

Graduate School
University of South Florida
Tampa, Florida

CERTIFICATE OF APPROVAL

Ph.D. Dissertation

This is to certify that the Ph.D. Dissertation of

MELANIE A. SUTTON

with a major in Computer Science and Engineering has been approved by
the Examining Committee on May 2, 1997
as satisfactory for the dissertation requirement for the
Doctor of Philosophy in Computer Science and Engineering degree

Examining Committee :

Co-Major Professor: Kevin W. Bowyer, Ph.D.

Co-Major Professor: Louise Stark, Ph.D.

Member: Anita L. Callahan, Ph.D.

Member: Dmitry B. Goldgof, Ph.D.

Member: Thomas A. Sanocki, Ph.D.

Member: Sudeep Sarkar, Ph.D.

**FUNCTION FROM VISUAL ANALYSIS AND PHYSICAL
INTERACTION: A METHODOLOGY FOR RECOGNITION OF
GENERIC CLASSES OF OBJECTS**

by

MELANIE A. SUTTON

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in Computer Science and Engineering
Department of Computer Science and Engineering
College of Engineering
University of South Florida

August 1997

Co-Major Professor: Kevin W. Bowyer, Ph.D.
Co-Major Professor: Louise Stark, Ph.D.

DEDICATION

To Dunkin' Donuts coffee

ACKNOWLEDGEMENTS

I would like to thank my co-major professors, Kevin Bowyer and Louise Stark, and each of my committee members, Anita Callahan, Dmitry Goldgof, Thomas Sanocki and Sudeep Sarkar, for their encouragement and advice over the last few years. Also, thank you to the staff of the Computer Science Department at the University of South Florida, particularly Rachel, Nancy and Cay, for their assistance each semester with paperwork.

I'd also like to acknowledge the many students who have traveled through the computer vision lab for providing helpful critiques and sharing their programming experience. In particular, I would like to express my gratitude to Bonnie, Adam, Alok, Barry, Gillian, Himanshu, Jennifer, Kevin G., Kevin W., Lynn, Maha, Mark, Matt, Mike, Minesh, Nat, Nina, Ram, Senthil and Tripp.

Many thanks to the staff (in particular Arlene and Laura) of the University of West Florida for all their assistance with course preparations and students. In addition, I'd like to thank several faculty members and friends, Ed and Carol Rodgers, James Bezdek, Lucy, Thomas, John Coffey, and Art Haney, for such a warm welcome to the university and the Pensacola area.

Finally, thanks to the Florida Space Grant Consortium (NASA graduate fellowship), the Air Force Office of Scientific Research (grant F49620-92-J-0223) and the National Science Foundation (grant IRI-9120895) for making graduate school possible. Portions of the work described in this dissertation have appeared in the following publications: [86, 90, 91, 92, 93, 94, 95, 96, 97].

TABLE OF CONTENTS

LIST OF TABLES	v
LIST OF FIGURES	vi
ABSTRACT	viii
CHAPTER 1. INTRODUCTION	1
1.1 Generic Object Recognition	1
1.2 Functional Descriptions of Objects	2
1.3 Overview of Dissertation	3
1.4 Objectives	6
1.5 Layout of Remaining Chapters	6
CHAPTER 2. LITERATURE REVIEW	7
2.1 Psychology: Category Concepts	7
2.2 Artificial Intelligence: Symbolic Labeling of Objects	8
2.3 Object Representations: Incorporating Function	11
2.4 Function from X	12
2.4.1 Function from Shape	12
2.4.2 Function from Motion	15
2.4.3 Sensor Guided Robotic Interaction	17
2.4.4 Function from Manipulation	23
2.4.5 Function from Navigation	25
2.5 Motivation for Dissertation	26

CHAPTER 3. EXPERIMENTAL PLATFORM	30
3.1 System Setup and Coordination	30
3.2 Overview of GRUFF-I Subsystems	33
3.2.1 Model Building Subsystem	35
3.2.2 Shape-based Reasoning Subsystem	37
3.2.3 Interaction-based Reasoning Subsystem	39
3.3 Design of Simulation Mode	42
3.4 Dimensions Which Affected System Development	43
CHAPTER 4. MODEL BUILDING	46
4.1 Calibration Procedure	46
4.2 Range Segmentation Algorithm	50
4.3 OPUS Model Building Algorithm	50
4.4 Model Validity Check	55
4.5 Post Processing of Acquired Model for the GRUFF-I System	55
4.6 Implementation Details	56
4.7 Summary	56
CHAPTER 5. SHAPE-BASED REASONING	57
5.1 The Representation of Functional Knowledge	58
5.2 Static Shape-based Knowledge Primitives	58
5.2.1 Primitive Clearance	58
5.2.2 Primitive Dimensions	60
5.2.3 Primitive Enclosure	62
5.2.4 Primitive Proximity	63
5.2.5 Primitive Relative orientation	64
5.2.6 Primitive Stability	64
5.3 Integrating Shape-based Functional Evidence	66

5.4	Deciding Categorization of the Shape	67
5.5	Implementation Details	69
5.6	Summary	69
CHAPTER 6. INTERACTION-BASED REASONING		71
6.1	Dynamic Interaction-based Knowledge Primitives	71
6.1.1	Primitive apply force	73
6.1.2	Primitive observe deformation	76
6.2	Interaction-based Functional Requirements for Category Cup . . .	80
6.2.1	Confirm graspability	80
6.2.2	Confirm containment	83
6.2.3	Confirm stable containment	84
6.3	Interaction-based Functional Requirements for Category Chair . .	84
6.3.1	Confirm sittability	86
6.4	Implementation Details	88
6.5	Summary	89
CHAPTER 7. EXPERIMENTAL RESULTS		92
7.1	Evaluation Techniques	92
7.1.1	Design of Test Objects	92
7.1.2	Recognition Experiments	94
7.1.3	Verification and Validation Methodology	97
7.2	Demonstration of System Competence	100
7.2.1	Model Building	102
7.2.2	Shape-based Reasoning	102
7.2.3	Interaction-based Reasoning	109
7.3	Discussion of Results	110

