Transforming Behavioral and Physical Representations of Mechanical Designs

by

S. Finger, J. R. Rinderle

EDRC 24-35-90

Transforming Behavioral and Physical Representations of Mechanical Designs

Susan Finger
Research Scientist
The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

James R. Rinderle
Associate Professor
Mechanical Engineering Department
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

We are exploring the use of formal grammars to represent two distinct but interconnected attributes of mechanical designs: geometry and behavior. By creating a formal description of a limited set of behaviors for mechanical designs and a corresponding description of physical components, we can generate the description of a physical system that takes advantage of the multiple behaviors of its components.

Our approach is based on the following assertions:

- **The behavioral *requirements* of mechanical systems can be represented using a graph grammar.**
- **The behavioral *characteristics* of components can be represented using a graph grammar.**
- **The physical characteristics of designs and components can be represented using formal topological and geometric models.**
- **The behavioral and physical graphs of components can be linked parametrically or algorithmically.**
- **The behavioral specifications graph can be formally transformed into a description of a physical system with associated behavioral and geometric representations.**

The goal of our research is to create a transformational strategy by which the design specifications for a mechanical system can be transformed into a description of a collection of mechanical components. For any given design specification, many different physical systems could potentially meet that specification. Hence there are potentially many transformations that could be applied to a particular design specification. We are interested in those transformations of the design specifications which preserve the original behavior of the design. In this paper, we explore a set of transformations that enable us to move from specifications to topological configurations and from topological configurations to geometric configurations.

1. Introduction

During the design process, a designer transforms an abstract functional description for a device into a physical description that satisfies the functional requirements. In this sense, design is a transformation from the functional domain to the physical domain [Mostow 85, Rinderle 82]; however, the basis for selecting appropriate transformations and methods for accomplishing transformations are not well understood The implicit basis for design transformations in circuits [Steinberg 86], software [Wirth 71], and some architectural applications [Fenves 87] result in a degree and type of modularity not well suited to mechanical devices [Rinderle 86].

The goal of our research is to create a transformational strategy by which the design specifications for a mechanical system can be transformed into a description of a configuration of mechanical components. Both behavioral and physical *requirements* as well as behavioral and physical *characteristics* of the available mechanical components must be represented to execute such a transformational approach to design. Because the interactions of components are important in our synthesis strategy, the representation of the behaviors of mechanical components must be linked to the representation of their physical characteristics; that is, we are concerned with modeling the relationship between form and function of components. Finally, we need a strategy that enables us to transform an abstract description of the desired behavior of a device into a description that corresponds to a collection of physical components.

To realize the goal of formalizing the transformation from the behavioral to the physical domain, we have begun to explore a small domain within mechanical design, the domain of gear box design. Clearly, one reason for selecting this domain is that gear box design is a well-understood, highly-parameterized area of mechanical design. Nevertheless, we believe that our representation and transformation formalism will be applicable in other mechanical design domains, particularly to the class of design problems that we call *configuration design.* By configuration design, we mean designs composed from standard component families but for which allowable configurations are not specified *a priori.*

The representation of a design includes a specified configuration of a set of components and their associated geometrical and behavioral representations[1]. Partially complete designs also consist of components and junction structures and some number of behavioral primitives not yet associated with a component. Since the components have associated with them both behaviors and physical characteristics, the distinction between the representation of designs and the representation of components is in the representation of the physical *configuration* and its associated behavior.

---

[1]The representation of a design also can include manufacturing, assembly, or maintenance information, but we are not concerned with these here.

2. Related Work

Our research builds upon research from several different areas including bond graphs, representation of mechanical behavior, grammars for shape representation, and configuration design. In this section, we briefly discuss related research in these areas.

2.1. A Brief Introduction to Bond Graphs

Our underlying representation for behavior is based on bond graphs. Bond graphs, which were created by Paynter [Paynter 61], provide a convenient and uniform representation for the dynamic behavior of a broad class of physical systems, including those within the mechanical, electrical, hydraulic, thermal, and biological domains. Bond graphs have been used extensively in many different application areas including vacuum cleaners [Remmerswaal 85], robotic manipulators [Margolis 79], and torque converters [Hrovat 85]. A very brief introduction to bond graphs is given here. For a complete discussion of bond graphs and their uses, see Karnopp and Rosenberg [Karnopp 75].

Bond graphs enable mechanical and hydraulic systems to be represented in a manner equivalent to electric circuit diagrams. For example, a spring in a mechanical device acts like an electrical capacitor by storing and releasing energy. One of the most powerful attributes of bond graphs is that they can be used to model integrated electrical, mechanical, and hydraulic systems. Using bond graphs, physical systems are represented as a graph of lumped-parameter, idealized elements. Power is the currency of bond graphs; power flows through the bonds (edges) in the graph, and power is dissipated, stored, supplied, and transformed at the ports (vertices) in the graph.

