated by assuming an interaction between the emitter current of the left transistor and the base current of the right. This can be avoided by using a prerequisite condition for the emitter current and the base current of a physical feature to be of the same transistor.

Additional slots can be used to extend the general scheme for physical features so that influences on states of objects can be described. They include;

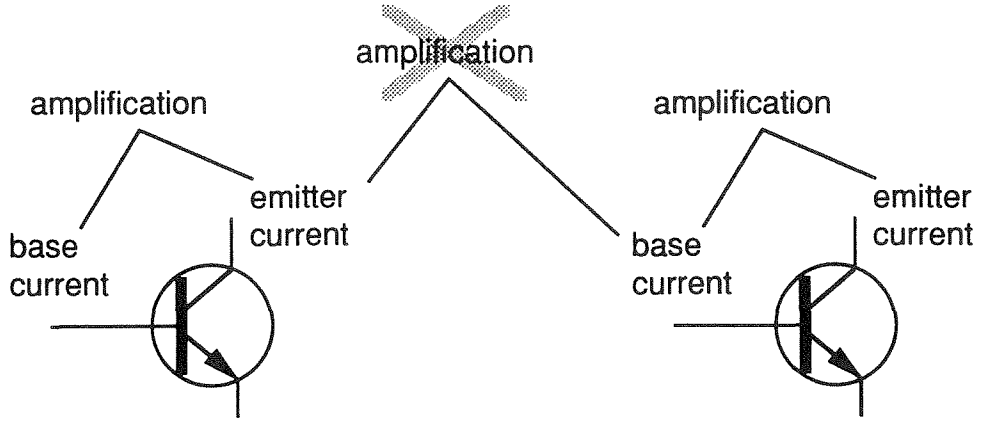

Figure 3: An amplification physical feature

- influences on objects to change their states,
- influences to generate new objects, and
- influences to make objects disappear.

3.5 Relations

A relation represents relationships among objects such as on, above, below, support, and connection. A relation has assertions which are added to the world when it is instantiated. For instance, a connection between entities A and B asserts

connection(A, B), connection(B, A).

Relations are hierarchically defined, and an abstract relation is implied by a specific relation. For example, if electric-connection is a subclass of connection, an electric-connection between A and B also implies a connection between them.

Figure 4 summarizes the conceptual hierarchy in the physical feature database.

4 Building the Physical Feature Database

4.1 Preliminary research

We have started a project to build a large-scale physical feature database of engineering knowledge. Hayes roughly estimated the number of tokens of human knowledge about the physical world as approximately 10^4 to 10^5 [5]. The Cyc project sets its goal at collecting entries of the order of 10^6 [10]. We believe at least about ten thousand

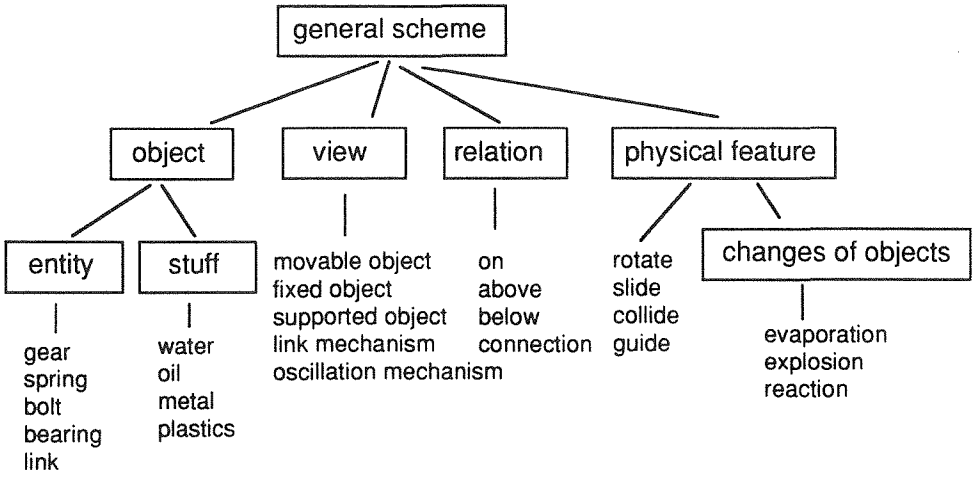

Figure 4: Hierarchy in the physical feature database

objects, views, relations, and physical features are necessary to evaluate the usefulness of the database.