CHAPTER 8. CONCLUSION	113
8.1 Summary of Dissertation	113
8.2 Contributions	114
8.3 Implications in the Field of Computer Vision	115
8.4 Future Work	117
LIST OF REFERENCES	120
APPENDICES	128
APPENDIX A SYSTEM TERMINOLOGY	129
APPENDIX B HISTORY OF THE GRUFF SYSTEM	133
APPENDIX C SYSTEM PARAMETERS	136
VITA	End Page

LIST OF TABLES

Table 1.	Summary of Test Object Characteristics	96
Table 2.	Experimental Results (furniture-like objects)	101
Table 3.	Experimental Results (dish-like objects)	101
Table 4.	Summary of Unpredicted Subsystem Failures	101
Table 5.	Thresholds for Dimensions Primitive	137

LIST OF FIGURES

Figure 1. Approaches to recognition of generic classes of objects	4
Figure 2. Recognizing and interacting with generic classes of objects	31
Figure 3. Experimental setup for the GRUFF-I system	32
Figure 4. Operating envelope constraints for the Microbot robot arm	34
Figure 5. Flowchart for the Model Building Subsystem	36
Figure 6. Flowchart for the Shape-based Reasoning Subsystem	38
Figure 7. Flowchart for the Interaction-based Reasoning Subsystem	41
Figure 8. Overview of the simulation mode of the GRUFF-I system	42
Figure 9. Factors which influence system complexity	44
Figure 10. Calibration procedure	47
Figure 11. GRF-2 calibration cube	48
Figure 12. Flowchart for range image segmentation	51
Figure 13. Parameters for the Model Building Subsystem	52
Figure 14. Flowchart for the OPUS model building algorithm	53
Figure 15. Example of functional requirements	59
Figure 16. Knowledge primitive “clearance”	60
Figure 17. Knowledge primitive “dimensions”	61
Figure 18. Using shape properties to calculate an evaluation measure	61
Figure 19. Knowledge primitive “enclosure”	62
Figure 20. Knowledge primitive “proximity”	63
Figure 21. Knowledge primitive “relative_orientation”	64
Figure 22. Knowledge primitive “stability”	65
Figure 23. Example of combining measures	67

Figure 24. Functional properties for cup objects	68
Figure 25. Verifying the functional plan for a cup	72
Figure 26. Positioning of the Microbot robot arm	74
Figure 27. Force profile of the Microbot robot arm	75
Figure 28. Analysis of deformation using intensity images	78
Figure 29. An example of image processing results	79
Figure 30. Interaction tests for a cup object	81
Figure 31. Path planning for cup objects	83
Figure 32. Functional properties for chair objects	85
Figure 33. Verifying the functional plan for a chair	86
Figure 34. Interaction test for a chair object	87
Figure 35. Path planning for chair objects	88
Figure 36. Communication in the GRUFF-I system	91
Figure 37. Functional and non-functional object models	95
Figure 38. Projected failure points for chair objects	98
Figure 39. Projected failure points for cup objects	99
Figure 40. Representative OPUS models for chair objects A1-A5	103
Figure 41. Representative OPUS models for chair objects A6-A9	104
Figure 42. Representative image sequences for cup-like objects B1-B8	105
Figure 43. Object orientations in which no OPUS model could be created	106
Figure 44. Object orientations in which resulting OPUS model was invalid	107
Figure 45. Object orientation which failed shape-based reasoning.	108
Figure 46. Functionality in the ‘large’	119
Figure 47. Complete category definition tree	134
Figure 48. History of the GRUFF system approach	135

**FUNCTION FROM VISUAL ANALYSIS AND PHYSICAL
INTERACTION: A METHODOLOGY FOR RECOGNITION OF
GENERIC CLASSES OF OBJECTS**

by

MELANIE A. SUTTON

An Abstract

Of a dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in Computer Science and Engineering
Department of Computer Science and Engineering
College of Engineering
University of South Florida

August 1997

Co-Major Professor: Kevin W. Bowyer, Ph.D.
Co-Major Professor: Louise Stark, Ph.D.

The **GRUFF-I** (**G**eneric **R**ecognition **U**sing **F**orm, **F**unction and **I**nteraction) system reasons about and generates plans for interaction with 3-D shapes for the purpose of generic object recognition. A researcher selects an object and places it in an observation area. An initial intensity and range image are acquired and provided as input to a three-stage recognition system. The first stage builds a 3-D model. The second stage considers the shape-suggested functionality of this model by applying concepts of physics and causation (e.g., to infer stability) to label the object's potential functionality. The third stage uses this labeling to instantiate a plan for interaction to confirm the object's functional use in a task by incorporating feedback from both visual and robotic sensors. Results of this work are presented for eighteen chair-like and cup-like objects. Major conclusions from this work include: (1) metrically accurate representations of the world can be built and used for higher level reasoning, (2) shape-based reasoning *prior to* interaction-based reasoning provides an efficient methodology for object recognition, in terms of the judicious use of system resources, and (3) interaction-based reasoning helps to confirm the functionality of a categorized object without explicitly determining the object's material composition.