The ports, or vertices, of bond graphs are divided into three categories:

- 1-port elements dissipate power, store energy, and supply power. Dampers, springs, and masses are the mechanical elements represented by the passive 1-port elements. Force (effort) and velocity (flow) sources are represented as active 1-ports.
- 2-port elements transform power. Transformers are 2-port elements that represent an imposed proportionate relationship between similar quantities, e.g. a gear pair constrains rotational speeds. Gyrators are a 2-port elements that impose a proportionate relationship between dual quantities, e.g. a torque converter constrains the relationship between torque and angular velocity. Power is conserved across a 2-port.
- N-port elements represent the structure of the system corresponding to the connections among the elements. There are two types of N-port elements: 0-junctions and 1-junctions, which correspond respectively to "same force" and "same velocity" connections. Equivalents of Kirkoff's laws apply to N-port elements: the sum of the efforts around a 1-junction is zero (the bonds share a common flow); the sum of the flows around a 0-junction is zero (the bonds share a common effort). N-port elements are power conserving.

In [Finger 89a], we discuss in greater detail our use of bond graphs for synthesis, as opposed to modeling.

2.2. Representation of Function and Behavior

Mechanical engineers tend to use the words function and behavior interchangeably. Qualitative physicists make a distinction between these words; that is, the design's *function* is what it is used for, while its *behavior* is what it does. For example, the function of a clock is to display the time, but its behavior might be the rotation of hands. Similarly, a motor may be designed to function as a prime mover, but can also function as a door stop because it has additional behaviors due to its mass. In this paper, *function* is used to indicate the subset of *behaviors* which are required for the device to perform satisfactorily.

The representation of the function and behavior of mechanical designs has been explored by, among others, Lai [Lai 87], Crossley [Crossley 80], Pahl and Beitz [Pahl 84]. The function structures of Pahl and Beitz provide a graphical system for laying out the functions of a design. In this system, functions such as "mix" or "deliver" are arranged in a graph to represent the overall function of the design. This is similar to our idea of a specifications graph, discussed below; however, there are several important differences. Pahl's work does not discuss how to integrate form specifications or functional constraints with this functional representation. Therefore, while the function structures are used to study functional configurations in Pahl's synthesis strategy, there is little guidance for transforming the function structure to a physical description of the device. Lai has created a formal, English language-based system, called FDL, for representing the function and structure of mechanical designs. In FDL, nouns and verbs are used to create sentences that represent the function of a design, and design rules operate directly on the nouns and verbs in the sentence. Allowable verbs, for example "fasten," do not have physical or mathematical representations and so their meaning is determined by the rules that use them. While the FDL language can represent the function and form of a design it provides no assistance in transforming a functional description into a physical description.

More closely related to our approach for representing behavior is the work of Fenves and Baker [Fenves 87] and of Ulrich and Seering [Ulrich 87, Ulrich 88, Ulrich 89]. Fenves and Baker have created a spatial and functional representation language for structural designs. They use operators that execute a known grammar to generate architectural layouts as well as structural and functional configurations; however, they assume that the layout and structure are independent if they are generated sequentially. Ulrich and Seering also use a formal representation of function based on bond graphs; however, their goal is to create a system in which the bond graphs for single-input, single-output dynamic systems can be automatically synthesized and transformed into physical components. Using a strategy they call *design and debug*, they transform a graph of design requirements directly to functionally independent physical components. Reconfiguration for function sharing is performed after the components have been selected. Prabhu and Taylor [Prabhu 88, Prabhu 89] have also used bond graphs to generate conceptual parametric designs.

2*3. Grammars for Representation of Geometry

Formal grammars that can **generate** and parse valid strings in a language have proven useful in **a number of** fields, most notably, linguistics and computer science. Recently, interest **has been** growing on the use of formal grammars in engineering design. Our work draws primarily from these engineering applications of grammars. We are specifically concerned with graph grammars, the class of grammars that operates on graphs. Tutorials on graph grammars and their applications are given in [Ehrig 87] and [Nagl 87].

Stiny [Stiny 75], who was among the first to apply grammars in design, created shape grammars based on the formalisms of computational linguistics [Chomsky 57]. Architects in particular have been interested in shape grammars, using them to generate families of floor plans, ornamentation, and building layouts [Stiny 78], [Downing 81], [Koning 81], [Flemming 87]. Fitzhorn [Fitzhorn 89] has shown the formal relationship between language theory and solid modeling systems. Woodbury and Heisserman [Woodbury 88], [Heisserman 89] are creating grammars to generate solid models for designs. Pinilla *et al.* [Pinilla 89] have created a grammar that can be used to describe and represent the geometric features of a design. They use a non-manifold topological representation of a design to create a general, but formal, representation of form features.

2.4. Configuration Design

Configuration design and parametric design are active areas of research in mechanical engineering design. For a more complete discussion of this body of work, see [Finger 89b, Finger 89c]. Our research combines configuration design and parametric design because we generate both the structure of the design and the individual components. Many design systems, such as HI-RISE [Maher 85], AIR-CYL [Brown 85], and VT [Marcus 86], utilize either a set of predetermined decompositions of the structure of a design, or utilize design methods that generate, as part of the design process, a decomposition belonging to such a set. Therefore, all of the designs generated by these systems will share, at a relatively low level, a structural similarity. For design domains in which the most desirable structures can be enumerated or explicitly decomposed in advance, this approach proves useful. But there are many design problems where the structural decomposition of the solution is not pre-determined.

