

Geometric Reasoning for Shape Design

George Turkiyyah and Omar Ghattas

Department of Civil Engineering

Carnegie Mellon University

Pittsburgh, PA 15213

ght@cs.cmu.edu, ghattas@ce.cmu.edu

Abstract

This paper describes a methodology for the design of shapes. Starting from an initial shape, a geometric reasoning kernel is used to generate and control a sequence of numerical optimization subproblems that converges to a final design—a topology and associated geometry—that can be significantly different from the starting shape. A subproblem in the sequence is systematically formulated from a geometric abstraction of current shape, and its objective function, constraints and bounds are dynamically derived. The geometric representation of the shape is adaptive and changes throughout the problem solving process to accommodate the shape change trends that occur. Shape evolution takes place within each subproblem and between subproblems. Intra-subproblem evolution is responsible for geometric modifications while inter-subproblem evolution handles topology modifications. The combination of geometric reasoning and numerical optimization techniques provides a robust and systematic methodology for shape synthesis that can generate new design shapes without relying on heuristic or domain specific knowledge.

The Nature of Shape Design

Shape design can be defined as the problem of generating a topology and associated geometry, i.e., a domain in a Euclidean space. Not unlike other design problems, the goal is to obtain a solution that can fulfill a desired functionality and satisfy some externally specified set of constraints. In the case where it is possible to generate a number of solutions and quantitatively evaluate them, we seek the solution that optimizes some evaluation criteria. We can contrast shape design to two other classes of design problems: parameter and configuration design.

Parameter design is the class of design problems that can be formulated as a process of searching well-defined design spaces, to find the best values to assign to specified design variables. Structure—or configuration—

design is concerned with synthesizing a compound artifact from a known set of primitive elements, as to achieve required specifications. Unlike parameter design, where the kinds of design parameters of the problem are known, the set of variables that are relevant to a solution, and must be assigned values, changes dynamically in response to decisions made during the course of problem solving [Mittal and Falkenhainer, 1990]. At the start of the design it is not known what pieces the artifact will have or how they will connect and interact. As the design proceeds, parts are chosen from a palette of prototypes (i.e., parameterized parts), modified if necessary, assigned values and combined. It is often the case that mathematical programming techniques, dynamic propagation techniques, and significant amount of domain knowledge (in the form of rules, plans, abstraction hierarchies, modification advice, appropriate aggregations, etc.) must be combined to tackle the very large design spaces.

Shape design exhibits a different character than parameter or structure design: neither design parameters are defined, nor prototypes that can be combined are available. In structure design, the design variables, although not available as part of the problem specifications, can be obtained as the union of the parameters that define the constituent components and the variables that define their connections. In shape design, no such mechanism for deriving design variables exists.

Previous Work

Approaches to shape design can be classified in three classes: parameter-based, grid-based and knowledge-based. Parameter-based methods preparameterize the design space, grid-based methods discretize the design domain, and knowledge-based methods exploit the structure of the problem domain to modify the shape as the design progresses. We discuss these methods and their limitations below.

Parameter Based Methods. Parameter-based methods transform the shape design problem into a numerical optimization problem, whose solution is the desired final shape. The initial shape is parameterized, often manually, using one or more of the following

types of variables: coordinates of boundary points, distances along a direction from a fixed datum (e.g., rays emanating from a central point, segments perpendicular to a straight line), coefficients of boundary curves and surfaces such as splines or polynomials, etc. These strategies for choosing the design variables have several shortcomings: they do not insure shape integrity (the shape may overlap or fold upon itself); they restrict the range of shapes that can be reached, since particular assumptions are imposed on the form of the design space; and they lack adaptivity. Design variables are static and cannot take into account shape evolution even though a different choice of design variables becomes more relevant as the design progresses. Parameter-based methods are best used to fine-tune the boundaries of an already designed shape rather than generate new designs.

Grid Based Methods. Grid based methods discretize the design domain and use the resulting grid as the design space of the synthesis process. Although the grids can vary in shape (they can be cartesian, triangular, hexagonal, polar, etc.), grid based methods all operate at the cell level. Several grid-based approaches have been proposed. One approach makes local design decisions based on some criteria of an individual cell and possibly its neighbors. For example, if an evaluation criterion at a cell is below a certain threshold the cell is removed from the design. Other approaches attach binary variables to each of the cells (0 denoting that the cell is not part of the shape, 1 otherwise), or equivalent continuous variables and formulate a mixed-integer programming problem whose solution is the final design. Deficiencies of grid-based methods include: (1) geometric and other constraints cannot always be imposed on the design and taken into account by the design process; (2) grid-based designs are ambiguous since it is not clear how to interpret the occupancy grids generated, which often include isolated cells or islands and other anomalies; and (3) if formulated as optimization problems, they are computationally very expensive.

