

Function-Based Generic Recognition for Multiple Object Categories ¹

Melanie Sutton, Louise Stark and Kevin Bowyer

Department of Computer Science and Engineering
University of South Florida
Tampa, Florida 33620 USA
sutton or stark or kwb @csee.usf.edu

Abstract. In *function-based* object recognition, an object category is represented by knowledge about object function. Function-based approaches are important because they provide a principled means of constructing *generic* recognition systems. Our work concentrates specifically on the relation between shape and function of rigid 3-D objects. Recognition of an observed shape is performed by reasoning about the function that it might serve. Previous efforts have dealt with only a single basic level object category. A number of important issues arise in extending this approach to deal with multiple basic level categories. One issue is whether the knowledge about object function can be organized into general primitive chunks that are re-usable across different categories. Another issue is how to efficiently index the knowledge base so as to avoid exhaustive testing of an object shape against each known category. In order to better explore these issues, we have implemented a system whose domain of competence is a number of different object categories within the superordinate categories furniture and dishes. The performance of this system has been evaluated on a database of over 400 shapes.

1. Introduction

Most work to date on the problem of object recognition has assumed that a geometric model is made available, a priori, for each object that the system will need to recognize. Something more is clearly needed if the goal of visual perception for “autonomous” and “real world” systems is to be achieved. Even in principle, it is simply not possible to give a real world system a geometric model of each object that it will need to recognize. A real world system must be able to deal with objects for which it has *no explicit prior shape model*. A function-based model does not specify any explicit geometric or structural plan. Instead, an object category is defined in terms of knowledge about what is necessary in order to function as an instance of the category. The underlying representation is in the form of primitive chunks of knowledge about shape, physics and causation that may be used as primitives in building definitions of required function. The *form and function* concept is intuitively appealing and has been discussed by a number of researchers in AI and computer vision [1, 3, 5, 7, 8, 12, 13]. Brady and co-workers implemented a system which performs a type of function-based reasoning based on a 2-D outline of the object in its typical pose [4], and also outlined goals for a more ambitious system [3]. Di Manzo *et al.* [5] described a design for a system to

¹This research was supported by AFOSR grant F49620-92-J-0223, NSF grants IRI-9120895 and CDA-91-00898 (Research Experiences for Undergraduates), and a NASA Florida Space Grant Consortium graduate fellowship.

recognize shapes of chairs in 3-D. Vaina and Jaulent [12] investigated the use of both conceptual and structural object description for functional recognition. Kitahashi *et al.* [7] brings the user, or “functant,” into the representation to help define prototypical shapes. Stark and Bowyer reported on a system that analyzes 3-D shapes in arbitrary initial orientation to determine if they could fulfill the function of a chair [10] and later described an extension of the system for the superordinate category furniture [9]. The work described in this paper explores the issues that must be addressed in generalizing the function-based approach to deal with multiple basic-level categories. Our ideas have been realized in an implemented system and experimental results are reported. Section 2 outlines the structure and operation of the **GRUFF-3** system (**G**eneric **R**epresentation **U**sing **F**orm and **F**unction). The complexity analysis in section 3 shows that recognition is at worst case polynomial time. Section 4 describes an evaluation of the system’s competence in recognition. Section 5 presents a function-based category indexing scheme that leads to greater efficiency. Finally, section 6 summarizes the current state of the project and suggests directions for continued research.

2. The Function-Based Recognition System

This section details the operation of a function-based recognition system for the superordinate category *furniture*, comprised of the basic level categories *chair*, *table*, *bench*, *bed* and *bookshelf*, and the superordinate category *dishes*, comprised of the basic level categories *cup/glass*, *bowl*, *plate*, *pot/pan*, and *pitcher*. We first describe the six *knowledge primitives* that are the basis of the function-based definitions of the object categories. Next we show how a sequence of invocations of the primitives is used to define a *functional property* that is specific to some particular category. We then present the overall definition of the basic level categories and their subordinate categories. The analysis of an example shape is traced to provide a better understanding of how recognition occurs. During the recognition process an *association measure* is accumulated to reflect the level of confidence that the system holds in support of the object belonging to the hypothesized category. This section ends with a description of how this association measure is accrued throughout the evaluation process.

2.1. The “Knowledge Primitives”

A knowledge primitive can be thought of as a chunk of parameterized procedural knowledge that implements some basic concept about shape, physics or causation. Each knowledge primitive takes some (specified portions of a) 3-D shape as its input, along with parameter values for the primitive, and returns an *evaluation measure* between zero and one. (Individual evaluation measures from a series of primitive invocations are accumulated into an overall association measure for the functionality of the shape.) The six knowledge primitives used by the system are as follows.

- **relative_orientation (normal_one, normal_two, range_parameters)**

This primitive determines if the angle between the normals for two surfaces (*normal_one* and *normal_two*) falls within a desired range. There are four *range_parameters*: *least*, *low_ideal*, *high_ideal*, and *greatest* (see Figure 1 a). These parameters are used to calculate a value for the evaluation measure based on where the relative orientation falls

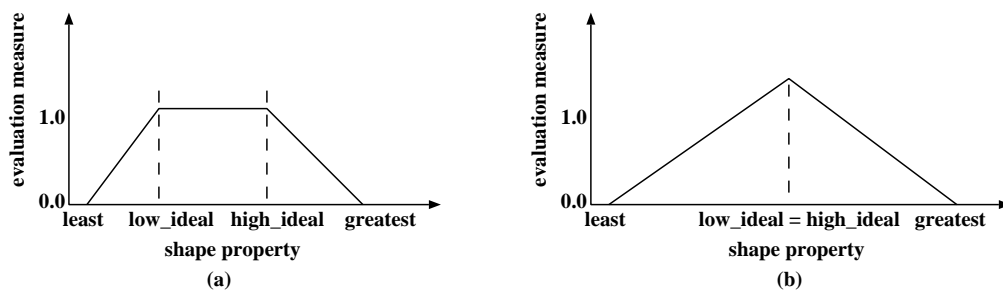


Figure 1. Using Shape Properties to Calculate Evaluation Measure

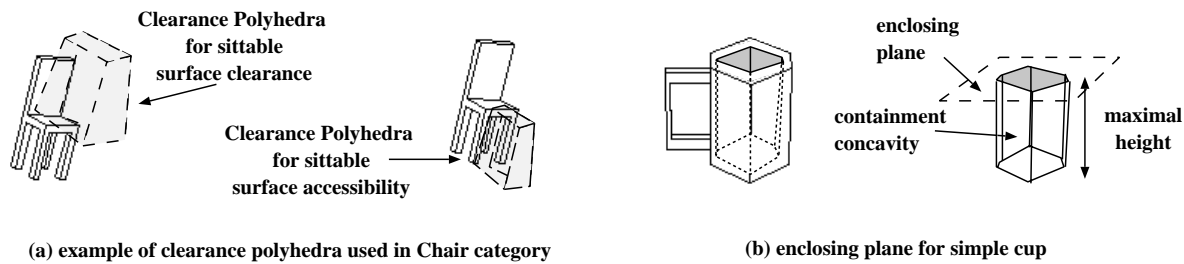


Figure 2. Examples of Knowledge Primitive Tests Performed

within the specified range. Any value between `low_ideal` and `high_ideal` results in a measure of one. Values outside this range but between `least` and `greatest` fall off linearly to zero. (`low_ideal` and `high_ideal` could be set to the same value, resulting in a “triangle rule” for calculating the measure value (see Figure 1 b).) This primitive can also be used to calculate the transformation which would position the shape in the desired orientation. For example, a common operation is to find the transformation of the shape which would orient a specified surface parallel to the support plane. When relative orientation is used in this manner, a measure of one is always returned.

- **dimensions** (`shape_element`, `dimension_type`, `range_parameters`)

This primitive can be used to determine, for example, if the width or depth of a surface lies within a specified range. The evaluation measure is calculated using four parameters, in much the same manner as for relative orientation.

- **proximity** (`shape_element_one`, `shape_element_two`, `range_parameters`)

This primitive can be used to check qualitative relations between `shape_elements`, such as *above*, *below* and *close to*. Again, the measure is calculated as described above.

- **clearance** (`object_description`, `clearance_volume`)

This primitive can be used to check that there is a specified volume of unobstructed free space in a particular location relative to a particular part of the shape. The volume is specified by a *clearance polyhedron* (see Figure 2(a)), implemented as a rectangular

volume of space specified by a set of six faces and eight vertices. This primitive is generally used for the purpose of checking the accessibility of a potential functional element of an object shape. The evaluation measure is one if the volume specified is unobstructed, or zero if it is obstructed.