3. Representation of Behavior of Specifications and Components

Our underlying representation for behavior is based on bond graphs [Paynter 61]. Using bond graphs, we can construct a formal grammar that gives us a general representation of classes of mechanical behavior. In common practice, bond graphs are constructed to model the behavior of physical systems. We use bond graphs not only to model the behavior of physical systems, but also to represent behavior in the abstract, as with a design specification. Thus a device configuration can be generated by transforming a specification bond graph into a functionally equivalent graph which corresponds to a configuration of available components. The type of graph transformations used are those

that decompose, aggregate, and redistribute graph primitives. In [Finger 89a], we **presented a** bond graph grammar for representing the behavioral requirements of a mechanical system.

A major advantage of using bond graphs to represent design requirements is that we can define transformation rules that alter the structure of the bond graph but that do not alter the dynamic behavior of the system represented by the graph. The implications of this statement are important. Because we can transform the specifications graph to represent many different physical systems, we do not impose an initial structure or configuration on the physical design; that is, we do not require an *a priori* decomposition of the design specifications.

3.1. Representation of Design Specifications

We make the observation that system specifications for engineering designs are of two types: *behavioral* and *physical* That is, some specifications describe at an abstract level the desired behavior of the overall system while others describe physical restrictions or requirements on the final design. For example, the requirements for a vibration absorber might include the *behavior* of the device in terms of frequency and rejection ratio and might also specify *physical* properties such as allowable size and weight The physical and behavioral specifications express the design objective. Since the specifications are given without regard to design configuration, they are independent of each other. (That is, they are independent unless the requirements are contradictory, for example by virtue of physical law.) Physical specifications for a design may or may not be given, while at least some behavioral specifications must be given, since the behavioral specifications express the central aspect of the design objective.

Although the behavioral and physical specifications are independent in the *functional* domain, they are coupled in the *physical* domain, because any physical arrangement results in a specific set of behaviors. In the physical domain, the physical and behavioral characteristics of individual components depend on one another, and the behavior of the whole design depends strongly on the configuration and interaction of components. The representation of interactions is essential for our purposes since we are investigating the effects of and the means for achieving functional integration in design.

Physical and behavioral specifications and characteristics can be represented as combinations of abstract primitives. They are abstract because each primitive corresponds to only one behavioral or physical characteristic. Individually they do not correspond to any particular component or configuration of components, but collectively they may represent the design specifications or the form and behavior of components. There are two important criteria that a set of primitives must satisfy. One requirement is that the set of primitives chosen be complete; that is, all relevant behavioral and physical characteristics must be representable by some combination of primitives. The other is that the number of primitives must be small, although not necessarily minimal. The latter requirement minimizes combinatorial problems associated with representing a single

behavior in different ways. In addition, the primitive behavioral and physical elements should enable commonly used components and typical specifications to be represented easily.

3.2. Representation of Behavioral Requirements of Mechanical Systems

When the behavioral primitives correspond to bond graph elements, the behavioral specifications can be represented easily in a bond graph. This specification bond graph is not unique, indeed any behaviorally equivalent graph is an acceptable representation of the specifications. The specifications graph represents only the desired behavior of the design.

Deriving a correct specifications graph for a design problem is, in itself, a major research problem. Initially, for transmission design, we assume that deriving a specifications graph from the problem statement is straightforward. Later, as less constrained design domains are explored, we will look at the problem of constructing the specification graph from the design requirements.

For example, consider the design of a simple gear box which is to have a single input shaft and two output shafts. The first output shaft is to be offset a prescribed amount from the input shaft and is to operate at a 20:1 speed reduction relative to the input shaft. The second output shaft operates at a 240:1 reduction and must be in-line with the input shaft. The direction of shaft rotation is not relevant. The kinematic specification can be written as a bond graph as shown in Figure 1.

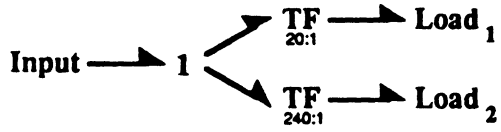

Figure 1: Behavioral Specification Graph for a Gear Box

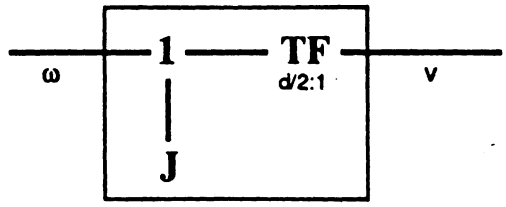

Figure 2: Behavior Graph for a Single Spur Gear

3.3. Representation of Behavioral Characteristics of Components

One goal in transforming the specification graph is to make the correspondence between the required behaviors and the available components; therefore, it is convenient to represent the behavioral characteristics of components with the same basic structure that is used for specifications, *i.e.* bond graphs. Each component has a bond graph associated with it that represents its behavior *in isolation.* As an example consider a single spur gear as a component. The behavior graph for the component is shown in Figure 2. In this example, a relationship is imposed between flow quantities rotational speed, co on the left and surface speed, v, on the right. The graph also includes the gear inertia.