Knowledge Based Methods. Knowledge-based approaches have been suggested for shape design. Knowledge-based approaches rely on particular characteristics of the problem space to devise specific problem solving methods. For example, significant object characteristics are often captured by a symbolic shape description on which a variety of domain-dependent operators apply [Murthy and Addanki, 1987]. Other approaches decompose the object to be designed, in a domain dependent manner, into a set of subobjects—each of which can be described in terms of known functional parameters (often feature based descriptions), that can be heuristically set and modified. Although knowledge based approaches are reasonable for limited domains, they lack the generality and robustness that are needed for a general system. They suffer from the combinatorial explosion of knowledge structures (rules,

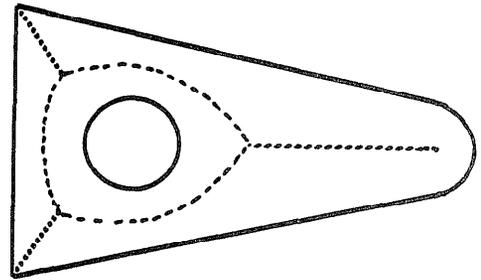


Figure 1: Example of a Skeleton

operators, plans, critics, etc.) needed to cover significant domains.

The goal of this paper is to develop a methodology for the design of shape that overcomes the deficiencies of the above methods.

Geometric Abstractions for Shape Design

An important element of shape synthesis is the shape description. A robust approach to shape design must include a geometric representation that is general purpose, simple to reason about, stable under minor object changes and capable of topological and geometric evolution. As described above, the geometric descriptions used by previous approaches to shape design (sequence of boundary points, B-spline parameters, occupancy grids, symbolic domain-specific representations, etc.) have been inadequate for general synthesis. The shape representation that we propose here is based on a geometric construct called the skeleton. The skeleton can be viewed as an abstract canonical representation of shape [Turkiyyah and Fenves, 1990].

The skeleton can be described in terms of the distance of a point x to a boundary A which is defined as the minimum distance to any point on the boundary: $d(x, A) = \min d(x, z), \forall z \in A$. Given a region with boundary A , we can associate with every point of the region its distance to A . For some points, however, we notice that the distance is not achieved uniquely. For such points *more than one* boundary point satisfies the minimum distance property: these “singular” points define the skeleton. In other words, the skeleton is the set of points that are minimally equidistant from at least two boundary points [Duda and Hart, 1973]. An Example of a shape skeleton is shown in Figure 1.

The skeleton of a shape is an intuitive and appealing representation. It is a first order approximation of the shape and captures significant aspects of intrinsic shape characteristics. For example, subshapes that are elongated have a corresponding skeleton arc that follows their middle axis, pointed subshapes have a skeleton arc that follow the bisector and rounded subshapes

have skeleton arcs that end at a point whose distance to the boundary is equal to the minimum radius of curvature of the rounded region. Moreover, the arcs of the skeleton form a planar graph whose topology is directly related to the topology of the original shape. In particular, the skeleton of a simply connected region is a tree; the skeleton of a multiply connected region is a graph with a number of cycles equal to the number of holes in the original shape. Moreover, the skeleton is information preserving: since each point x retains the information on the minimum distance to the boundary, the original shape can be recovered as the union of disks centered on the skeleton points. The notion of a skeleton can be generalized to include both interior and exterior skeleton arcs, where the exterior skeleton is defined as the is the topological closure of the locus of the maximal inscribable disks included in A^c —the complement of A .

Line segments that join a skeleton point to its corresponding boundary point will be termed *rays*. From the definition of the skeleton, it follows that rays are perpendicular to the boundary, provided that the corresponding boundary is locally smooth. If the end point of a ray is located at a position that has a discontinuous derivative (e.g., concave angular corner), then the direction of the ray will fall between the right-side and left-side perpendiculars to the boundary. If the start and end points of a ray coincide (e.g., convex corner), the direction of the ray will be defined as the direction of the skeleton at that point. The skeleton direction at such points has a slope equal to the average of the slopes of the right and left perpendiculars to the boundary. Vertices of the skeleton are equidistant from at least three boundary points.