- **stability (shape, orientation, applied_force)**

This primitive can be used to check that a given shape is stable when placed on a supporting plane in a given orientation and with a (possibly zero) force applied. It is assumed that the object has homogeneous density, so that the center of mass may be calculated directly from the shape description. This primitive returns an evaluation measure of one if the shape is stable in the specified orientation, or zero if it is not.

- **enclosure (concavity, orientation, enclosing_plane)**

This primitive is used to determine if there exists a concavity in the shape which can be “enclosed” by a single plane introduced parallel to the support plane with the shape in a given orientation. For a cup, in its normal upright orientation, a plane which encloses the concavity can be introduced parallel to the support plane (see Figure 2(b)).

Actually, an infinite number of planes could be introduced at different *levels*, each enclosing a different volume. We are interested in finding the maximal volume that can be enclosed for a given concavity. If the invocation finds an enclosable concavity, then it returns a measure of one along with a description of the concavity. If the invocation does not find an enclosable concavity, then it returns a measure of zero.

All of the system’s reasoning about 3-D shapes is constructed out of these six building blocks. The exact set of “primitives” is not so important— it would be easy to create a different set that allowed the construction of a system with equivalent capabilities. However, one of the underlying assumptions of our work is that a “small” number of primitives suffices to define a “large” number of categories, and that the number of primitives required grows “slowly” as a function of the number of categories. This represents one of the fundamental advantages of a function-based system over a more traditional model-based system; that is, that a small amount of a priori knowledge can establish a large domain of competence. Primitives 1 through 5 above were used to construct function-based recognition systems for the single category chair [10] and for a collection of basic level categories in the superordinate category furniture [9]. One of the contributions of this work is to demonstrate that only a single additional primitive suffices to allow a much larger domain.

2.2. Functional Properties Defined By Knowledge Primitives

An object category is defined in terms of its functional plan, specified as a set of *functional properties*. Each functional property is defined by a sequence of invocations of the knowledge primitives, the basic building blocks in the representation. Because functional properties tend to be specific to particular categories, there are far too many to cover each in detail here. Considering one in detail should give the flavor of the approach. The subcategory *king_size_bed* has the following definition:

$$\textit{king_size_bed} ::= \textit{provides_king_size_sleeping_surface} + \textit{provides_stability}$$

indicating that it is defined as a shape placed in an orientation that satisfies the conjunction of the two properties *provides_king_size_sleeping_surface* and *provides_stability*. Note that since the system reasons only on the basis of static 3-D shape, we do not consider properties such as “softness” that may be very important in a complete function-based definition. Sensing of such properties would obviously require more than just static shape as input.

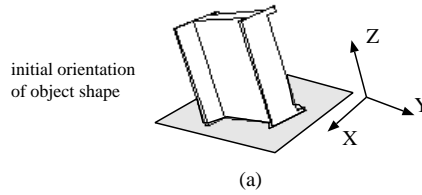
A pseudo-code description of the particular sequence of primitive invocations that constitutes the definition of *provides_king_size_sleeping_surface* is given in Figure 3(b), along with the actual measures returned during the evaluation. The order of the primitives is chosen to help eliminate surfaces from consideration as early as possible using computationally simple tests. For example, the first invocation of dimensions simply tests to see if a surface is within the proper range of area. This is a very simple test, since the area of each face is calculated prior to hypothesis generation. However, the area test does not ensure that the surface is of the proper dimensions in width and depth. These dimension tests require slightly more processing and are only invoked for surfaces which have already passed the area test. For the shape in Figure 3, only two surfaces survive the initial area test, and these are the only surfaces tested by the subsequent primitives of the functional property. Throughout the analysis, the evaluation measures returned from each primitive are combined to reflect an overall *association measure*. The calculation of the association measure is covered in a later section. At the end of the *provides_king_size_sleeping_surface* evaluation, there are two candidate surfaces, each with a different orientation for the shape and a different association measure.

To complete evaluation of the subcategory king size bed, stability has to be confirmed (see Figure 3(c)). The candidate surface corresponding to the bottom of the bed was eliminated by the *provides_stability* functional property, due to the shape not being stable when this surface is oriented parallel to the support plane. Thus there is only one orientation which allows the necessary function, and the overall association measure of this shape as a king size bed is 0.70 (see Figure 3(d)).

2.3. Basic Level, Subordinate and Superordinate Categories

The system’s knowledge about its domain of competence can be considered as a frame system organized into a tree structure. Each node of the tree is represented by a frame that has four fields: *Name*, *Type*, *Realized_By*, and *Functional_Plans*. Nodes are of one of four Types: *Superordinate*, *Category*, *Subcategory* or *Function_Label*. Each *Superordinate* frame has associated with it one or more *Category* frames which are linked to the *Functional_Plans* field.

Specifically in our current system, the knowledge base consists of the superordinate category *furniture*, with five immediate children representing the basic level categories *chair*, *table*, *bench*, *bed* and *bookshelf* and the superordinate category *dishes* with five immediate children representing the basic level categories *cup/glass*, *bowl*, *plate*, *pot/pan*, and *pitcher* (see Figure 4). At this level, the branches of the tree represent non-exclusive disjunction. That is, any one input shape may possibly be recognized as belonging to more than one category. Of course, there may be a different preferred orientation of the shape and a different association measure for each category. Each basic level category acts as a root node for the functional plans defined within that category. Figure 5 depicts the defining structures for two of the basic level categories



knowledge primitives for provides_king_size_sleeping_surface

	resulting evaluation measure at each step			
dimensions (surface, area, least_1, low_ideal_1, high_ideal_1, greatest_1) /* checks for the appropriate total sleeping surface area */		primitive=0.73 aggregate=0.73		primitive=0.97 aggregate=0.97
dimensions (surface, width, least_2, low_ideal_2, high_ideal_2, greatest_2) /* checks for the appropriate width of the sleeping surface area */		primitive=1.0 aggregate=0.73		primitive=1.0 aggregate=0.97
dimensions (surface, depth, least_3, low_ideal_3, high_ideal_3, greatest_3) /* check for the appropriate depth of the sleeping surface area */		primitive=1.0 aggregate=0.73		primitive=1.0 aggregate=0.97
dimensions (surface, contiguous_area, least_4, low_ideal_4, high_ideal_4, greatest_4) /* checks that the possibly aggregate faces of the surface have appropriate contiguity */		primitive=1.0 aggregate=0.73		primitive=1.0 aggregate=0.97
dimensions (surface, height, least_5, low_ideal_5, high_ideal_5, greatest_5) /* checks for the surface being within the appropriate height range */		primitive=0.96 aggregate=0.70		primitive=0.58 aggregate=0.56
relative_orientation (surface, ground, least_6, low_ideal_6, high_ideal_6, greatest_6) /* finds the appropriate orientation of the surface to position parallel to the ground */		primitive=1.0 aggregate=0.70		primitive=1.0 aggregate=0.56
clearance (object, 3d_volume_above_sleeping_surface) /* checks for appropriate clearance above the sleeping surface */		primitive=1.0 aggregate=0.70		primitive=1.0 aggregate=0.56
clearance (object, 3d_volume_to_left_side_of_sleeping_surface) /*checks for appropriate accessibility from the left side */		primitive=1.0 aggregate=0.70		primitive=1.0 aggregate=0.56
clearance (object, 3d_volume_to_right_side_of_sleeping_surface) /* checks for appropriate accessibility from the right side */		primitive=1.0 aggregate=0.70		primitive=1.0 aggregate=0.56

(b)

knowledge primitives for provides_stable_support

	resulting evaluation measure at each step			
stability (object_shape, given_orientation, self_stable) /* checks for self stability of object with no external forces except gravity */		primitive=1.0 aggregate=0.70		primitive=0.0 aggregate=0.0
stability (object_shape, given_orientation, external_forces_on_sleeping_surface) /* checks for stability of object with external forces applied to sleeping surface */		primitive=1.0 aggregate=0.70		

(c)