Abstract Approved : _____
Co-Major Professor: Kevin W. Bowyer, Ph.D.
Professor, Department of Computer Science and
Engineering

Date Approved : _____

Abstract Approved : _____
Co-Major Professor: Louise Stark, Ph.D.
Assistant Professor, Department of Electrical and
Computer Engineering, University of the Pacific

Date Approved : _____

CHAPTER 1

INTRODUCTION

1.1 Generic Object Recognition

For a number of years, research in computer vision has addressed a problem humans seem to solve quite easily hundreds of times each day; that is, recognition of objects in the world around us. Early work in this area focused on the use of explicit geometric models for object recognition. Numerous surveys give examples of such research efforts in “model-based” and “CAD-based” vision (e.g., see [4, 7, 23]). However, for any reasonably complex environment, such approaches are ultimately limited by either the storage space required for the set of models or the processing speed at which matching can be performed. Even given sufficient computing resources, it is unlikely that all the parameterizations or component descriptions necessary to sufficiently generalize a prototype object model to an entire object category could readily be anticipated.

Therefore, while traditional model-based vision has been and still is a useful research paradigm, there is an increasing awareness that something more and different will be needed in order to achieve visual perception for “autonomous,” “real world” systems. The system must be able to recognize and deal with truly novel objects; that is, objects for which it has *no explicit prior model*, at least in the traditional sense of the word “model.” This trend has led to computer vision research which emphasizes the development and use of generic object models. A model which is “generic” is meant to capture a category of objects, as opposed to one particular instance. Part

of the inspiration for this approach has come from the field of psychology, where researchers such as Rosch have done extensive work on how humans develop category concepts [69, 76, 77]. Categorization by humans is attributed to the idea that all possible combinations of attributes do not occur in the physical world. Therefore, humans develop an organization of the world in terms of hierarchical category concepts, grouping objects with similar characteristics. In Rosch's hierarchy, there are three major levels of specificity. For example, a basic (or entry level) category is the name most commonly given to an object, such as "cup." To represent a refinement or specialization of a basic category there are subordinate categories, like "mug" or "juice glass." Finally, a generalization of a basic category would be a superordinate category description, such as "dishes."

1.2 Functional Descriptions of Objects

Researchers in artificial intelligence (AI) were the first to begin incorporating ideas about object classes into recognition systems. Well-known early work in the area was done by Winston, Binford and co-workers [102]. They pointed out that there can be an infinite number of different shape descriptions for objects in a category as simple as *cup*, but that a single functional description can be used to represent all cups in a concise manner.

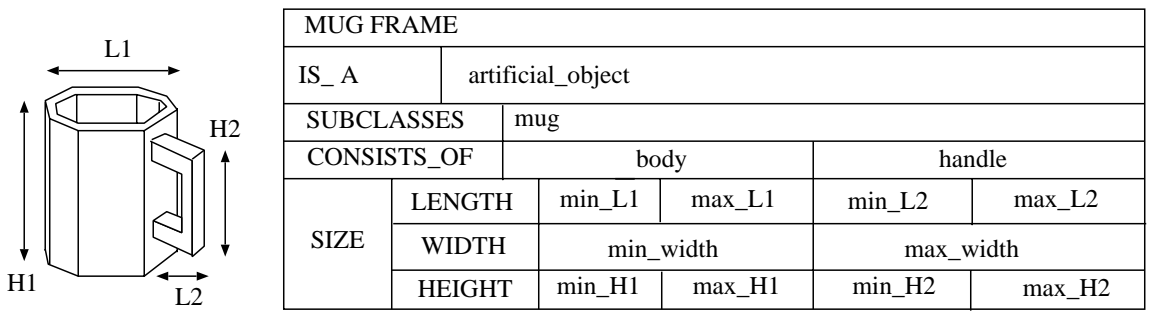
As researchers in these fields and computer vision have gained a greater understanding of the complexity of object recognition, the idea of using functional characteristics of an object as an aid to recognition has further developed. In brief, functional analysis of an object relies on analyzing the object to determine if it can satisfy the requirements of some category of objects or tasks. For example, from Figure 1, we can see that one approach to recognizing cups would be (a) to determine the necessary parameterizations of the object dimensions, or (b) to determine the number

of parts that any given cup in the environment might be composed of. Alternatively, one could describe members of this category in terms of (c) functional characteristics, such as *provides graspability* or *provides containment*. In this way, objects which are truly “novel,” or which contain a unique configuration of parts which has never been seen before, can still be categorized as being able to potentially *function as* a member of a particular category of objects.