The skeleton is particularly suited as a representation for shape design. It overcomes the restricted flexibility and robustness of other approaches because of its:

- *Generality.* The skeleton is a canonical representation of shape. It provides a general purpose mechanism for representing a geometric object by a graph. Graphs are more easily amenable to manipulation in design systems.
- *Adaptivity.* The structure of the skeleton graph tracks object shape. The topology and geometry of the skeleton intrinsically depend on object shape and boundary curvature, and continuously change as the object changes. Hence skeleton-based representations appear appropriate for following and defining shape evolution during optimization.
- *Object Centeredness.* The skeleton of a shape is unique and only depends on the spatial features of the object independent of specific coordinate axes choices or object locations in space. A corollary of this, as we will describe in §4, is that the skeleton can guarantee shape integrity throughout the design process.

Description of the Methodology

Our shape synthesis methodology consists of iterating on the following four steps: (1) computing the skeleton of the current shape (techniques for generating skeletons are describes in [Turkiyyah and Fenves, 1990]); (2) choosing design variables based on the computed skeleton; (3) formulating an optimization problem with appropriate objective, bounds and constraints; and (4) generating a new shape that becomes the basis of a new design iteration.

The design process begins from an arbitrary initial shape and continues until no significant change in shape and/or objective measure occurs, or until the user is satisfied with the existing shape. Our initial experiments indicate that the starting shape is not very important. To a large extent, a different initial shape will typically take a different path to eventually arrive at the same final shape, although we can imagine cases where the final design is sensitive to the initial starting shape.

Design Variables

Design variables that describe the shape for design optimization will be chosen as skeleton rays (Figure 2). As described in §2, rays are segments emanating from points on the skeleton and perpendicular to the boundary (if the boundary is smooth), or along a direction contained between the right and left perpendiculars to the boundary (if the boundary is not smooth).

The procedure for choosing design variables can be summarized as follows. First, from each vertex of the skeleton use perpendiculars to the boundary—rays—as variables. The vertices can be either triple points or end points. They can be vertices on the interior or exterior skeletons. Then, on each arc of the skeleton, select a few points and use the rays emanating from these points as design variables.

This strategy does not specify a particular scheme for the number and distribution of design variables along skeleton arcs. Such decisions could depend on the nature of the problem, the total number of desired design variables, the required shape resolution, the stage of the design, etc. A typical general-purpose choice might be to choose points in the middle of the arcs and then add additional middle points to each subarc recursively. Other choices could take into account the nature of the distance function on the arc and choose points at the locations where the function has an extremum value. Our results indicate that the particular scheme used to choose design variables along skeleton arcs is not important to reach the final shape. The number and locations of skeleton rays used only affects the number of design iterations it takes to reach the final shape: we exchange fewer subproblems for more work per subproblem resulting from a greater number of variables.

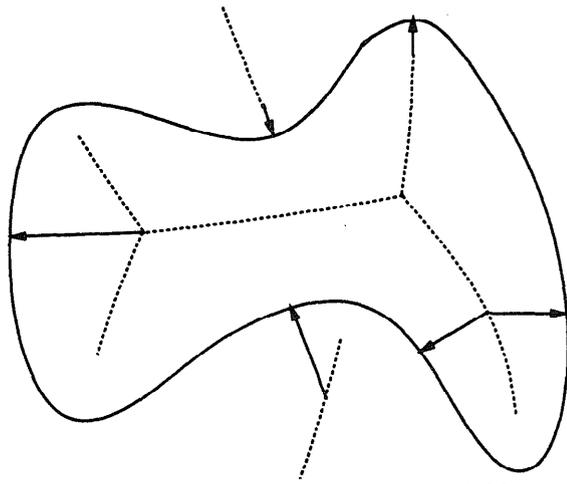


Figure 2: Skeleton Based Design Variables

Subproblem Formulation

Once the design variables are chosen, the next step is to formulate a numerical optimization problem whose solution can improve the current shape. To define a numerical optimization problem, we need to impose appropriate lower and upper bounds on design variables, and express the objective and design constraints in terms of these design variables.