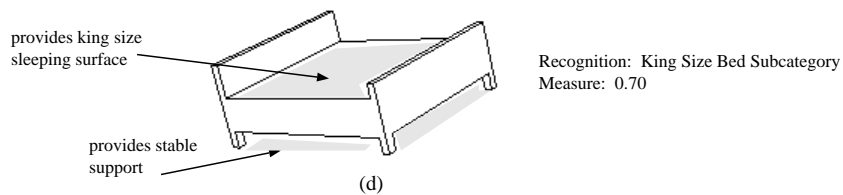


Figure 3. Knowledge Primitives Defining King Size Bed Subcategory

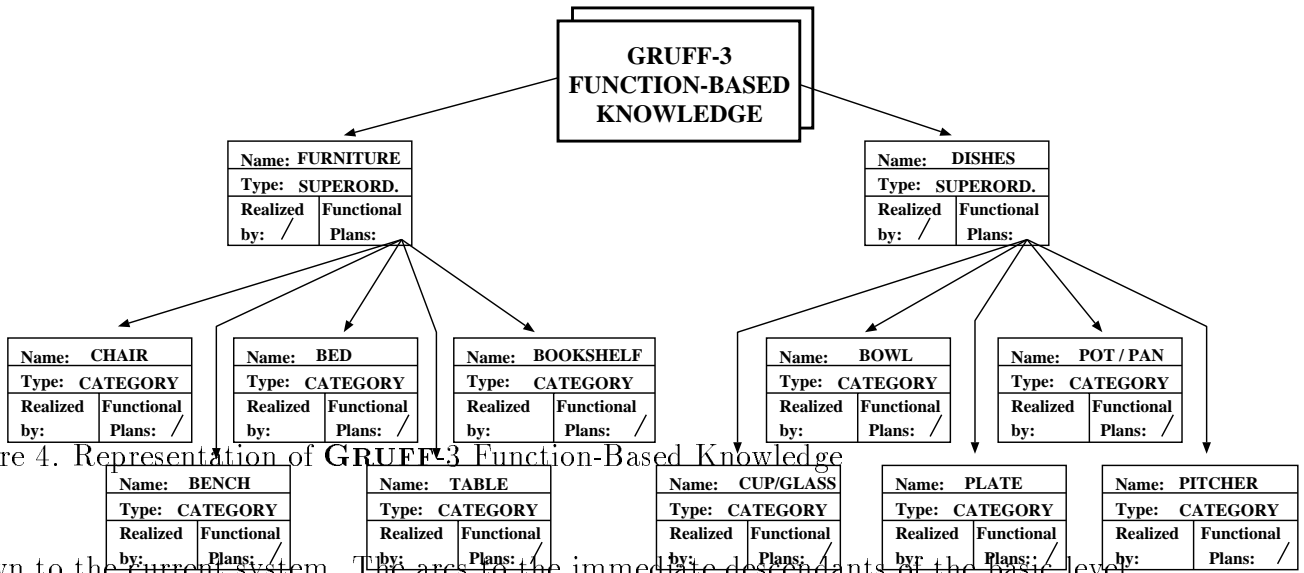
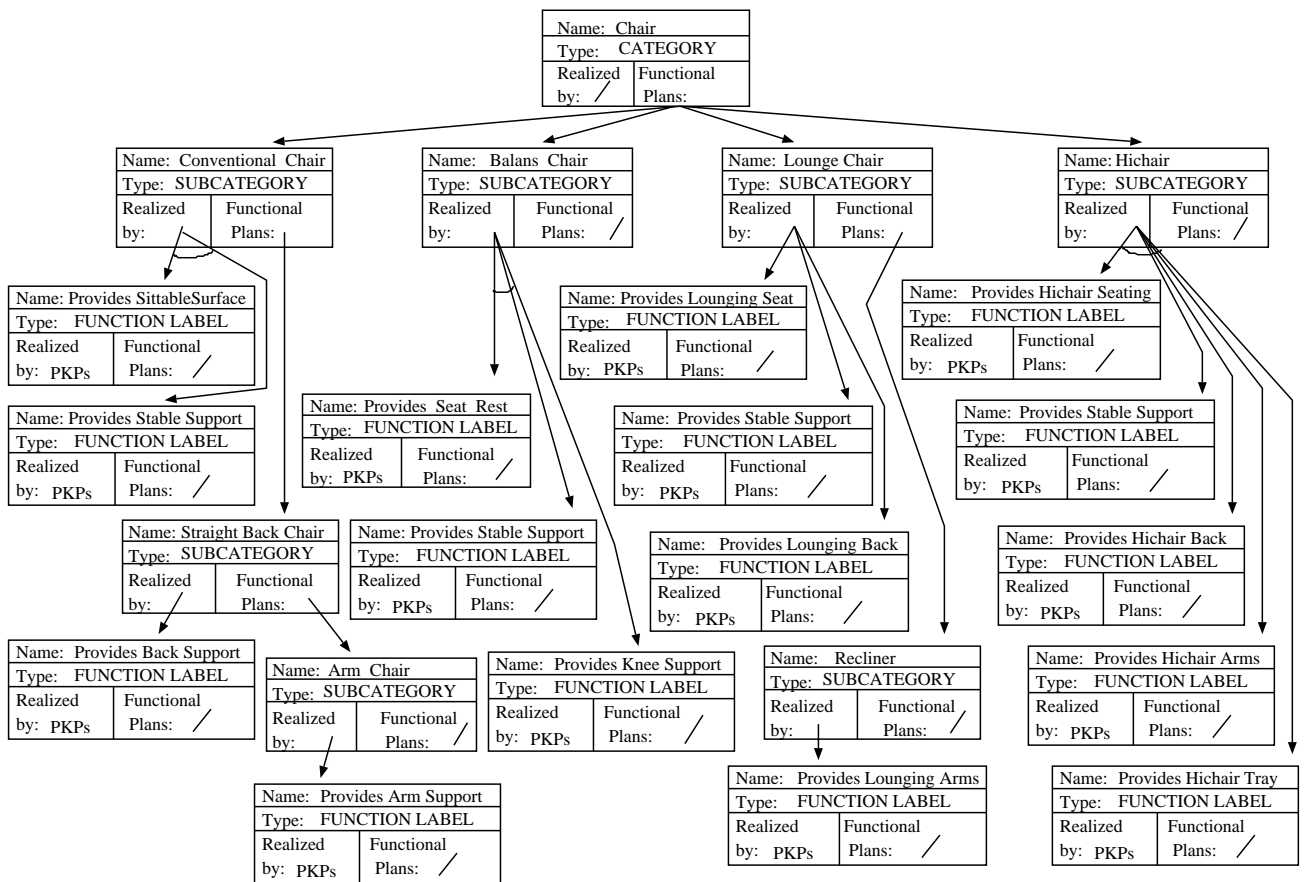


Figure 4. Representation of GRUFF-3 Function-Based Knowledge

known to the current system. The arcs to the immediate descendants of the basic level category nodes also represent a type of disjunction. The different subcategory nodes at this level represent functional variations of the parent basic level category that are important enough to be separately named, such as “lounge chair” and “Balans chair.” Below any immediate descendant of a basic level category, the nature of the representation changes somewhat. The essential functional plan is fixed, and is represented by the conjunction of the functional properties listed in the *Realized_By* field of the subcategory node. However, there may be different levels of refinement or elaboration of the essential functional plan that are important enough to be separately named. These would be listed in the *Functional_Plans* field of the same subcategory node, and they represent a required conjunction of additional functional properties. Thus a *conventional chair* is anything that *provides_sittable_surface* and *provides_stable_support*. A *straight back chair* adds the requirement of *provides_back_support* to the requirements for *conventional chair*. This distinguished naming of specializations of the essential functional plan may be carried more than one level deep. For example, *arm chair* adds the requirement *provides_arm_support* to the requirements of *straight back chair*.

2.4. Flow of Control in Interpretation

The input shape is defined as a set of faces, with each face defined by an ordered set of vertices. The first step in the recognition process is evaluation of the input shape to tabulate some information that will be used throughout the recognition process. This information includes the volume of the shape, center of mass of the shape, convex hull of the shape, the volume and center of mass of the convex hull, a list of convex 3-D sub-volumes of the shape, a list of groups of faces which are essentially coplanar and close to each other so that they can act as a surface for functional purposes, and all groups of faces that define enclosable concavities of the shape. (We will use the term *face* when referring specifically to one face of the input shape model and the term



(Figure reprinted from L. Stark and K. Bowyer, Achieving Generalized Object Recognition Through Reasoning About Association of Function to Structure, IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 13, pages 1097-1104. Copyright, 1991, IEEE.)