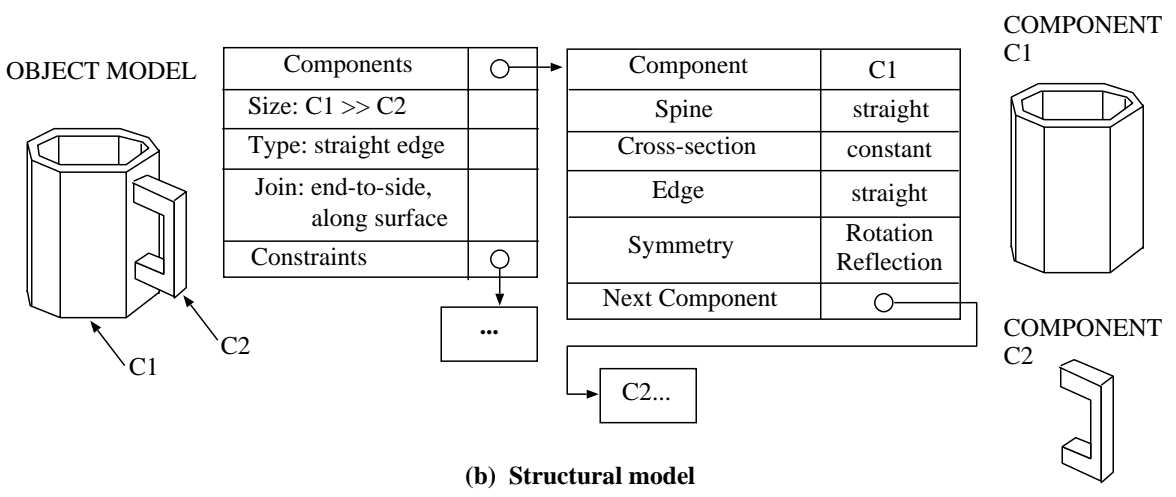
The goal of this dissertation is to demonstrate a function-based object recognition system which begins with real sensory data from a scene and carries through initial reasoning to interact with an object in the workspace. Of course, function-based reasoning assumes that functional attributes can be attached to objects in the environment, and thus it is by no means considered to be *the* solution to the automation of machine object recognition. However, even in humans, a number of studies of visual cognition have indicated the possibility that different levels or types of processing occur, depending on the current visual input or attention level of the subject [70]. As a result, function-based reasoning as a methodology for object recognition can be viewed as an approach applicable to environments in which the objects which need to be recognized were designed and are used with specific purposes in mind. Although beyond the scope of this dissertation, function-based reasoning has also been discussed outside the domain of typical manufactured items. For example, Iberall, *et al.* address the question of capturing the functionality present in human prehension [48].

1.3 Overview of Dissertation

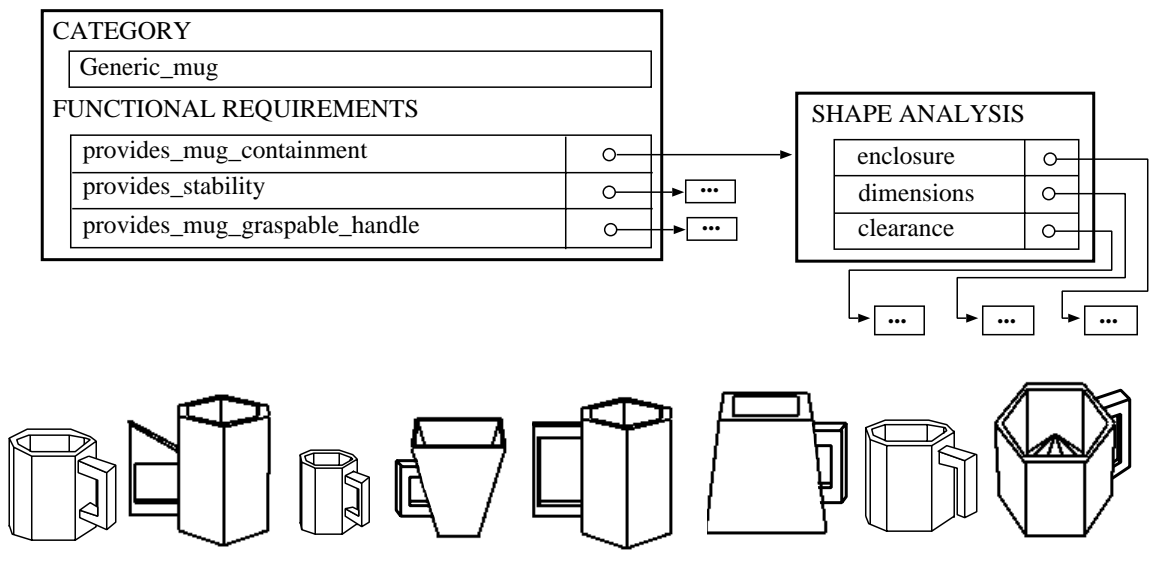
This dissertation describes the implementation of a function-based object recognition system which operates to recognize objects by integrating feedback from both vision and robotics. The system is called **GRUFF-I**, which stands for **G**eneric **R**ecog



(a) Parameterized model



(b) Structural model



(c) Functional model

Figure 1. Approaches to recognition of generic classes of objects.

nition **Using Form, Function and Interaction**. The input to the system is a set of range and intensity images, and the output is an analysis of whether the object can or cannot function as a member of a particular generic category of objects such as furniture or dishes. Each class of objects demands that specific shape-based functional requirements be met, such as *provides graspability* or *provides cup/glass containment*. Once the shape-based analysis labels the 3-D shape of the object to indicate areas of functional significance, a *plan for interaction* is created to direct a robot arm to interact with the object and confirm its suggested functionality.

There are two significant advantages to this approach. First, shape-based reasoning *prior to* interaction allows us to confirm recognition in an efficient manner, when we consider the costs in terms of power and time to calibrate and position mechanical devices such as a robot arm. Secondly, the use of shape-based reasoning *in addition to* interaction-based reasoning allows us to validate recognition of objects that might fool one sensor. For example, feedback from both vision and robotics allows us to disambiguate categorization of an object which may “look” like a member of some category of objects, when in fact it does not have the necessary material properties. This dissertation therefore demonstrates the feasibility of a *form follows function* approach to generic object recognition which considers both the *design* (shape) of the object and analysis of its *use* in a task.

Finally, although the primary goal of the **GRUFF-I** project has been research-oriented to develop a multi-disciplinary research prototype for accomplishing object recognition of classes of manufactured objects, there are potential real-world applications. For example, the ability to automatically recognize dishes/containers would be beneficial to aid the manipulation strategies of mobile autonomous agents dealing with hazardous waste cleanup or removal [57]. In addition, recognition of such items, including furniture, would have relevance to various tasks requiring human-machine cooperation, such as assisting the disabled or handicapped [8].