An important characteristic of the skeleton is that it is the basis for prescribing natural lower and upper bounds on design variables that guarantee shape integrity. Figure 2 illustrates how shape integrity can be guaranteed if the values that the design variables can assume are bounded from above and below by the corresponding skeleton arcs. The interior skeleton is the lower bound of design variables emanating from the interior skeleton as well as the upper bound of design variables emanating from the exterior skeleton. Similarly, the exterior skeleton is the upper/lower bound of design variables emanating from the interior/exterior skeleton. Qualitatively, the interior skeleton limit guarantees that the shape does not produce “negative regions” by collapsing upon itself, while the exterior skeleton limit guarantees that the shape will not overlap, i.e., produce “double regions”.

It is typical in practical design problems that geometric shape constraints need to be imposed on the design for manufacturability, fit, or other concerns. These constraints are typically expressed globally and are applicable to the overall object. Examples of such constraints include such criteria as “no thickness shall be less than two inches”, “no radius of curvature at corners shall be less than one inch”, “object must fit in a 10x10 space” etc. Because these design constraints are often expressed as constraints on the final shape, i.e., not attached to a particular design instance, or particular design stage, we need to transform them at every design iteration into constraints on design variables. The transformation of geometric shape con-

straints into constraints on skeleton variables involves two steps. The first step is to identify the skeleton variables that model the particular feature referenced — thickness along an extrusion, curvature at a corner, etc. The second step involves expressing the criterion limits as algebraic constraints in the optimization subproblem. In §5, we describe and give examples of two classes of features that can be transformed into algebraic constraints on the design variables— thickness and overall size. Notice that the referent of a constraint changes as the design evolves and can only be properly defined with respect to the current design. The definition on another iterate in the design process may take a totally different form. Therefore, these transformations must be done at each step of the design process, as the shape evolves (giving rise to new object features), and new variables become more appropriate descriptors of the design. The classes of geometric design requirements that can be generically expressed (i.e., not with respect to *specific* design variables) and automatically transformed into algebraic constraints relating a subset of the corresponding design variables is an open question. However, it is clear that all design criteria that can be expressed as constraints on the location of boundary points can be transformed to constraints on skeleton-based design variables by simple geometric transformations.

Generating a New Shape

Once the numerical optimization is formulated, it can be solved using standard NLP techniques. Powerful numerical optimization algorithms exist and are commercially available [IMSL, 1985]. The optimization procedure results in new values for the design variables. These values are interpreted as defining an improved shape that can then be used as the starting point of another design iteration. This new starting shape is *qualitatively different* from the initial one: indentations, protrusions, and holes can appear or be removed.

For example, an interesting shape transformation happens in the limiting case when related sets of design variables assume their lower and upper bounds. For example, if two design variable rays originating on the same skeletal point of a protrusion and facing in opposite directions collapse to their common base (i.e., both take on their lower bounds), the thickness of the protrusion at that point becomes zero. If pairs of variables along this protrusion all assume their lower bounds, this is an indication that the protrusion is impeding progress towards the optimum, and should be removed. A similar effect can fill indentations: sets of variables originating from an exterior skeleton branch reach their lower bound or variables originating from interior branches reach their upper bound.

More importantly, topological changes can occur when sets of design variables reach their upper or lower bounds. Consider, for example, an internal hole. The