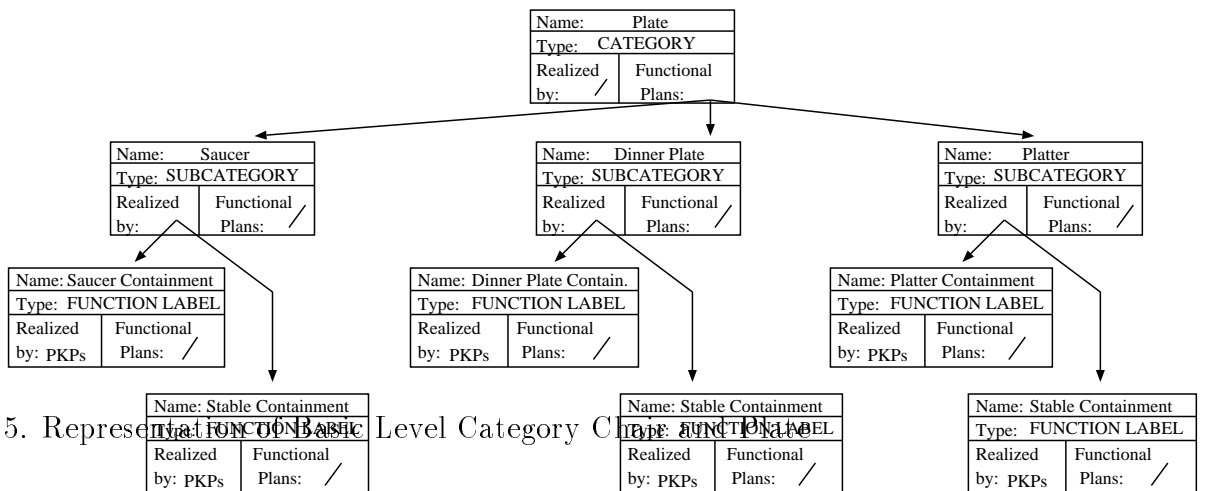


Figure 5. Representation of Basic Level Category Clusters

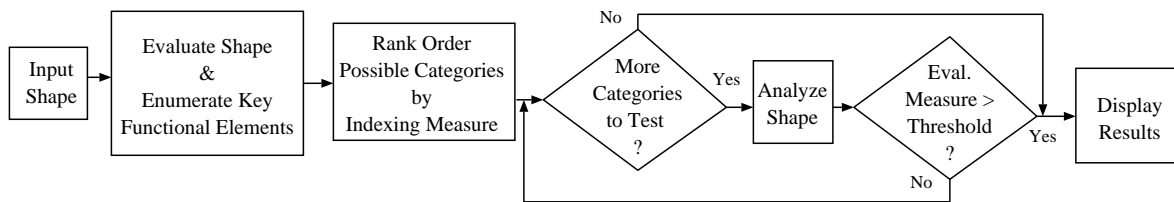


Figure 6. Flow of Control in Interpreting an Object Shape

surface or *virtual surface* when referring to one or a group of faces that can act together as a functional surface.)

In the next stage, the system hypothesizes an ordered list of categories to use in evaluating the shape. This processing begins conceptually at the root of the category definition tree and flows toward the leaves. The ordering of categories is done by comparing indices computed from the shape in the first step with index ranges kept at each category node and computing an *indexing measure* for each category. The index ranges are broad, due to the varied shapes possible within each category, but help to eliminate “impossible” shapes. For example, these heuristics help eliminate evaluation of shapes the size of a conventional chair for possibly fulfilling the functional requirements of a dish or even a bed. A more detailed description of the indexing scheme is given later.

The categorization performed by the system identifies functional elements of a shape by associating them with their proper *function label* (e.g. *provides table surface*, or *provides saucer containment*). Once a category is hypothesized, that subtree of the category definition tree is used as a control structure in the analysis of the shape. If a basic level category has multiple subcategories, indexing is done again at this level.

For each hypothesized subcategory, there is a subtree that defines the *functional plan* of that subcategory. The traversal of each subtree proceeds in a depth-first manner. For example, analysis of the shape depicted in Figure 3 begins with confirming some surface(s) that could satisfy *provides king size sleeping surface*, and then analyzes whether the shape could *provide stable support* for a candidate sleeping surface. The purpose of the next section is to provide a better understanding of how evidence is gathered by the system and how an overall association measure is calculated.

2.5. Accrual of Association Measure

The system attempts to categorize an input shape as belonging to some subcategory, with some cumulative *association measure*. As noted earlier, evidence is initially gathered directly from the knowledge primitives which define each required functional property. This evidence is combined to reflect an overall measure of how well the functional properties for the category are fulfilled. The aggregation calculi used should behave so that if a required knowledge primitive invocation returns an evaluation measure lower than the current association measure, it lowers the overall evidence for that subcategory. Thus the association measure for the minimum level of membership in a category is dominated by the weakest of the set of required primitives.

When the subcategory being investigated is a refinement of its parent subcategory, in the way that straight back chair is a refinement of conventional chair, the evidence gathered at the parent subcategory node must be combined with the present evidence. In this case the behavior of the aggregation calculi should be to increase the association measure at the parent subcategory node by some factor associated to the evidence gathered at the present node.

This establishes two types of behavior for the aggregation calculi, depending upon the level of processing. An investigation of different calculi was performed using the criteria defined above [11]. It was found that a pair of T-norm and T-conorm functions provided the best performance.

2.5.1. Distinguishing Levels of Processing

An approach has been adopted that maintains an accrued measure on the interval [0,1]. Evidence gathered is combined in either a conjunctive or disjunctive manner, depending upon the *level* of processing. The first level of processing reflects how well a specific portion of the structure fulfills required functions (combining evidence at the knowledge primitive level). For example, evidence is gathered to reflect how well a surface can act as a sittable surface of a chair and whether the shape is stable when that surface is oriented parallel to the support plane. The second level combines evidence relating the interaction of different functions; for example, how well the sittable surface and back support relate to perform as a straight back chair (functional plan level).

2.5.2. Combination of Evaluation Measures from Knowledge Primitives

At the knowledge primitive level, *evaluation measures* are returned and combined to obtain an *association measure* for each functional requirement. All requirements must meet some threshold measure in order to consider the requirements of the function label as being fulfilled. Representative instances of three families of aggregation calculi [2] were evaluated. The operation that gave the best performance is the T-norm or conjunctive operation:

$$T(a, b) = ab$$

Assuming that the association measure calculated to reflect a specific functional requirement is initialized to one at the beginning of evaluation, the accrual of evidence is accomplished as:

$$new_association_measure = present_association_measure * evaluation_measure$$

where the *evaluation measure* is the value returned by each primitive invocation. Further processing leads us to the level at which we combine information concerning specialization properties (e.g. a straight back chair is a specialization of a conventional chair).

2.5.3. Combining Evidence across Functional Plans

Evaluation of the input shape follows a depth first traversal of the control structure (see Figure 5). For example, to establish that the shape belongs in the subcategory straight back chair it must first be established that the shape meets the functional requirements of the conventional chair. An aggregate measure is calculated by combining evidence of how well each of the functional requirements (*provides sittable surface* and *provides stable support*) was met. Processing then continues with the subcategory node straight back chair. An association measure local to the subcategory node is calculated for the single functional requirement specified for this node (*provides back support*). Now the evidence must be combined across functional plan levels. The evidence gathered in support of conventional chair is aggregated with the evidence gathered in support of straight back chair using the T-conorm operator [2]:

$$S(a, b) = a + b - ab$$

giving a final association measure to assign to the straight back chair subcategory. Processing continues down the control structure until the functional requirements cannot be met or the graph cannot be traversed downward any further. Each subcategory node which has been visited holds an association measure on the interval [0,1]. Each subcategory node which has not been visited due to lack of evidence at a parent subcategory node holds an association measure of zero. Recognition is accomplished by reporting the nodes with the maximum association measures above a set threshold value.

3. Complexity of Shape Evaluation

In the current implementation, the input shape is assumed to be polyhedral (either because the object is truly polyhedral or because the shape acquisition mechanism produces a polyhedral approximation). A polyhedral shape with N faces is $\theta(N)$ in the number of faces, edges and vertices. An analysis of the time complexity of system execution can be broken into two main stages: pre-evaluation of the shape followed by function-based interpretation.

The first stage of the processing involves evaluation of the overall shape and formation of a list of all potential *functional elements*. Specifically, we perform the following three operations.

One, compute the volume of the shape. An algorithm due to Edelsbrunner [6] is used to divide the shape volume, and in fact all of 3-D space, into a unique set of $\theta(N^3)$ convex volumes, and then computing the sum of the volumes ($\theta(N^2)$ step) of those cells which correspond to the shape. For a polyhedral shape with N faces, computation of the volume is therefore $\theta(N^5)$.