One crucial difference between specifications and components is that the behaviors and physical characteristics of a component are inherently linked; no single characteristic, either behavioral or physical, can be obtained in isolation. So, for example, the spur gear in Figure 2 may be selected for its power transformation characteristics, but its inertial characteristics are implicitly selected as well. The next section addresses some of the issues in linking the representation of the behavioral and physical characteristics of components.

3.4. Representation of Designs

A complete design is the specified configuration of a set of components. Since the components have associated with them both behaviors and physical characteristics, the distinction between the representation of designs and the representation of components is in the representation of the behavioral and physical *configuration.* The behavioral configuration fundamentally consists of the kinematic connections, e.g. mounting to a frame, rigid connections, or rolling. Most of the common kinematic arrangements can be categorized [Reuleaux 54] and reduced to a bond graph junction structure. A 1-junction, for example, represents a common translation^ velocity. In this way, the behavior of complete devices may be represented in terms of the behavioral bond graphs of components and a number of bond graph junction structures representing kinematic connections. Partially complete designs also consist of components and junction structures and some number of behavioral primitives not yet associated with a component.

Ultimately, we will represent the physical characteristics of the device in much the same way, i.e. as a composite of the topology and geometry graphs of the components and configuration. To date we have represented the physical characteristics of devices by aggregating, on an *ad hoc* basis, the geometric parameters of the components and the layout.

Physical components, like specifications, have both behaviors and physical characteristics. Again, the behaviors and physical characteristics of a component are inherently linked; no single characteristic can be obtained in isolation. Each physical component is represented as an object that has a behavioral representation, a physical representation, and an explicitly represented interaction between the two. The

relationships can take the form of design equations, analytical models that relate geometric characteristics to behavioral characteristics, or data base entries that prescribe a relation between the physical and the behavior characteristics. For example, the weight of a helical coil spring, which is a physical characteristic, is proportional to the product of stiffness and the square of allowable deflection, which are behavioral characteristics. These relationships are often critical during the design.

It is important to note that the completeness of the behavioral and physical representations determines the extent to which the additional behaviors can be exploited. For example, with a worm gear, the right angle shaft configuration and the self-locking property must be represented explicitly, or be derivable from the representation, for it to be recognized and utilized in the design process. Therefore, this methodology requires a known component base.

To complete this methodology, we plan to represent the physical characteristics of designs and components using another graph grammar that is based on an augmented topology graph [Pinilla 89] and non-manifold geometry [Gursoz 89]. We plan to link parametrically the bond graph representation and the topology and geometry graph representation. With a geometric representation, characteristics such as the volume or mass can be modeled and computed, and from the bond graphs, the dynamic behavior of the final design can be modeled.

4. Transformation of Specifications into Physical Descriptions

The specification graph is composed of abstract behavioral primitives that do not yet correspond to any physical components. To arrive at a design, we transform the specification graph, without changing function, to obtain a graph that more nearly corresponds to a collection of components. In this section, we discuss the principles that we use in transforming the specifications into a physical system. The transformation process, which is described in greater detail in [Hoover 89], is guided by the function integration and incidental behavior principles and by knowledge of the available components. It provides an approach to finding configurations that satisfy the physical requirements, thus eliminating blind search. This approach has been successful for the design of mechanical power transmissions and serves to illuminate several important issues for extending this technique to other domains.

4.1. Behavior-Preserving Transformations

The specification graph is not unique. Many equivalent graphs can represent the same behavior. By transforming the specification graph, without altering behavior, we can explore design alternatives that have the same behavior. Some of the resulting configurations result in physically desirable designs and some do not. In addition to knowing the general rules for behavior-preserving transformations, we need to know which transformations to apply and the sequence of application. Guidance in selecting which behavior-preserving transformation to apply comes from the physical requirements

of **the** system, from the physical characteristics of the components, and from the **relationship between** geometry and behavior of the component

For **the gear** box design, more economical designs often have maximum commonality of the power paths. Therefore, one type of behavior-preserving transformation of interest is one that preserves the dynamic behavior but which implies a different kinematic configuration. For example, Figure 3 is a behavior-preserving transformation of the specification graph from Figure **1.**

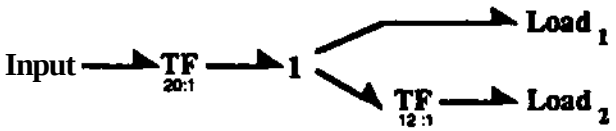

Figure 3: Functionally Equivalent Graph using a Common Power Path

4.2* Component-Directed Transformations

Because mechanical designs are characterized by a high degree of integration, transformations should be directed toward this goal. Integration in a design requires the appropriate utilization of the behavioral and physical characteristics of its components. Therefore, intelligent application of transformations requires utilizing both the behavioral and physical representations of the components to guide the selection process. Transformations selected in this way are *component-directed transformations* because their selection and use are directed by knowledge of the available components. It is the class of component-directed transforms that ultimately enables the selection of a single component to fulfill multiple functions, e.g. selecting a worm gear to execute speed reduction, shaft offset and right angle functions as described more fully in [Hoover 89].

Because the library of components consists only of spur gears, and because the speed ratio that can be obtained with a pair of spur gears is limited to 8:1, the specification graph shown in Figure 3 cannot be transformed directly into a collection of components. A component-directed transform can be applied to split the transformer elements into elements that can be mapped into individual pairs of spur gears. The result of one such transformation is shown in Figure 4.