1.4 Objectives

In summary, the objectives of this dissertation can be stated as follows:

- Demonstrate that metrically accurate representations of the world can be built and used for higher level reasoning.
- Demonstrate how shape-based reasoning *prior to* interaction-based reasoning provides an efficient methodology for object recognition, in terms of the judicious use of system resources.
- Suggest ways in which interaction-based reasoning helps to confirm functionality of a categorized object without explicitly determining the object's material composition.

These objectives are realized in a working prototype system.

1.5 Layout of Remaining Chapters

The layout of the remainder of this dissertation is as follows. Chapter 2 explains related research in function-based object recognition. Chapter 3 provides an overview of the experimental platform and equipment upon which the **GRUFF-I** system is based. Chapters 4, 5 and 6 discuss the Model Building, Shape-based Reasoning and Interaction-based Reasoning subsystems in greater detail. Chapter 7 provides experimental results. Finally, Chapter 8 provides a conclusion and directions for future research. Since this research encompasses a number of terms from artificial intelligence, computer vision and robotics, the reader may find it helpful to review Appendix A for system terminology before continuing.

CHAPTER 2

LITERATURE REVIEW

Reasoning about function has been addressed at different levels in a number of fields including psychology, artificial intelligence, computer vision and robotics. This chapter summarizes the diversity of approaches taken in the area and establishes the framework upon which the **GRUFF-I** system approach was based.

2.1 Psychology: Category Concepts

As mentioned in the previous chapter, research in psychology has suggested how humans might form category concepts. This work was primarily pioneered by Rosch [69, 76, 77]. A focus of her work has been to explain why categorization occurs in a non-arbitrary manner. Essentially she reasons that because all possible combinations of attributes do not occur with equal probability in the physical world, the world is “cut up” in a systematic way. The fact that cuts tend to occur at a particular level is based on cognitive economy (obtaining the most information with the least cognitive effort) and probabilistic cue validity (the validity of a given cue x as a predictor of a given category y , determined by the degree a cue is associated with one category and not others). Rosch’s work has indicated that cue validities are highest for “basic” level categories. Rosch has further found that basic level objects are those which are the most inclusive because they represent clusters of correlated attributes which a set of objects have in common. Research has also supported that given a category

name from one of the three category levels (subordinate, basic, or superordinate level), category names at the basic level will lead to the longest attribute lists, will be associated with more common motor movements, will be most similar in shape and will be recognized more easily from average shapes.

These last two characteristics of basic level categories have led Rosch to postulate a theory of stored prototypes. She reasons that prototypes represent the redundancy structure of the category as a whole, which can save laborious cognitive processing of specific cues when they are not relevant to the task at hand. A more extensive line of research has been in the context of determining what the internal structure is when recognizing instances, judging category membership, searching categories, or determining the logic of the use of category terms in linguistic contexts. In other words, Rosch's work has sought to find evidence that the internal structure of natural categories affects cognition. Tversky and Hemenway later built upon this research by incorporating the notion of parts and part configuration in category perception [100].

In visual cognition research, both Pinker and Kosslyn suggested various uses of classes in visual recognition [55, 70]. Pinker summarized the use of general routines to categorize entities in the 2-1/2-D sketch of a scene, in addition to more specific routines executed for specific classes of objects (rigid, natural, etc.). Kosslyn acknowledged the need for categorical and co-ordinate based representations to process classes of flexible and rigid objects, respectively.

2.2 Artificial Intelligence: Symbolic Labeling of Objects

In the field of artificial intelligence (AI), a number of researchers have investigated how systems might use functional characteristics to discriminate among a set of objects. These systems often begin with a symbolically-labeled object and attempt to

classify it. For example, Dey, *et al.* incorporated ideas about recognition and learning of generic classes of objects in designing a system to create a knowledge base for the handtools domain [28]. The system began with object descriptions in terms of primitive parts, associated features and relations known to the system. Features included such attributes as the material composition, part dimensions (length, etc.) and part shape. The system learned to distinguish between curved-spined hammers (e.g., claw and rip hammers) and straight-spined hammers (e.g., tack, mallet and ball hammers), and other categories of handtools, such as wrenches (e.g., spanner wrenches). The system had the ability to determine if attributes occurred frequently enough in subclass definitions to become associated with a parent node, for space optimization. Furthermore, the system could perform additional matching for identification to a predefined set of known objects.

In the system designed by Winston, Binford and co-workers, functional descriptions were used to represent objects from the cup category [102]. The definition for a cup was organized in a semantic net with characteristics such as “cups are those objects which are stable, liftable, open vessels.” The system was then provided with the physical description of the geometry of an input object, in English. This description described the object using the generalized-cylinder vocabulary of the ACRONYM system and could be enhanced with additional non-geometric or non-visual details about the object, such as its material composition or weight. For example, a sample description would be that the input object was light, had a handle, a flat bottom, a small body and an upwardly-pointing concavity. The system would then determine whether the object was in fact a cup by, for example, attributing the functional requirement of stability to the flat bottom structure and the functional requirement of liftability to its light weight and handle structure. The system additionally incorporated learning of functional requirements and generalization using examples from other categories of objects and analogy.

Minsky argued persuasively for the connection between structure and function, using the category “chair” as an example [68]. Minsky believed that structural descriptions based on parts (e.g., a chair is something which is composed of legs, a back and a seat) were inadequate descriptions, since many objects in a category as simple as a chair did not divide easily into such separate parts. Alternatively, he believed that connections must be established between the structure of a chair and its intended use by a human. In this manner, a chair would be described as something which provided some structure to support the sitter’s body, structure to maintain seat height and structure which allowed room for knees.