Two, compute the center of mass of the shape, assuming that the object has uniform density. This is an $\theta(N^3)$ step.

Three, form a list of all the potential functional surfaces (individual faces of the shape and “virtual surfaces” formed by collections of faces which are “essentially coplanar”), all the enclosable concavities of the shape, and all the convex cells which make up the shape. The list of “virtual surfaces” is formed using a three-step heuristic. First, the N faces are sorted based on area (a $\theta(N \times \log N)$ step). Then this list is divided into non-overlapping sublists of “essentially parallel” faces (a $\theta(N)$ step). Finally, each

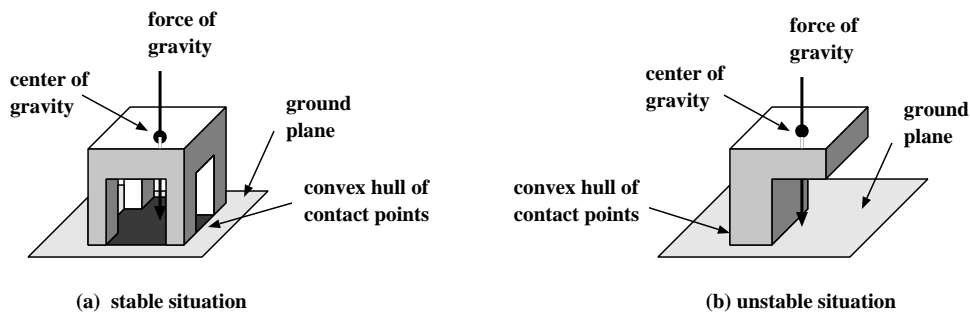


Figure 7. Testing Stability of Object Shape in Given Orientation

sublist is further divided into non-overlapping sublists of “essentially coplanar” faces (a $\theta(N \times \log N)$ step). The result is a list of at most $\theta(N)$ surfaces (object faces and groups of faces that can act as a functional surface). The list of concavities is formed through a two step process. The first step involves finding all surfaces of the object that are not part of the convex hull (concavity surfaces). This is a $\theta(N^2)$ step. There are at most $\theta(N)$ concavity surfaces, each of which can be part of a single concavity. The next step is to form lists of non-overlapping surfaces, one for each concavity ($\theta(N^2)$ step).

The second stage of the processing is to interpret the shape according to the functional plan of the hypothesized categories. Each category is represented by a sequence of required functional properties, and each functional property is realized by a sequence of invocations of the knowledge primitives (KPs). The complexity of each KP is as follows.

- *relative orientation* - Finding all pairs of surfaces in a list of N surfaces which satisfy a particular relative orientation relationship is $\theta(N^2)$, there being $\theta(N^2)$ pairs in the list and the time for a surface normal evaluation and a difference between normals both being constants.

- *dimensions* - Each of the N edges of a surface is aligned with the positive \mathbf{X} axis in an \mathbf{X} - \mathbf{Y} plane. In each alignment, each of the N vertices is examined and a record of the min/max \mathbf{X} value and min/max \mathbf{Y} kept. Thus finding all orientations of a given N -edge surface for which the extent of the surface falls within a given width and depth range is at worst $\theta(N^2)$.

- *stability* - Consider a given orientation of a shape to be checked for stability. Finding all the vertices at the minimum \mathbf{Z} value (and so potentially in contact with the support plane) is a $\theta(N)$ step. If there are three or more non-collinear points of contact, then the convex hull of the contact points is found (a $\theta(N \times \log N)$ step). Then, based on gravity resulting in a downward force, a vector is projected from the center of mass of the shape perpendicular to the supporting plane (a constant time step). If the vector projects to a point inside the convex hull, then the object is stable in that orientation and if it does not then the object is not stable in that orientation. Figure 7 depicts a stable and unstable situation. (Checking this is a $\theta(N)$ step). Thus, checking the stability of a particular hypothesized orientation of an object with N faces is a $\theta(N \times \log N)$ process. In some cases, it is necessary to re-apply the stability test to an orientation already found to be stable, with forces applied to some points on the object. Any constant number of such checks still results in a $\theta(N \times \log N)$ process.

If a shape is not stable in the given orientation, it may be necessary to generate all possible stable orientations. This can be done by considering all triplets of vertices (a $\theta(N^3)$ process). Hence, checking for all possible stable orientations is at worst a $\theta(N^4 \times \log N)$ operation.

- *proximity* - Checking a list of N surfaces, where each surface has at most M vertices, to find all those surfaces which fall within the specified proximity to a given surface is a $\theta(N \times M)$ process.
- *clearance* - The check is made by testing each of the six faces of the clearance polyhedra to make sure that it does not intersect any of the N faces of the object, resulting in a $\theta(N)$ process.
- *enclosure* - Each concavity is defined as a set of faces, with each face defined as a sequence of vertices ($\theta(N)$ vertices). For a given orientation, a plane is introduced at the maximal height of the concavity (the level of the vertex which is farthest from the support plane). This plane is intersected with the concavity (a $\theta(N^2)$ process). It must be established that the intersection of the plane that is introduced with the concavity forms one or more closed faces and that by combining these enclosing faces to the concavity description a valid closed solid is formed. Testing for closed faces is a $\theta(N^2)$ process. Testing for a valid closed concavity description is also $\theta(N^2)$. If the plane introduced at the *maximal* vertex level does not enclose the concavity, a new plane is introduced at the height of the next highest vertex. Therefore, the $\theta(N^2)$ process described above could be repeated up to $\theta(N)$ times giving a $\theta(N^3)$ process overall. Evaluation of an input shape involves a sequence of operations. The maximum length of the sequence is fixed by the category definition tree and is independent of the object shape. Thus the complexity is that of the maximum complexity step, the $\theta(N^5)$ step of computing the volume.

4. Evaluation of System Competence

Appropriately evaluating the competence of a function-based recognition system is not simple. The difficulty is inherent to the nature of the approach— it is entirely reasonable (in some cases, even desirable) for a shape to belong to more than one category. For example, the shape depicted in Figure 8(a) is about equally functional as either a simple chair (a stool) or as an end table. In this case, there does not seem to be a strong preference for one interpretation over the other. Similarly, the shape depicted in Figure 8(b) can function about equally well as either a bookshelf or as a work table. However, in this case most people would have a strong preference for naming the object as a bookshelf. This preference may be driven by considerations that are not purely function-based; the presence of the regularly-spaced parallel surfaces may be considered so “non-accidental” that it must be related to what the object is “meant to be” (as opposed to what it can function as).

Thus at one level, the competence of a function-based recognition system might be measured by how correctly it can determine all of the possible functions of an object. At a more detailed level, the competence of a system might be measured by how correctly it can determine the rank ordering of the possible functions. The problem in either case is how to judge the correctness of the system’s answers. The standard for comparison must be some human interpretation of the same shape. However, the human interpretation of a shape is based on a much larger domain of competence and possibly

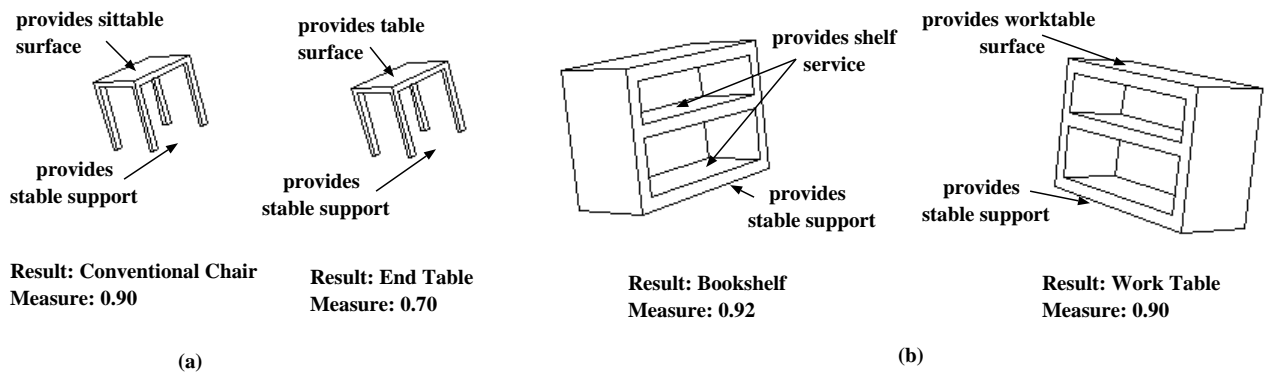


Figure 8. Examples of Shapes That Could Function in Different Ways.

also on context and cues for what a shape is meant to be. Thus it is clear that the results of such a comparison must be considered carefully.