The transformation shown in Figure 4 imposes a relationship between two rotational speeds that is achieved using a *pair* of spur gears. Applying a lower-level, component-directed transform results in a behaviorally equivalent graph in which the transformer elements correspond to individual spur gears as shown in Figure 5. This graph preserves the required kinematic behavior of the gear box. If individual transformer elements are interpreted as gears, then the graph implies topological relationships among gears, that is,

it implies which gears mesh. The topological information, however, is not complete. The graph shown in Figure 5 does not dictate how gears are arranged on shafts. Both the geometrical and topological information associated with shaft layout is absent.

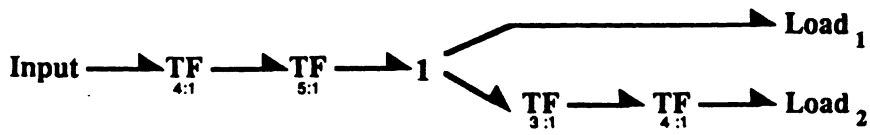

Figure 4: Graph with Kinematics Equivalent to Figure 3

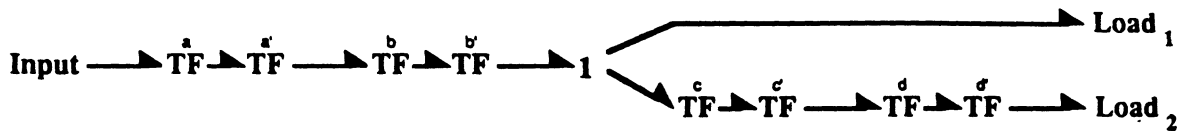

Figure 5: Transformation into Spur Gear Equivalents

a			
a'	b		
	b'	c	
		c'	d
			d

Figure 6: Shaft Matrix Corresponding to Figure 5

5. The Shaft Matrix

We are now interested in investigating alternatives that have the same kinematic functionality but different physical configurations. The power path topology of Figure 5 can be represented in a matrix format. The individual *TF* elements that correspond to individual gears are labeled *a*, *a'*, *b*, *b'*, etc. Using a matrix-like format, each row of the matrix corresponds to a unique rotational speed. The columns in the matrix correspond to unique translational speeds, for example, the surface speed at the pitch diameter of a gear. Since the gears *a* and *a'* mesh with each other, they have a common velocity at their pitch diameters and so are represented in the same column of the matrix. Similarly, *a'* and *b* share a rotational speed and therefore fall within the same row of the matrix. The shaft matrix is readily completed by inspection as shown in Figure 6. Note that

column exchanges have no impact on the implicit commonalities among gear surface speeds or shaft rotational speeds. Similarly, exchanging rows has no impact on power path topology or the functional kinematics.

In a sense, this matrix is a canonical form for the gear box and could be implemented with a shaft and associated bearings for each of the five rows in the matrix. Although implementation may be direct it is not economical due to the absence of any shafting commonality. Elements with different rotational speeds may indeed correspond to the same shaft position. For example, a gear stack may rotate relative to its supporting shaft relying on the shaft only to maintain its position in space. Similarly shafts may be nested. One such configuration is shown in Figure 7.

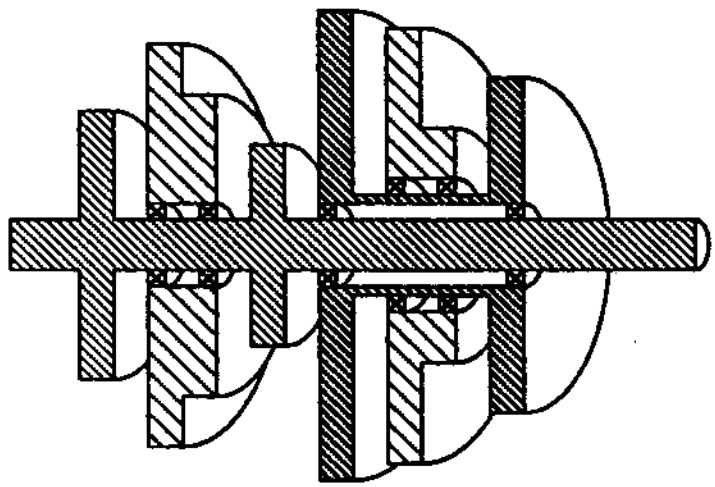

Figure 7: Nested Shafts

Although shafts may be shared, the allowable mechanical configurations impose certain nesting requirements on shafts that manifest themselves as topological constraints. These constraints can be represented and enforced by delimiting the elements in each row of the shaft matrix as shown in Figure 8.