DiManzo proposed a recognition system design that utilized knowledge about function within an expert system framework [29]. The design was limited in knowledge about different subcategories. Primitives were defined in the form of individual expert systems that evaluated the 3-D information using constraints defined by the user. A prototype system was proposed which would receive a 3-D description of a scene generated by an octree solid-modeler.

Vaina and Jaulent investigated the use of both conceptual and structural object descriptions for functional recognition [101]. Their work emphasized the auxiliary functions of objects (how objects may be used in actions other than those originally intended by the designer) and how these add to the complexity of recognition based on function. They outlined a system using fuzzy set and possibility theory which relied on determining the degree of match between an object’s functional properties and the action requirements which accomplished a specific goal.

Within the context of device function, Goel investigated a system which analyzed the design of physical devices such as electrical circuits and heat exchangers [32]. Given as input a transformation function of a desired device, his KRITIK system provided as output a structural modification plan that could be applied to one of ten available device designs to provide the desired function. In other words, the

system provided an explanation of how the structure of a device contributed to its ability to provide a specific intrinsic function and causal behavior. The system also demonstrated the evaluation strengths of functional models for new designs, which could differ in several features from previously seen designs.

2.3 Object Representations: Incorporating Function

As work in the area of generic object recognition has continued, a number of researchers have begun investigating the idea of incorporating functional characteristics into the representations of objects, for the purposes of recognition, manipulation, and/or navigation. For example, Hodges developed the EDISON system which uses computational models that can solve problems and reason creatively about mechanical devices [40, 41]. In his Functional Ontology for Naive Mechanics (FONM) representation model, Hodges stressed the importance of structural characteristics in addition to a representation of experience based on function. This is because mechanical improvisation scenarios require the problem solver to recognize in a device some functional capacity which it may not have been used for previously, or was not designed for.

Brand has described a system using causal and functional analysis to interactively build a model of relationships between parts in a scene [14, 15, 16]. Additional work with Cooper, *et al.* focused on understanding image scenes from the point of view of providing a causal explanation of the scene to explain how an agent could interact with it [26]. Three systems (BUSTER, Fido and MugShot) addressed the issues of directing focus of attention, handling occlusion and grasp planning. The MugShot system used a fixed interaction strategy based on a causal scene explanation developed by the system designers, in order to direct a robot arm to lift a mug placed in a workspace. This work stressed the importance of including in the rep-

resentation the potential for action with the object, which some other agent could perform.

Finally, Gupta, *et al.* discussed the representation of design knowledge for engineering knowledge bases and stressed the importance of incorporating function, form (geometry and non-spatial structure), and behavior into the representation [36]. They believed that representation systems should be more flexible and integrate more thoroughly the influence of behavior on components, in addition to form. Behavior involved physical laws and relationships with form, in addition to interrelationships between behavior patterns, and also helped to identify artifacts which could be used in a given context.

2.4 Function from X

A number of additional methods, which can be termed *function from X* techniques, have been developed for extracting “functional” information from input images, models, or interaction. For example, research in *function from shape* uses the idea that the 2-D or 3-D shape of an object provides some indication of its function. *Function from motion* attempts to recognize the function some object is serving, as the system observes a task being performed with the object. Finally, *function from manipulation/navigation* addresses recovering the function of an object by manipulating or navigating around it.

2.4.1 Function from Shape

With respect to *function from shape*, early work by Brooks on the ACRONYM system recognized the relevance in the world of man-made objects of including classes of objects based on function discernible by analysis of geometric structure [17, 18, 19].

Objects were modeled as subpart hierarchies of generalized cones, where variations in size, structure and spatial relationships were permitted for recognition.

Solina and Bajcsy presented preliminary work on ideas about shape and functional categories using a part-based approach [78]. Parts were modeled using superquadric volumetric primitives. Each category of objects was composed of objects made of the same set of parts, which could be represented using a single prototype. Variation within a category was allowed by specifying *a priori* parameters for the variability of part positions and shape deformations of the parts. A pair of reflectance images of an object presented in its canonical position (upright) was analyzed for features (i.e. handles, parallel edges, etc.), which were subsequently used to index into a data base of known model prototypes. Recognition was accomplished by deforming each part of the stored prototypes to match the corresponding object part and selecting the prototype with the closest match, which was assumed to be the prototype requiring the least amount of deformation. Gross shape characteristics were assumed to account for various object properties. For example, flat surfaces were assumed to indicate “support,” whereas elongated surfaces were an indication for “manipulation” areas.

Although the idea of functional categories is mentioned in Solina’s work, the approach is more similar to parameterized structural model approaches, as outlined in [93]. For example, in Solina’s work, all cups are required to have one handle, and new categories must be established for two-handled cups, no handled cups, etc. This deviates from the idea that the category of cups (regardless of exact structure) satisfies the single functional requirement of providing containment.

Brady and colleagues discussed the relation between geometric structure and functional significance in their design of the “Mechanic’s Mate” system [13, 25]. In part of this work, semantic network descriptions were computed from 2-D shapes, and a generalized structural description was learned from a sequence of positive examples.

For the object category examined, *hammers*, the elements of a structural definition, “handle” and “head,” closely correspond to the elements of a function-based definition, “graspable” and “provides a nail-striking surface.”