4.1. Is/Is Not Competence

We created a database of 418 shapes to use in evaluating the competence of the system. A subset of these shapes is depicted in Figures 9 - 11. One of the authors considered each shape from the point of view of “could it function as an X?”. Thus each shape was assigned to one or more of the ten categories known to the system or to “no known category.” If the shape was assigned to more than one category, then the categories were ordered from most appropriate to least appropriate. Each shape was also analyzed by the system. No indexing strategy was used in this analysis, so that each shape was evaluated for each category known to the system. If the evaluation measure fell below 0.4 for a given category, then the shape was considered as not capable of providing the specified function. The categories resulting in a measure of greater than 0.4 were ranked by their measure. (The value of 0.4 was chosen empirically as a threshold which gave good overall performance.)

The first level of comparison is how often the human and the system agree that a shape could provide the function of a given category, ignoring the relative ranking of categories. We refer to this as the *is / is not competence* of the system. The results of the comparison at this level are summarized in Table 1.

On the whole, the system agreed with the human interpretation approximately 85% of the time. The largest discrepancy was the 55% agreement in category table. The 14 discrepancies for this entry are depicted in Figure 9. To gain a better understanding of system performance, these discrepancies will be explained in greater detail. A more thorough breakdown of system performance will be presented later.

The shapes can be divided into subsets according to the reason for disagreement. The shape in Figure 9 (a) failed due to not being able to find a surface large enough (according to system constraints) that could function as a table. The shapes in Figure 9 (b-e) initially identified surfaces which fell in the end table size range, but later failed due to being too tall (according to system constraints) to function properly as end tables. Object (f), which has a small lip running along one side of the top surface, failed due to not being able to establish clearance. We have included in the functional

Table 1
 Evaluation of “Is / Is Not” Competence in Function-Based Recognition.

	HUMAN INTERPRETATION	SYSTEM INTERPRETATION	PERCENT AGREEMENT
CHAIR (is/is not)	66 / 352	56 / 331	85 / 94
TABLE (is/is not)	31 / 387	17 / 378	55 / 98
BENCH (is/is not)	31 / 387	27 / 366	87 / 95
BOOKSHELF (is/is not)	17 / 401	15 / 399	88 / 99
BED (is/is not)	27 / 391	25 / 389	93 / 99
CUP/GLASS (is/is not)	35 / 383	35 / 383	100 / 100
BOWL (is/is not)	44 / 374	39 / 360	89 / 96
PLATE (is/is not)	13 / 405	13 / 403	100 / 99
POT/PAN (is/is not)	26 / 392	25 / 392	96 / 100
PITCHER (is/is not)	2 / 416	2 / 412	100 / 99
NO KNOWN OBJECT	152	136	89

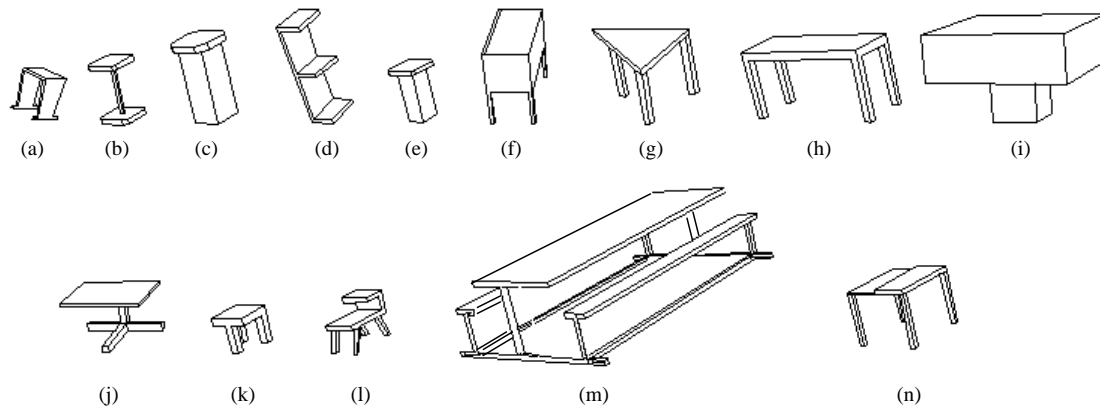


Figure 9. Disagreement between System and Human Interpretation of Tables

properties of clearance and accessibility the requirement that the table surface be completely clear and accessible from all sides. Objects (g) and (h) did not pass because their measures (0.24 and 0.37, respectively) did not meet the threshold of 0.4. For shapes (a-h), recognition as tables could be accomplished by simply adjusting our functional property constraint values or possibly our threshold value.

Objects (i-k) were actually recognized as tables, but were flagged as potentially unstable if a force is exerted on the table surface near the edges. Objects (l-n) failed due to the lack of knowledge by the system of the concepts of offset or “tiered” table surfaces and the concept of picnic table. The system has no explicit functional plan for a picnic table, and so object (m) does not pass as a table since the table surface would not be accessible from all directions, and it does not pass as a bench since the benches are partially occluded by the table. A new functional plan would need to be added which allows such a relationship.

Table 2
Evaluation of Preference Competence in Function-Based Recognition

System Interpretation	Human Interpretation										
	chair	table	bench	bookshelf	bed	cup/glass	bowl	plate	pot/pan	pitcher	not known
chair	50	3	1	1							6
table		12									4
bench	1	4	20								3
bookshelf				15							
bed					25						1
cup/glass						35					
bowl							38		4		1
plate							1	13			1
pot/pan									22		
pitcher										2	
not known	6	4		1	2		5				137

4.2. Preference Competence

A more subtle level of comparison is to look only at the first-ranked interpretations of each shape. We will refer to this as the *preference competence* of the system. This evaluation looks at how well the system makes appropriate rankings between categories. The results of this comparison are summarized in Table 2. The rows represent the system interpretation that resulted in the highest evaluation measure. Similarly, the columns represent a person’s preferred interpretation. If the system’s relative ranking of categories was totally independent of the human ranking of the categories, then the numbers in a given row should be evenly distributed across the columns. If the human and system interpretation were perfectly correlated, then the only non-zero entries would be on the diagonal. Over the entire database of shapes, the system’s preferred interpretation agreed with the human’s preferred interpretation approximately 90% of the time. Figure 10 shows a representative set of these shapes by category.

The shapes for which the system’s interpretation did not agree with the person’s interpretation provide more insight into the problem than do the shapes for which there was agreement. These shapes are depicted in Figure 11. Many of these are the result of the person’s interpretation being driven by what the shape was “meant” to be and not easily being able to consider a shape purely on function-based grounds.

Recognition failures depicted in Figure 11 can be attributed to different factors. As explained earlier, some shapes do not pass analysis because the system cannot confirm that there is a functional element which fulfills the requirements within the specified constraint values. Objects 3-5, 8-13, 21, 36-43, and 46-49 failed due to such conditions. On the other hand, at times the system was able to confirm functional elements of the proper constraints, whereas the creator of the object believed these limits had been exceeded (objects 44-45). Another reason for recognition disagreements was due to the fact that the system was looking for stable conditions when forces were applied to specified functional elements. Objects 6, 14, 16, 17, and 19 all failed due to stability tests. (Object 14 has one leg which is shorter than the other three.)