The column position within the matrix can be interpreted to correspond to physical position. For example, column one can be interpreted as the left end of the gear box while the last column is at the right end. Columns of the matrix can be interchanged to obtain input and output gears at the proper physical locations. The shaft matrix shown in Figure 8 is already an acceptable configuration because the input, *a* is nearest the left of the gear box, while the two outputs, *d* and c, are accessible to the right edge. We now seek commonality in the shafting configuration to maintain not only the power path topology, but also the physical access of input and output shafts to the appropriate position on the gear box. Rotational elements with different speeds can be combined on

<a>			
[a[1]	b]		
	(b*	c}	
		(C	d)
			<d'>

Figure 8: Shaft Matrix with Nesting Constraints

<a>			<d'>
[a'	b]		
	{b[1]	c}	
		(C	d)

Figure 9: Shaft Matrix Transformed so the Input and Output Shafts are In-line

<a >			<d'>
[a[1]	b]	(c[1]	d)
	{b[1]	c}	

Figure 10: Shaft Matrix Transformed to Reduce the Number of Shafts

a single shaft by combining two rows whenever the appropriate shaft matrix elements are null. Gear *d* corresponds to the 240:1 speed shaft and must be in-line with the input shaft. Combining the first and last rows of the shaft matrix shown in Figure 8 results in the shaft matrix shown in Figure 9.

Because one requirement of the original specification was that the 20:1 speed shaft be offset from the input shaft, the *V - c* row cannot be combined with the first row; however, the second and fourth rows can be combined on a common shaft as shown in Figure 10. Note that this shaft matrix represents shafts at three physical locations on which all 8 gears reside. Note also that it is not possible to reduce the number of rows in this shaft matrix further, therefore, this configuration has the minimum number of shaft locations for this specific power path topology. Direct translation of the shaft matrix in Figure 10 is shown in Figure 11.

Alternative power path topologies may in fact result in fewer required shafts. Using

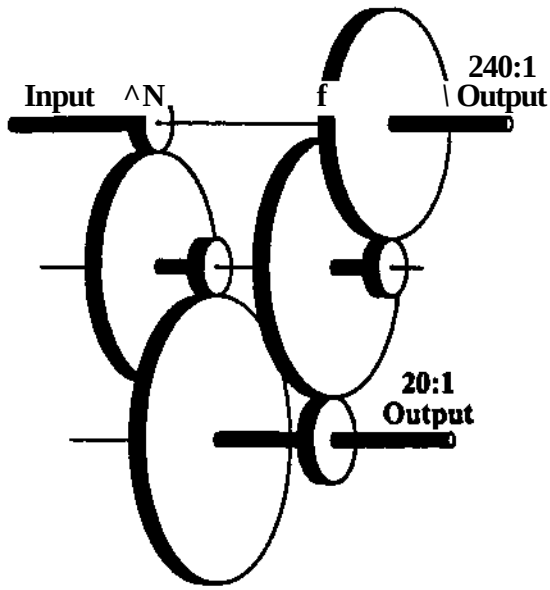

Figure 11: Shaft and Gear Configuration Corresponding to Figure 10

similar methods, the original kinematic specification from Figure 1 may be transformed into a functionally equivalent specification as shown in Figure 12. If each of the transformers is interpreted to be a pair of spur gears than this configuration will consist of 10 individual spur gears compared to the previous configuration involving only 8. However, when this power path topology is mapped onto to the shaft matrix and the number of shafts is minimized, a configuration using only two shafts can be found. The final shaft matrix and the physical configuration corresponding to this alternative are shown in Figures 13 and 14.

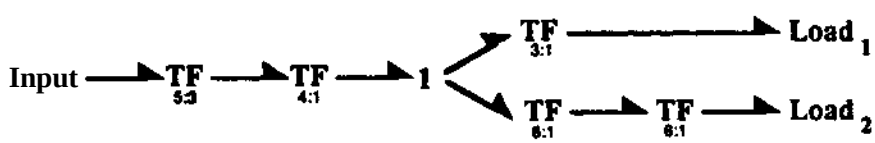

Figure 12: Functionally Equivalent Specification

6. Results and Future Work

In this paper, we have presented a transformational paradigm for design in which design specifications are transformed into physical descriptions. We focus on the class of configuration design in which designs are composed from standard component families.

<a>	{b[1]	c	d[1]}	<e* >
[a[1]	b]	(c[1])	[d	e]

Figure 13: Final Shaft Configuration for Figure 12

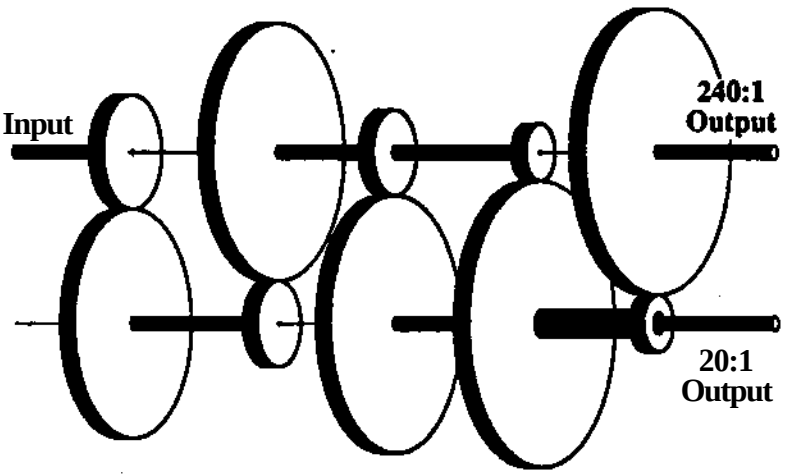

Figure 14: Gear and Shaft Configuration for Figure 13

but for which no *a priori* configuration is assumed. In the domain of simple gear design, we first apply behavior-preserving transformations to maximize common power paths. We then apply component-directed transformations which preserve behavior and power path topology and which make the best use of component characteristics. Finally, we apply transformations that preserve the kinematic topology and which result in compact layouts and shaft access. At each stage, many alternative transformations are valid, each corresponding to a design alternative. We select and apply transformations to obtain synergistic interactions among components.