Ho considered the function-based modeling of the object category *chair* [39]. Ho’s model was formed in terms of the physical structure of the object, termed the “artifactual concept,” in conjunction with the spatial relationships of visual features that may exist, termed the “functional-relational concept.” The system worked with the ideal 2-D cross-section of the object and assumed that it appeared in its typical upright orientation.

Kitahashi, *et al.* brought the “functant” into their representation to help define prototypical shapes [54]. Functants were objects which were recipients of functions and thus aided in constraining object shape.

In early work by Davis, analysis of hard rigid objects such as boxes, cages, buttons and rings suggested ideas about how function related to shape and could be used to design geometric heuristics for analyzing the use of the object in a situation [27]. For the objects presented, Davis considered the most important geometric analysis to be that related to testing containment, fitting through and support. For example, constraints such as “gravity” involving primitive ideas about physics could be defined by the conjunction of constraints such as “hard objects,” “must fall” and “stationary ground.”

In a more generalized approach to modeling functional categories, a system designed by Stark, *et al.* described the 3-D functional requirements for objects to be members of the category *chair* [83]. The system attempted to answer the question—given a description of the complete 3-D shape of an object, can the system appropriately categorize the object according to the function(s) that it can serve? Functions included requirements such as *provides sittable surface* and *provides stable support*, which could be confirmed by analyzing a 3-D model of the given object. This system

was later expanded to include functional descriptions for 400+ rigid shapes in the superordinate categories furniture, dishes and handtools [85, 90, 91, 92, 94, 95].

Rivlin, *et al.* have done extensive work in the area of reasoning about the functionality of an object's parts, recovered from 2-D or 3-D image data [72]. For the category hammer, after part recovery, reasoning was performed about the functionality of the parts and interactions, such as relative orientation, size, or motion. Top-down analysis of an intensity image of a functional mallet was presented to demonstrate the use of their approach in expected object recognition, where knowledge of the stored internal model of a hammer was used to constrain image search for potentially functional regions. Bottom-up analysis of an intensity image of a non-functional hammer was presented to demonstrate the use of the system for unexpected object recognition, where the task was to segment the image into regions and recover qualitative shape information and relations to generate possible object hypotheses.

Kim and Nevatia used functional descriptions to describe objects when creating a map for a mobile robot [52]. The input to their system was derived from a set of images, and knowledge of the world was based on functional categories. For example, a door was defined as an opening that allowed passage through it and a means of closing it, and a desk was defined as an object that a human could work comfortably at and could have objects placed on it. Characteristics in the intensity image, such as size, height, shape and orientation of edge segments, were used for recognition.

2.4.2 Function from Motion

Tversky and Hemenway have argued that while one can gather perceptual properties from observing a static object, knowledge of behavioral (functional) properties of an object involves observing the object in use or in motion [99]. Research in *function from motion* therefore addresses determining the suitability of some object for

satisfying a particular function, and/or recognition of what particular function an object is serving, as it is observed by a system.

Early work in this area by Bogoni, *et al.* investigated manipulatory interactions, such as piercing [9, 11]. Three different tools were placed within a Puma-560 gripper, equipped with a Lord force sensor, and were subsequently used to pierce three target objects whose shape and hardness varied (e.g., objects composed of Styrofoam, balsa wood and pine). As the operation was being performed, data was gathered on the visual height of the tool, the force signal orthogonal to the object surface and the position of the end effector. Analysis of this information indicated whether or not the object was capable of performing the operation of piercing. In this research, the emphasis was on the verification and recovery of the material properties of an object, using exploration techniques. Although the shape of the object could be monitored as the interaction was commencing, the shape itself was not used for recognition of the object prior to interaction.

Duric, *et al.* examined motion sequences of known objects as they were being used [30, 31]. Using information about the object, including its typical uses and the analysis of its motion, this system attempted to recognize functions such as jabbing, chopping, stabbing, scooping, hitting, tightening and hammering. Objects were modeled as shapes composed of primitive parts (i.e. sticks, strips, plates and blobs), and sequences of images of the object were analyzed for primitive motions (i.e. rotation and translation). For example, for the knife category of objects, once the object was located in the image (using the average of all edge points for which normal flow was computed), motion of its main axes relative to some action surface (termed the *actee*) allowed the recognition of manipulation tasks such as chopping and stabbing.

Kise, *et al.* proposed a functional model that was capable of encoding dynamic functions such as force and motion common to objects in the category handtools [53]. For recognition in the domain of wrenches and bottle openers, 3-D descriptions of

objects were “mechanically” decomposed into machine elements, such as levers, to confirm functional constraints.

Green, *et al.* continued work on the system designed by Stark, *et al.* and expanded it to include analysis of the 3-D functional requirements of some non-rigid objects, whose function depended on parts joined by articulated connections [33, 34, 35]. Examples from this hierarchy of objects included pliers and scissors. From a sequence of complete 3-D shape descriptions of an object, an articulated shape model was created which was composed of part descriptions, connections and linkage relations which explained the observed motion sequence. This articulated model was then analyzed for such functional requirements as *provides opposing finger grasp* or *provides opposing cutting blades*. Experimental results for this system were provided for 24 different shapes, with varying degrees of membership for the category scissors.

2.4.3 Sensor Guided Robotic Interaction

Function from manipulation/navigation addresses recovering the function of an object by manipulating or navigating around it. However, before discussing research in this area, it is relevant to summarize related work in the field of robotics, where research has addressed the various ways in which different sensors can control or guide robotic interaction with objects.