Other shapes fail as a result of the system identifying novel orientations in which the shapes function as the designated category. For example, shapes 25-27 were labeled as unknown in the human interpretation. However, the system allows the shapes to be

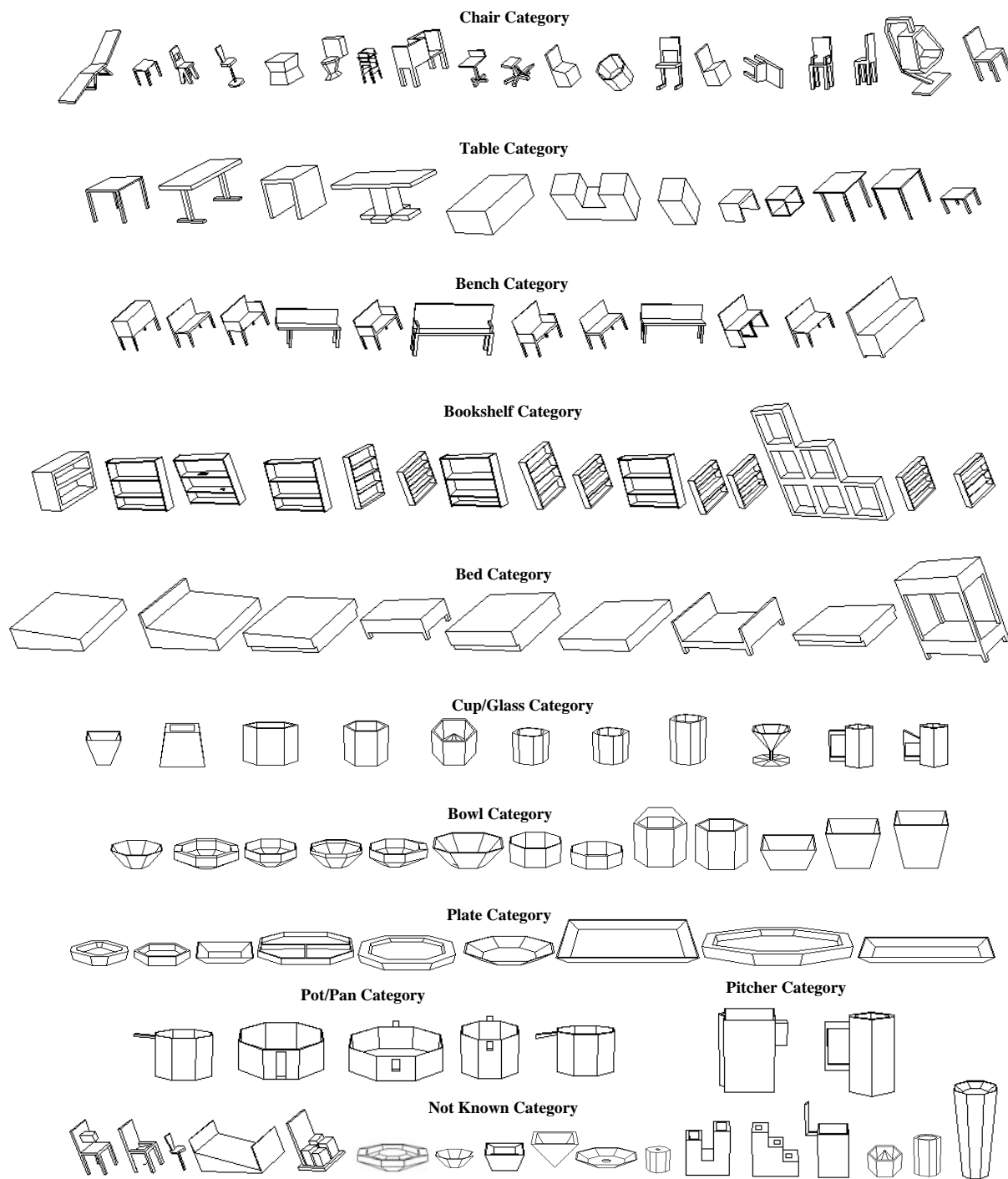


Figure 10. Shapes with Human and System Interpretation in Agreement

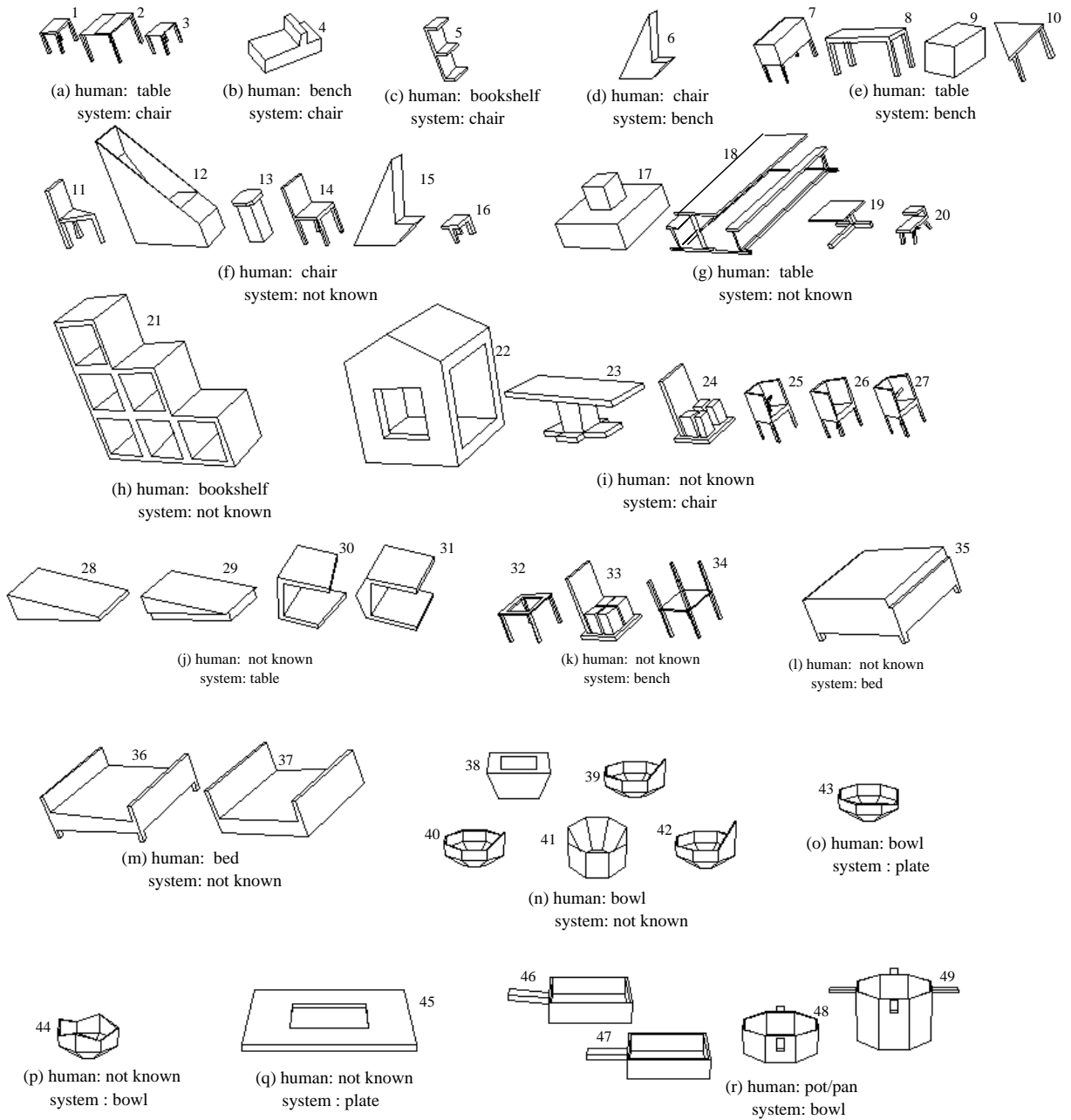


Figure 11. Shapes with Human and System Interpretation in Disagreement

turned over to use the bottom of the chair as a sittable surface. Object 32, which resembles a table with a hole in the center, was found to be able to function as a bench (in four different orientations) at a low measure (0.4) because the system found enough surface area of the proper dimensions at the proper height that could serve as an extended sittable surface.

Overall, we feel that the system has demonstrated great success in performing function-based recognition of shapes in its intended domain of competence. By considering the shapes in Figure 10, it is easy to verify that there are no glaring errors in the shapes that it has recognized as providing the function of one of its known categories. It takes somewhat longer to verify that the shapes classified as “no known category” could not in fact provide the function of any of the known categories, but a careful consideration of the shapes depicted in Figure 11 should be convincing.

5. Evaluation of an Indexing Strategy

We can distinguish between the *competence* and the *efficiency* of a recognition system. The system’s competence is related to how frequently it correctly recognizes objects, and its efficiency is related to how much processing is required for recognition. In the evaluation of the system’s competence, each shape was processed against each known category. For greater efficiency, the normal mode of system operation is to use an indexing strategy in order to avoid as much processing as possible.

The indexing strategy in the current system uses the results of the pre-evaluation of a shape to eliminate “impossible” categories from consideration and to rank order the possible (superordinate/sub)categories. The pre-evaluation yields information such as the number and size of surfaces for both the shape and its convex hull and also the volume of the shape and its convex hull. At least two quantities from the pre-evaluation are obvious candidates for use in indexing the possible categories— 1) the volume of the shape or its convex hull, and 2) the area of possible functional surfaces of the shape. The purpose of an indexing strategy is to choose a key property that best distinguishes between one (superordinate/sub)category and another. A single property could be chosen and used at all levels, or different properties could be used. The current system uses both candidates listed above.