Our research will continue in the following areas:

- **Expanding the behavioral and physical characteristics that can be represented.**
- **Continuing to develop the representation of the interconnection between geometry and behavior.**
- **Increasing the number of components from different mechanical design domains represented in our system, and representing them at higher-levels of abstraction as well as at greater levels of detail. ***
- **Creating new transformation strategies, refining the existing ones, and in particular, creating general transformations that operate based on system-level characteristics rather than on component characteristics.**

Acknowledgments

The authors are pleased to acknowledge the support of the Design Theory and Methodology Program of the National Science Foundation (NSF Grants DMC-84-51619 and DMC-88-14760) and the Engineering Design Research Center at Carnegie Mellon University (NSF Grant CDR-85-22616).

References

[Brown 85] Brown, D. and Chandrasekaran, B., "Expert Systems for a Class of Mechanical Design Activity," in *Knowledge Engineering in Computer-Aided Design*, Gero, J., ed., North Holland, 1985, pp. 259-290.

[Chomsky 57] Chomsky, N., *Syntactic Structures*, Humanities Press, Atlantic Highlands, NJ, 1957.

[Crossley 80] Crossley, E., "Make Science a Partner," *Machine Design*, April 24 1980.

[Downing 81] Downing, F. and Flemming, U., "The Bungalows of Buffalo," *Environment and Planning B*, Vol. 8, 1981, pp. 269-293.

[Ehrig 87] Ehrig, H., "Tutorial Introduction to the Algebraic Approach of Graph Grammars," *Graph-Grammars and their Applications to Computer Science*, Springer-Verlag, New York, Lecture Note Series 1987, pp. 3-14.

[Fenves 87] Fenves, S. J. and Baker, N. C., "Spatial and Functional Representation Language for Structural Design," *Expert Systems in Computer-Aided Design*, Elsevier Science (North-Holland), IFIP 5.2 1987.

[Finger 89a] Finger, S. and Rinderle, J., "A Transformational Approach to Mechanical Design Using a Bond Graph Grammar," *Proceedings of the First ASME Design Theory and Methodology Conference*, American Society of Mechanical Engineers, Montreal, Quebec, September 1989.

[Finger 89b] Finger, S. and Dixon, J. R., "A Review of Research in Mechanical Engineering Design, Part I," *Research in Engineering Design*, Vol. 1, No. 1, 1989, pp. 51-67.

[Finger 89c] Finger, S. and Dixon, J. R., "A Review of Research in Mechanical Engineering Design, Part II," *Research in Engineering Design*, Vol. 1, No. 2, 1989, pp. 121-137.

[Fitzhorn 89] Fitzhorn, P., "The Formal Languages of Solid Representation," *Out for review in Design Computing*, 1989.

[Flemming 87] Flemming, U., "More Than the Sum of Parts: The Grammar of Queen Anne Houses," *Environment and Planning B*, Vol. 14, 1987, pp. 323-350.

[Gursoz 89] Gursoz, E- L. and Prinz, F. B., "Corner-Based Representation of Non-manifold Surface Boundaries in Geometric Modeling/' Technical Report, Engineering Design Research Center, Carnegie Mellon University, 1989.

[Heisserman 89] Heisserman, J. and Woodbury, R., "Solid Grammars for Generative Geometric Design," *NSF Engineering Design Research Conference,* University of Massachusetts, Amherst, Amherst, MA, June 11-14 1989.

[Hoover 89] Hoover, S. P. and Rinderlc, J. R., "A Synthesis Strategy for Mechanical Devices," *Research in Engineering Design,* Vol. 1,1989, pp. 87-103.

[Hrovat 85] Hrovat, D. and Tobler, W. E., "Bond Graph Modeling and Computer Simulation of Automotive Torque Converters," *Journal of The Franklin Institute,* 1985, pp. 93-114.

[Karnopp 75] Karnopp, D. and Rosenberg, R., *System Dynamics: A United Approach,* John Wiley & Sons, New York, 1975.

[Koning81] Koning, H. and Eizenberg, J., "The Language of the Prairie: Frank Lloyd Wright's Prairie Houses," *Environment and Planning B,* Vol. 8, 1981, pp. 295-323.

[Lai 87] Lai, K. and Wilson, W. R. D., "FDL: A Language for Function Description and Rationalization in Mechanical Design," *Computers in Engineering,* ASME, New York, 1987, pp. 87-94.

[Maher 85] Maher, M. L., "HI-RISE and Beyond: Directions for Expert Systems in Design," *Computer-Aided Design,* Vol. 17, 1985, pp. 420-427.