In the preliminary research, we chose the domains of kinematics, robotics, and classical physics as the knowledge sources, (i) because knowledge in these domains was essential for mechanical engineering, (ii) because domain theories were well established, and therefore (iii) because we could obtain systematic description of domain knowledge from textbooks (e.g. [6, 14]). Examples of physical features in kinematics include guide, slide, fix, and release. Physical features in robotics are, for instance, open, close, put, and hold. Physical features in classical physics represent more fundamental physical laws such as Newston's laws, Kirchhoff's law, and Faraday's law.

From these domains, we collected about two thousand objects and some hundred physical features. The objects were mostly kinematic components, since in kinematics a large number of mechanisms were compiled as design handbooks and therefore it was relatively easy to codify knowledge about objects. On the other hand, collecting physical features required more effort than doing objects. Unlike mechanical components, physical phenomena used for mechanical design were not specialized and separated from that commonly seen in everyday situations. The difficulty led us to additionally collect naive knowledge about common physical phenomena. Textbooks of physics and engineering were not helpful to do it, since they described theories on the basis of shared common sense. We surveyed school textbooks of sciences to list common physical phenomena and collected about two hundred physical features from them.

As we collected various physical features, it became necessary to extend the rep-

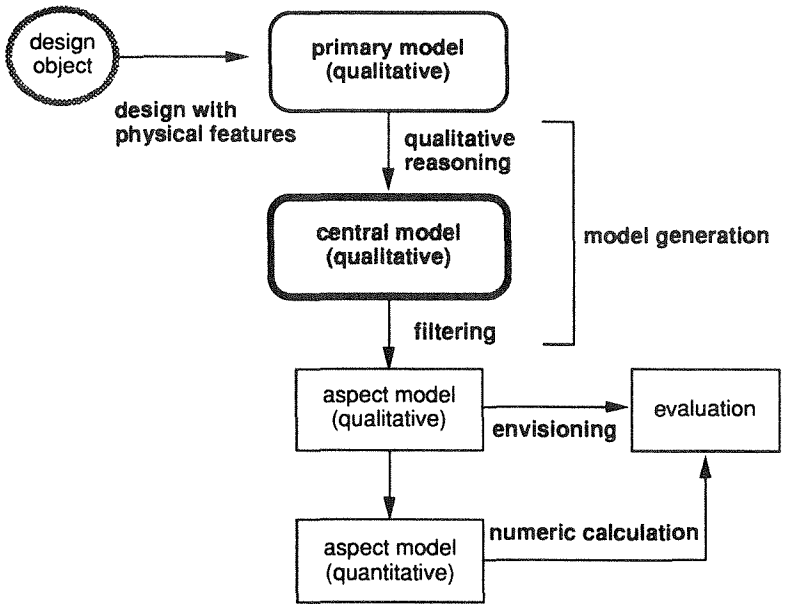

Figure 5: Design with physical features

resentation scheme for physical features. For instance, in order to model a chemical reaction, we had to add a new type of influence to describe generation and destruction of objects. It also became necessary to use additional attributes of objects to describe prerequisites for physical features. To do so, we took the advantage of incremental class definitions of Smalltalk to allow the database to be added new subclasses under the general class of physical phenomena. It avoided changing representation scheme for collected physical features and therefore made it easier to maintain compatibility among physical features in different versions of the database.

4.2 Use of the Physical Feature Database

In order to evaluate the physical feature database, we implemented a prototype system of the metamodel mechanism. As illustrated in Figure 5, the system allows the designer to build a primary model using the physical feature database. It then generates a metamodel and aspect models by referring to the physical feature database. Figure 6 depicts a primary model of an electromagnetic motor. The primary model is composed of electric-currents, voltages, magnetic-fields, attractions, repulsions, rotations, and other physical features. The metamodel of the electromagnetic motor is shown in Figure 7 as a dependency network. From the metamodel, the system generates qualitative aspect models of dynamics, heat, electricity, and layout. Figure 8 depicts a dynamic aspect model showing state transitions of the electromagnetic motor.