Information used by the indexing strategy is built into the category definition tree. Each subcategory is defined in terms of one or more functional properties that are associated to functional elements of a shape. One of the functional properties may be selected as the key property. In the current version of the system, the key property for furniture is surface area, associated to a surface functional element (sittable surface of a chair, sleeping surface of a bed, ...) while the key property for dishes is volume of the convex hull. For furniture, the constraint values that are used in the knowledge primitive invocations that define the functional property result in a minimum and maximum possible area for the surface that satisfies the functional property. The constraints on minimum and maximum possible volume values for dishes were estimated from prototypical dish descriptions. Thus at each subcategory node, there is a (min, max) range for the key property for the subcategory. At each basic level category node, a similar (Min, Max) range for the key value is computed by taking the minimum of the minimums for each subcategory and the maximum of the maximums for each subcategory. At the superordinate category node, another (MIN, MAX) range is

computed by taking the minimum of the minimums and the maximum of the maximums for the basic level categories.

When a new shape is encountered, the indexing strategy operates as follows. A list of the areas of its potential surface functional elements and the volume of the shape is consolidated. These values are checked against the key value range that is stored at the superordinate category node. If the object key properties do not fall within the (MIN, MAX) range for the superordinate category, then processing is discontinued with the decision that the shape cannot possibly belong to that superordinate category. If the shape meets the requirements of the key property constraints, processing continues to check against the (Min, Max) range of each of the basic level categories and an *indexing measure* is computed for each range. (The *indexing measure* is a different concept from the *evaluation measure*). At this level, area is the key property used for all categories. The indexing measure is one if the area falls directly on the midpoint of the (Min, Max) range, and falls off linearly to zero at the *Min* and *Max* values. We further incorporate the heuristic that if the surface size required for the functional property is larger, it should be weighted higher if a potential match is found. For example, if a shape were to function as a chair, it would be less likely that the shape would have a surface large enough to provide any type of sleeping surface. On the other hand, it would be likely that a shape that could function as a bed would have surfaces within the proper size range to function as a chair, table, bench or bookshelf. The weighting factor is simply the area of the surface being tested. This is multiplied by the raw indexing measure to obtain the weighted indexing measure. Larger surfaces are therefore given a higher weighting factor.

After all of the surfaces derived from the shape have been checked, the categories are ranked according to the greatest indexing measure that they generated. A similar indexing process can then be applied to determine the order in which to consider the subordinate categories. The input shape is then considered against the possible categories in the order of their ranking, until the shape is found to fulfill the functional requirements of some category with an evaluation measure of 0.4 or better. This leaves open the possibility that the system would recognize the shape as one category when in fact there is a lower-ranked (by the indexing strategy) category which would result in a higher evaluation measure.

The ideal indexing strategy would require a “small” amount of work that was independent of the number of categories known to the system, and would always select as the first category to be considered that category for which the shape would have the highest final association measure. The amount of processing required in the pre-evaluation depends on the complexity of the shape but is independent of the number and type of categories known to the system. The number of category evaluations saved by the indexing strategy, as compared to not having an indexing strategy, is summarized in Table 3. With 418 shapes tested and ten basic level categories known to the system, a total of 4180 category evaluations would be done if the system used no indexing strategy. With the simple indexing strategy described above, a total of 1573 potential category evaluations are immediately eliminated because the key value list for the shape has no matches to the key value ranges for the superordinate categories on a whole and also for some basic level categories. With recognition processing ending as soon as the shape is found to fulfill the functional requirements of some category with an evaluation

Table 3
 Evaluation of the Effectiveness of the Indexing Strategy

	Number Of # Objects	# Of Possible Evaluations	Evaluations Actually Made	Evaluations Saved	Indexing Efficiency
No Threshold	418	4180	2607	1573	38 %
0.40 Threshold	418	4180	2207	1973	47 %

measure of 0.4 or better, an additional 400 category evaluations are avoided. Thus 1973 of the 4180 total possible category evaluations are eliminated by the indexing strategy. As mentioned earlier, there exists the possibility that, with the indexing and cutoff at the first plausible evaluation measure, the system could recognize a shape as one category when in fact there is some other category which would result in a higher evaluation measure. This occurred in 25 cases, but in 14 of those the difference in the evaluation measure for the recognized category and the highest possible evaluation measure was less than 0.125. This suggests that most recognition “errors” introduced by the indexing will be between categories which the shape fulfills almost equally well.

6. Discussion

We have evaluated a function-based recognition system that takes an uninterpreted 3-D shape as its input and reasons to determine what object categories the shape might belong to. The system has analyzed over 400 input shapes and the results largely agree with human interpretation. The greatest source of variation from human interpretation occurs with “near miss” shapes which humans label as not one of the known categories, but which have some novel orientation in which they could in fact serve the function for some category. This is a natural reflection of the fact that the system uses a purely function-based definition of the category whereas humans use other types of cues and context in their interpretation.

This work confirms that a relatively small number of knowledge primitives may be used as the basis for defining a relatively broad domain of competence. We have also shown that an indexing strategy may be devised to make the processing for function-based recognition more efficient with a relatively modest penalty in terms of accuracy. This work does not imply that more specific representations of individual objects are not needed. If the task is to label objects within a scene as specific individual objects (e.g. “table model XYZ”), the generic representation by itself cannot provide such information. However, by first categorizing objects within a scene using a generic representation, only those objects that fall within the proper category need to be further examined using more specific representations. Matching of the individual models to the chosen object in the scene can also be completed more efficiently by using the symbolic labeling provided by the generic recognition. For example, the portion of the structure labeled as providing a sittable surface could automatically be aligned with the portion of the structural description labeled seat.

The example object shapes used to evaluate the system in this paper were all defined using a solid modeler. One of the continued research directions that we are working on is to evaluate the system’s performance on models created by extracting surfaces from

real image data taken with a laser range finder. Since the function-based reasoning is naturally qualitative, we expect that the system will maintain good performance with this type of image data.

REFERENCES

- 1 Binford, T.O. 1982. Survey of model-based image analysis systems, *Int. J. of Robotics Research*, **1**, 18-64.
- 2 Bonissone, P.P. and Decker, K.S. 1985. Selecting Uncertainty Calculi and Granularity: An Experiment in Trading-off Precision and Complexity, in *Uncertainty in Artificial Intelligence*, L. Kanal, and J. Lemmer (Editors), North-Holland Publishing Company, 217-247.
- 3 Brady, M., Agre, P.E., Braunegg, D.J., and Connell, J.H. 1985. The Mechanics Mate, in *Advances in Artificial Intelligence*, T. O'Shea (ed.), Elsevier, 79-94.
- 4 Connell, J.H. and Brady, M. 1987. Generating and generalizing models of visual objects, *Artificial Intelligence*, **31**, 159-183.
- 5 Di Manzo, M., Trucco, E., Giunchiglia, F., Ricci, F. 1989 FUR: Understanding Functional Reasoning, *Int. J. of Intelligent Systems*, **4**, 431-457.
- 6 Edelsbrunner, H., O'Rourke, J., and Seidel, R. 1986. Constructing arrangements of lines and hyperplanes with applications, *SIAM J. Computing* **15**, 341-363.
- 7 Kitahashi, T., Abe, N., Dan, S. Kanda, K. and Ogawa, H. A Function-based Model of an Object for Image Understanding, in *Advances in Information Modelling and Knowledge Bases*, H. Jaakkola, H. Kanassalo and S. Ohsuga (editors), IOS Press, 91-97.
- 8 Minsky, M. 1985. *The Society of Mind*, Simon and Shuster, New York.
- 9 Stark, L., and Bowyer, K.W. 1992. Indexing function-based categories for generic object recognition, *Computer Vision and Pattern Recognition (CVPR '92)*, 795-797.
- 10 Stark, L., and Bowyer, K.W. 1991. Achieving generalized object recognition through reasoning about association of function to structure, *IEEE T-PAMI*, **13**, 1097-1104.
- 11 Stark, L., Hall, L.O. and Bowyer, K.W. An investigation of methods of combining functional evidence for 3-D object recognition, to appear in *Int. J. of Pattern Recognition and Artificial Intelligence*.
- 12 Vaina, L. and Jaulent, M. 1991. Object structure and action requirements: a compatibility model for functional recognition, *Int. J. of Intelligent Systems*, **6**, 313-336.
- 13 Winston, P., Binford, T., Katz, B., and Lowry, M. 1983. Learning Physical Description from Functional Definitions, Examples, and Precedents, *AAAI '83*, 433-439.