[Marcus 86] Marcus, S., Stout J. and McDermott, J., "VT: An Expert Elevator Designer that Uses Knowledge-Based Backtracking," Technical Report CMU-CS-86-169, Department of Computer Science, Carnegie Mellon University, 1986.

[Margolis 79] Margolis, D. L. and Karnopp, D. C, "Bond Graphs for Flexible Multibody Systems," *Transactions of the ASME,* Vol. 101,1979, pp. 50-57.

[Mostow 85] Mostow, J., "Toward Better Models Of The Design Process," *The AI Magazine,* Spring 1985.

[Nagl 87] Nagl, M., "Set Theoretic Approach to Graph-Grammars," *Graph-Grammars and their Applications to Computer Science,* Springer-Verlag, New York, Lecture Note Series 1987, pp. 41-54.

[Pahl 84] Pahl, G. and Beitz, W., *Engineering Design,* The Design Council, Springer-Verlag, London, 1984.

[Paynter61] Paynter, H. M., *Analysis and Design of Engineering Systems,* MIT Press, Cambridge, MA, 1961.

[Pinilla 89] Pinilla, J. M., Finger, S. and Prinz, F. B., "Shape Feature Description and Recognition Using an Augmented Topology Graph Grammar," *NSF Engineering Design Research Conference,* University of Massachusetts, Amherst MA, June 11-14,1989, pp. 285-300.

[Prabhu 88] Prabhu, D. R. and Taylor, D. L., "Some Issues in the Generation of the Topology of Systems with Constant Power-Flow Input-Output Requirements," *Advances in Design Automation,* Rao, S. S., ed., American Society of Mechanical Engineers, New York, NY, 1988, pp. 41-48.

[Prabhu 89] Prabhu, D. R. and Taylor, D. L., "Synthesis of Systems from Specifications Containing Orientations and Positions Associated with Generalized Flow Variables," *1989 Design Automation Conference,* American Society of Mechanical Engineers, New York, NY, 1989.

[Remmerswaal 85] Remmerswaal, J. A. M. and Pacejka, H. B., "A Bond Graph Computer Model to Simulate Vacuum Cleaner Dynamics for Design Purposes," *Journal of The Franklin Institute,* 1985, pp. 83-92.

[Reuleaux 54] Kennedy, A. B. W., editor, *The Kinematics of Machinery,* Dover Publications, New York, 1854.

[Rinderle 82] Rinderle, J. R., *Measures of Functional Coupling in Design,* PhD dissertation, Massachusetts Institute of Technology, June 1982.

[Rinderle 86] Rinderle, J. R., "Implications of Function-Form-Fabrication Relations on Design Decomposition Strategies," *Computers in Engineering, 1986,* American Society of Mechanical Engineers, Chicago, 1986, pp. 193-198.

[Steinberg 86] Steinberg L., Langrana N., Mitchell, T., Mostow, J. and Tong, C, "A Domain Independent Model of Knowledge-Based Design," Technical report AI/VLSI Project Working Paper No. 33, Rutgers University, March 1986.

[Stiny 75] Stiny, G., *Pictorial and Formal Aspects of Shape and Shape Grammars,* Birkhauser, Basel, 1975.

[Stiny 78] Stiny, G. and Mitchell, W. J., "The Palladian Grammar," *Environment and Planning B,* Vol. 5,1978, pp. 5-18.

[Ulrich 87] Ulrich, K. and Seering, W. P., "Conceptual Design: Synthesis of Systems Components," *Intelligent and Integrated Manufacturing Analysis and Synthesis,* American Society of Mechanical Engineers, New York, 1987, pp. 57-66.

[Ulrich 88] Ulrich, K. T. and Seering, W. P., "Function Sharing in Mechartical Design," *7th National Conference on Artificial Intelligence,* AAAI-88, Minneapolis, MN, August 21-26 1988.

[Ulrich 89] Ulrich, K. T. and Seering, W. P., "Synthesis of Schematic Descriptions in Mechanical Design," *Research in Engineering Design,* Vol. 1, No.

1,1989.

[Wirth 71J Wirth, N., "Program Development by Stepwise Refinement," *Communications of the ACM*, Vol. 14, No. 4,1971, pp. 221-227.

[Woodbury 88] Woodbury, R., Carlson, K., and Heissennan, J., "Geometric Design Spaces," *Second IFIP WG 52 Workshop on Intelligent CAD*, IFIP, Cambridge, UK, 19-22 September 1988.

Table of Contents

Abstract 1
1. Introduction 2
2. Related Work 3
    2.1. A Brief Introduction to Bond Graphs 3
    2.2. Representation of Function and Behavior 4
    2.3. Grammars for Representation of Geometry 5
    2.4. Configuration Design 5
3. Representation of Behavior of Specifications and Components 5
    3.1. Representation of Design Specifications 6
    3.2. Representation of Behavioral Requirements of Mechanical Systems 7
    3.3. Representation of Behavioral Characteristics of Components 8
    3.4. Representation of Designs 8
4. Transformation of Specifications into Physical Descriptions 9
    4.1. Behavior-Preserving Transformations 9
    4.2. Component-Directed Transformations 10
5. The Shaft Matrix 11
6. Results and Future Work 14
Acknowledgments 16
References 16