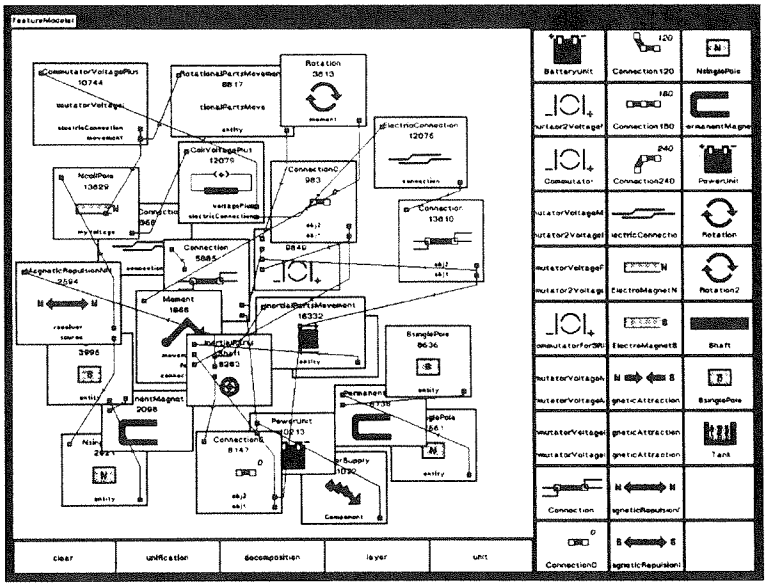

Figure 6: Primary model of a motor

4.3 Limitation of a single ontology

In collecting physical features, it became clear that the ontology of QPT was too weak to cover various physical phenomena. The process-like scheme for physical features was suitable for representing physical phenomena that could be characterized by quantities. But there was no appropriate quantity for representing physical phenomena like support, fit, and slide. Furthermore, physical phenomena such as slide was characterized by a precondition for the moving object contacting to the guiding surface. The meaning of *contact*, however, could not be defined in the physical feature database since there was no characteristic quantity for contact. Nevertheless, we used such vocabulary for describing preconditions, and it resulted in a problem of incompatibility among physical features collected from different domains. This problem derived from the limitation of relying on a single ontology of QPT.

It is not a problem of QPT but comes from the limitation of employing single ontology. In order to represent the physical world, we need to choose a suitable ontology in accordance with the purpose of modeling. Therefore, for representing knowledge about space, time, and causality in the physical feature database, we need to employ multiple ontologies including that of QPT.

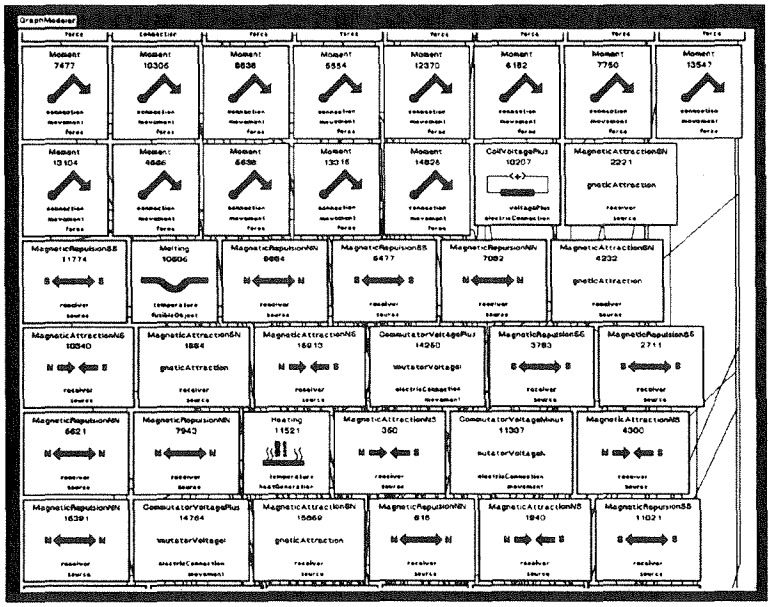

Figure 7: Metamodel of a motor

4.4 Extension to multiple ontologies

In the current implementation of the primary model, it represents only a qualitative state of the design object. But the designer has an image of the desired behavior in the mind. It must be expressed in the primary model to be compared with the result of envisioning. Furthermore, most machines operate under influences by programmed controls or environmental factors. For instance, a linear motor in Figure 9 is driven by changing the path and direction of the electric current through coils. The primary model of the linear motor must be able to represent the four states as the desired behavior. And in order to reason out the behavior of the linear motor from its structure, the sequence of control must be given. If a long term behavior is focused on, the four steps of moving must be aggregated to a continuous linear motion.

In order to represent temporal state changes and influences from outside of the mechanisms, we are trying to integrate temporal logic [1] into the metamodel system. It is used to compare the desired behavior against the result of analyses and to correlate behaviors in different time-scales.