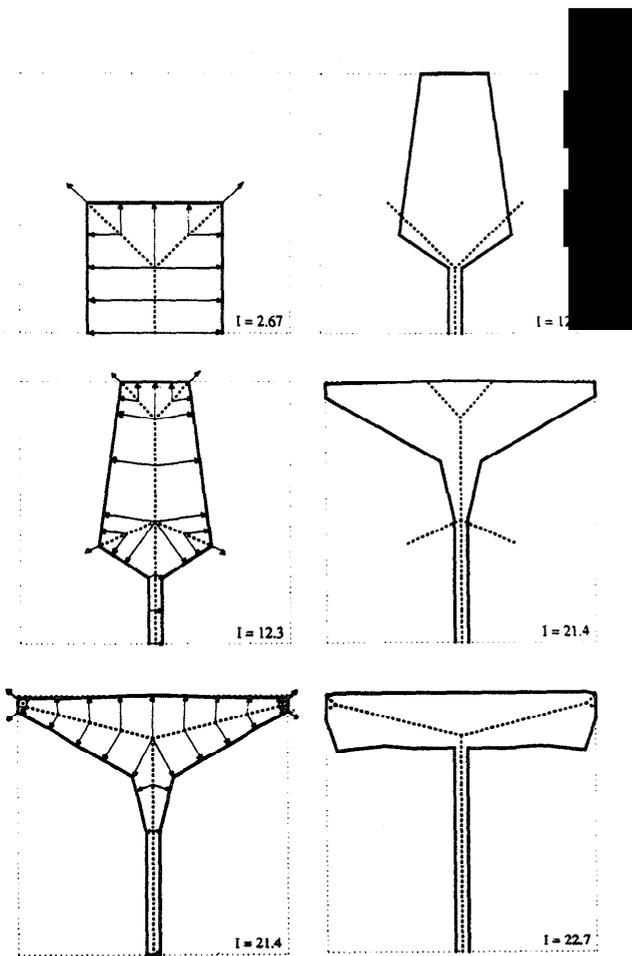


Figure 3: Design Sequence of a Wide Flange Section

hole has a corresponding (exterior) skeleton subgraph that defines the upper bounds on all design variables originating in the interior and terminating on the hole boundary. When these design variables reach their upper bounds, the hole is filled and should be removed. This new shape has a reduced connectivity and initiates a new design iteration. Holes can also appear at the end of the subproblem solution, as sets of related design variables from different skeleton arcs reach their upper bounds, and neighboring object regions join. §5 shows an example of hole formation.

Examples

This section illustrates the above methodology through two shape design examples. The first illustrates how large geometric shape changes can happen and how constraints are imposed; the second example illustrates how topological changes can occur during the design process.

Design of a Wide Flange Section

The goal of this design problem is to find the shape of a solid of fixed area which maximizes the moment

of inertia about a horizontal axis. The design is constrained to fit in an exterior rectangular box of given dimensions, and no thickness of any portion of the solid may be smaller than a given threshold.

Figure 3 shows the evolution of the design (only the upper half of the shape is shown—the lower half is symmetric). Starting from a rectangular initial shape (upper left), the skeleton is computed (dotted lines) and design variables (arrows) are chosen as described in §4. We have chosen only the interior skeleton to define design variables. Upper and lower bounds are imposed on the design variables so that the shape stays within the outer box (thin dotted line) and variables emanating from the main stem of the skeleton have lower bounds equal to half the minimum allowable thickness. Algebraic expressions defining the moment of inertia and the area of the solid can be easily written in terms of the design variables and the resulting optimization problem is solved. The upper right diagram shows the resulting shape (thick line) defined by the values of the design variables at the end of the optimization. The skeleton of the initial shape is also shown in the diagram for reference. A new iteration is then initiated and results in the shape at the middle right. Notice how the lower “protrusions” have collapsed onto the skeleton and automatically disappeared. A third design iteration gets very close to the optimal wide flange shape.

Design of a Hollow Tube Section

The goal of this problem is to design a shape of maximum polar moment of inertia constrained to fit inside a rectangular box. We initially constrain the area to a fixed value. The sequence of design iterations is displayed in Figure 4. During the first three iterations the shape moves away from the center (attempting to maximize the polar moment of inertia); when no further progress can be made, it starts “flowing” along the boundaries. The fourth iteration illustrates two important points. The first is that geometric constraints can be introduced at any point in the design process and need not be fixed throughout. For example, we allowed the total area of the shape to be increased by 50%. The dynamic introduction of design constraints is particularly important in interactive design environments. The second point is that cavities can be introduced in the design. In this case, two design variables from different portions of the skeleton reach their upper bounds (on the same exterior skeleton branch), hence surrounding and creating a hole. The last iteration further improves the design to generate a shape very close to the optimal shape. The shape has undergone a topological transformation.

Conclusions

In this paper, we have outlined a methodology for expressing and manipulating shape for design purposes. Shape design is transformed into an iterative process:

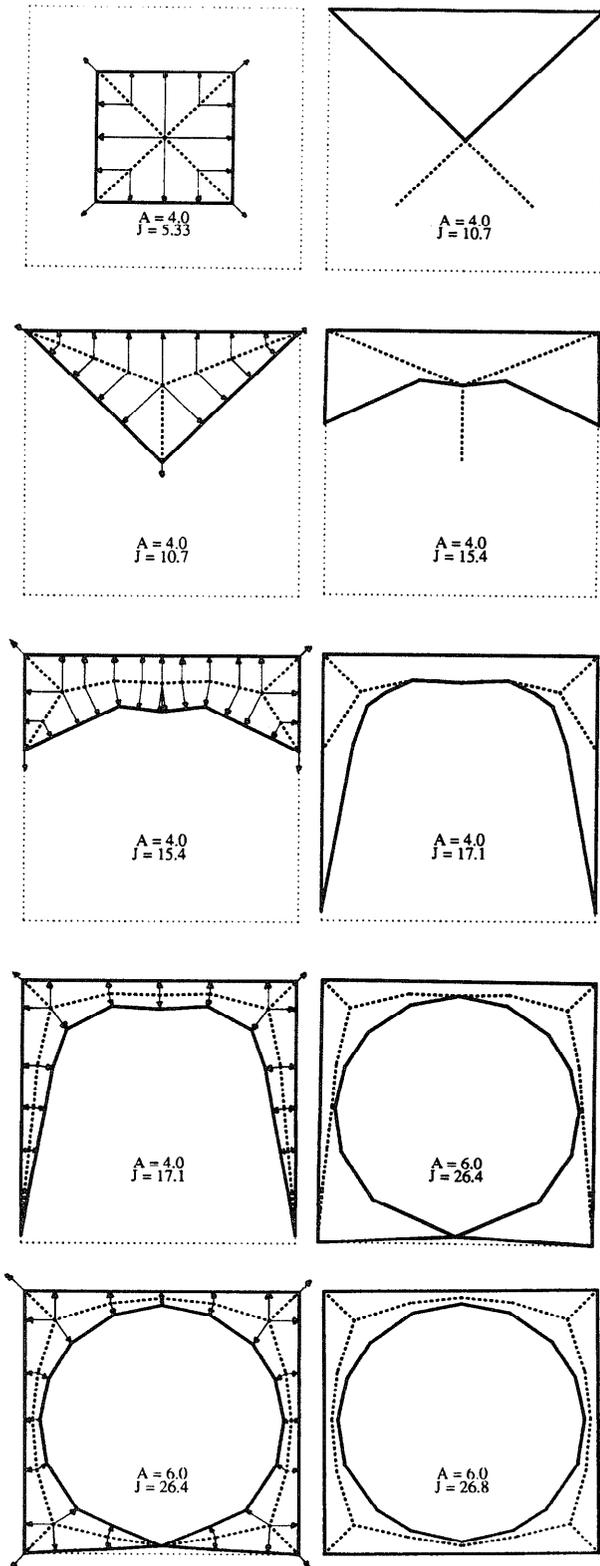


Figure 4: Design Sequence of a Hollow Tube

at each iteration a design model is generated from the skeleton, solved by numerical optimization, and the results interpreted to generate a new shape that can initiate a new design iteration. Advantages of the proposed methodology include: (1) it does not pre-impose a particular structure on the design space (as in parameter-based methods), design domain (as in grid-based methods), and composition of the design (as in knowledge-based methods); (2) conditions that guarantee shape integrity and validity can be systematically derived and imposed at each iteration; (3) the methodology is capable of generating novel geometrical and topological designs; (4) geometric constraints can be systematically incorporated in the design process; and (5) the methodology is adaptive. Qualitatively, we can interpret the design process as a sequence of motions in a design space. Given a current design description, we move as much as possible in the design space using the given representation. When no progress can be made using the current design description, a new description is generated to allow the shape to progress in another direction in the design space.

Acknowledgements. We wish to thank Steve Fenves for challenging us with the topological design problem. The support of the Ben Franklin Technology Center of Western PA (RC7086), the NSF (DDM-9009 597), and the Engineering Design Research Center—an NSF ERC at CMU—is gratefully acknowledged.

References

- B. Chandrasekaran. Design problem solving: A task analysis. *AAAI Magazine*, Winter 1990.
- R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. Wiley-Interscience Publications, 1973.
- S. Finger and S. Safier. Representing and recognizing features in mechanical designs. In *Second International Conference on Design Theory and Methodology*. Chicago, 1990.
- IMSL. *User's Manual*. Houston, Texas, 1985.
- Joskowicz and S. Addanki. From kinematics to shape: an approach to innovative design. In *AAAI-88*, 1988.
- S. Mittal and B. Falkenhainer. Dynamic constraint satisfaction. In *Proceedings Eighth AAAI Conference*, volume 1. AAAI Press, 1990.
- S. Murthy and S. Addanki. Prompt: An innovative design tool. In *AAAI-87*, 1987.
- I. Pitas and A. Venetsanopoulos. Morphological shape decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1), 1990.
- G. Turkiyyah and S. J. Fenves. Generation and interpretation of finite element models in a knowledge based environment. Technical Report R-90-188, CMU, 1990.