The current implementation of the metamodel is restricted to represent causal dependency. It lacks information of shape and layout. Such information can be represented using geometric modelers. In order to integrate geometric models into symbolic representation of the metamodel, the concepts used for geometric modeling must be available in the metamodel mechanism. We are trying to connect the metamodel

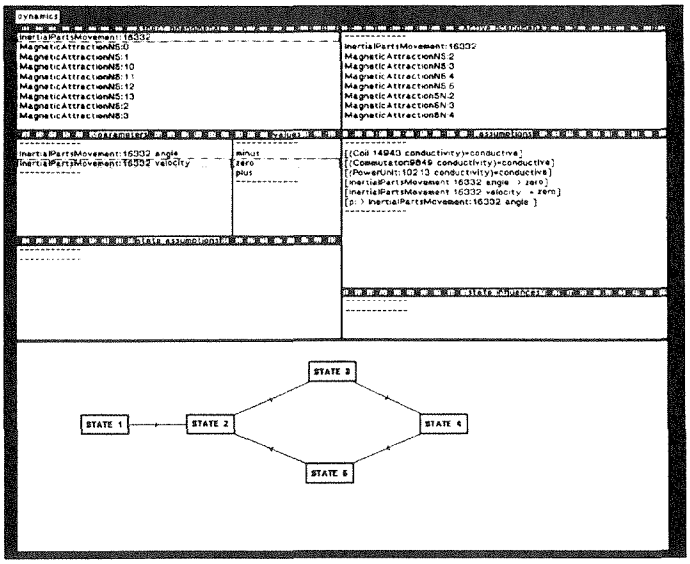

Figure 8: Dynamic aspect model of a motor

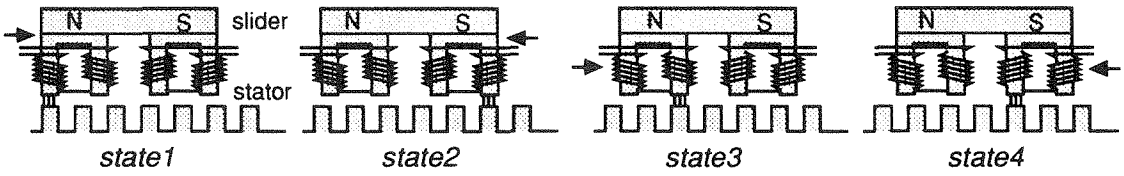

Figure 9: Linear motor

mechanism with a three dimensional solid modeler. To do so, it is important to make correspondences between preconditions for physical features and geometric models.

For instance, there are three qualitatively different layouts, *viz.*, right, front, and left, for a pair of teeth of the stator and slider. We study classification of spatial configuration [2, 8]. To identify critical configurations, we must choose an aspect and physical features related to it, detect preconditions for them, and map the preconditions to spatial representations.

The physical features are suitable for representing causality among physical phenomena. But they are not for describing relationships among vocabulary such as synonyms and antonyms. In collecting physical features, such relationship are hard to be defined by physical features and has been left. In order to cope with differences among ontologies, the metamodel mechanism needs to transfer vocabulary into suitable representation by referring to their definitions. The mechanism is considered to be based on a general knowledge representation like first-order predicate logic. Figure 10 illustrates combination of multiple ontology in the metamodel mechanism.

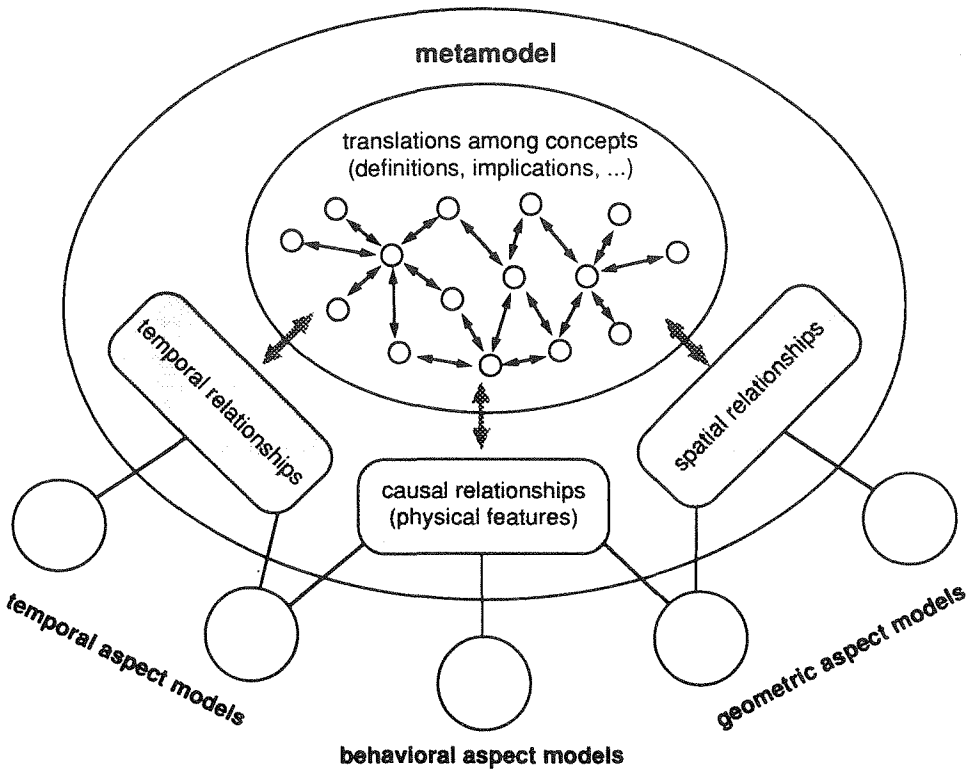

Figure 10: Multiple ontology for the metamodel mechanism

5 Conclusions

In this paper, we presented a project to build the physical feature database. A physical feature is a qualitative description of a physical phenomenon and related attributes. In the modeling environment of the metamodel mechanism, the physical feature database is used for building primary models and reasoning behaviors for dealing with aspect models. We use the framework of QPT as the fundamental scheme of the physical feature database. In collecting physical features, it turned out that a single ontology does not suffice to represent various knowledge about physical phenomena.

Although the necessity of large-scale knowledge bases became widely recognized, there is little methodology. One of the lessons we learned from the project so far is that if the domain knowledge is well systematized, it is only a matter of collection and codification. If not, however, it is extremely difficult even to collect knowledge. In other words, building a knowledge base is enabled by systematization of knowledge. Choosing a right domain as the knowledge source is important in this sense. In addition, it is necessary to articulate and codify common sense knowledge behind domain theories.

References

[1] Allen, J.: Maintaining Knowledge about Temporal Intervals, *Communications of the ACM* Vol. 26, No. 11, pp.832–843 (1983).

[2] Faltings, B.: Qualitative Kinematics in Mechanisms, in Proceedings IJCAI-87, pp. 436–442 (1987).

[3] Forbus, K.: Qualitative Process Theory, *Artificial Intelligence* Vol. 24, pp.85–168 (1984).

[4] Gruber, T.: The Development of Large, Shared Knowledge-Bases: Collaborative Activities at Stanford, Technical Report KSL90-62, Stanford Knowledge Systems Laboratory (1990).

[5] Hayes, P.: The Second Naive Physics Manifesto, in Hobbs, J. and Moore, R. C. (eds.), Formal Theories of the Commonsense World, Ablex, Norwood, NJ, pp. 1–36 (1985).

[6] Hix, C. and Alley, R.: *Physical Laws and Effects*, John Wiley & Sons, London (1958).

[7] Johnson-Laird, P.: *Mental Models*, Cambridge University Press (1983).

[8] Joskowicz, L. and Addanki, S.: From Kinematics to Shape: An Approach to Innovative Design, in Proceedings AAAI-88, pp. 347–352 (1988).

[9] Kiriyama, T., Tomiyama, T., and Yoshikawa.H., : Model Generation in Design, in Fifth International Workshop on Qualitative Reasoning about Physical Systems, pp. 93–108 (1991).

[10] Lenat, D. and Guha, R.: *Building Large Knowledge-Based Systems*, Addison-Wesley, Reading, MA (1989).

[11] Lieberman, H.: Using Prototypical Objects to Implement Shared Behavior in Object Oriented Systems, in Object Oriented Computing 1986, pp. 189–198 (1986).

[12] Nayak, P., Joskowicz, L., and Addanki, S.: Automated Model Selection using Context-Dependent Behaviors, in Fifth International Workshop on Qualitative Reasoning about Physical Systems, pp. 10–24 (1991).

[13] Neches, R., Fikes, R., Finin, T., Gruber, T., Patil, R., Senator, T., and Swartout, W.: Enabling Technology for Knowledge Sharing, *AI Magazine* Vol. 12, No. 3, pp.36–56 (1991).

[14] Roth, K.: *Konstruieren mit Konstruktionskatalogen*, Springer-Verlag, Berlin (1982).

[15] Tomiyama, T., Kiriyama, T., Takeda, H., Xue, D., and Yoshikawa, H.: Metamodel: A Key to Intelligent CAD Systems, *Research in Engineering Design* Vol. 1, No. 1, pp.19–34 (1989).