With respect to grasping strategies in general, Iberall developed a hierarchy of 19 unique grasp postures for the categories dishes and handtools (both rigid and articulated) [47, 48]. This subset included different postures which could be necessary for subtasks involving the same tool. Iberall used an action/object/context description of interaction. For example, to describe the functionality of hand posture, relative to handtool manipulation, parameters included the location, magnitude and direction

of an applied force and the width of the applicable hand surface. This representation was used in a grasp planner to map object features to hand posture and to capture the contexts within which the object would be used. Krishnan has also discussed similar planning strategies for robotic grasping and wielding [56]. Michelman, *et al.* have implemented a set of primitive manipulation functions which can be combined to solve complex tasks such as removing the top of a child-proof medicine bottle [65].

With respect to contact sensing, Krotkov has investigated using vision and robotics to perceive the object's material properties [57]. In this work, a *push and feel* interaction determined the compliance and shear strength of natural terrain (sand, sawdust, or soil) using a six-axis force-torque sensor and joint position sensors attached to a robot leg which exerted compressive and shearing forces on a terrain sample. A *tap and listen* interaction used acoustic sensors (microphone, amplifier and filter circuit) to classify materials (wood, concrete, clay, zinc, or ceramic) when they were struck with a cane. A *strike and watch* interaction estimated the mass and the coefficient of sliding friction of a material based on observing its trajectory when struck with a wooden pendulum.

More recent work by Krotkov has focused specifically on analysis of acoustic input using a microphone and rods composed of wood, brass, aluminum, glass and plastic (each in two lengths) [58]. Each rod was suspended and then struck with a well-damped solid object. The focus of this work has been to determine what acoustic information can be used to diagnose the material and is invariant to object size and shape. Krotkov pointed out that visual cues such as surface luminance can be a key to material properties (e.g., coefficient of friction), but can also be disguised in real-world environments. Along this vein, an additional representative example of research in non-contact sensing of material properties of objects would be the work of Campos, who used analysis of object temperature [20].

There are several researchers who have sought to more thoroughly integrate vision and robotics to perform interaction tasks. One way to accomplish this is to require only partial model reconstruction without recognition, assuming that vision alone may often be fooled by surface labelings or coatings, and is useless for properties such as friction. The work of Trobina and Leonardis is representative of the trend toward purposive vision systems in this area [98]. Such systems only extract that information from the scene which is relevant to the current task. Therefore, for a task such as grasping, there is no need to perform an explicit or complete model reconstruction and recognition stage. Specifically, the system designed by Trobina and Leonardis attempted to grasp rigid arbitrarily shaped 3-D objects such as milk cartons, tape dispensers and mugs from a pile, beginning with the highest one. Grasping strategies were based on information about planar patches recovered from a range image of the scene.

Hager used edge data derived from a single intensity image to consider all possible ways to approach an arbitrary object and grasp it stably by solving equations based on force, torque and friction [6]. In additional work, Hager, *et al.* have worked on a calibration-free system using a Zebra Zero robot arm with a PC controller and cameras to direct robot positioning for picking up arbitrary objects [38]. Rather than sending the robot to an absolute position in the robot frame of reference, which would require careful calibration, stereo vision was used to send the robot to a position which minimized the distance between the robot end effector and a pre-defined position in the visual reference frame. Experimental observations were taken during 50 trials where a robot was directed along a square trajectory. The type of motion between points (straight line or unconstrained), the control gains, the amount of time the manipulator paused at each point and the amount of camera movement during the experiment were varied. A maximum average error of approximately 1 mm for position accuracy was determined. In addition, the initial calibration did not need

to be completely accurate. The cameras could be moved up to 30 cm and rotated up to five degrees without any appreciable system degradation. This approach was thus useful where the task to be executed could be broken up into a set of visual reference points (image features) which could be tracked quickly and accurately [37]. In a similar line of work, Allen, *et al.* have addressed using real-time vision modules to compute applied forces for grasping tasks [3].

An alternative approach is to require more complete model recovery and recognition, prior to any interaction. Ade, *et al.* used intensity images derived from three CCD cameras (two lateral silhouettes and an image from above) to analyze collections of dishes and cutlery on cafeteria trays [2]. In later work, they avoided the use of explicit object models and used range data to recover geometric information which served to detect grasping opportunities for a two-fingered gripper in order to perform object removal [1]. Dishes were modeled as circles in the image; cutlery was modeled as symmetrical ribbon-like structures (curve pairs).

In Ade's system, using feature information acquired from the images and a domain-specific rule base (consisting of 46 rules), the highest object in the scene was identified, and a robot was directed to grasp and remove it. For this work 11 objects (4 porcelain and 3 glass dishes and 4 cutlery pieces) were used. Objects were expected to be presented in their typical upright or normal position, and the system was able to handle transparent materials. The system was tested on several hundred scenes, with a failure rate of around 5 percent for typical cafeteria tray configurations with no food remains. Errors could be attributed to the symmetry axes of the dishes being more than 15 degrees from vertical, the dishes being inverted, or the cutlery pieces being too polished.

Connell designed a system which considered the domain of brightly colored, cylindrically shaped handtools [24]. Objects were modeled as 2-D blob-like pixel regions in intensity images with expected characteristics such as size, position and

orientation and signature-like features, such as texture, color, gross shape and particular subpart configuration. For example, pliers were modeled as large-sized pixel areas in the image, blue in color, which had a shape which consisted of a pair of elongated areas (i.e. a pair of handles). Screwdrivers were modeled as medium-sized pixel areas, yellow in color, which had a single elongated area (i.e. one handle).

Connell emphasized the development of two distinct components in his system. The vision component was used to determine where the object was and at what angle to grasp it, using primitive behaviors such as *search above the table top* or *find big red things*. Morphology operators and edge constraints were used to resolve ambiguities among objects in the scene. For example, to distinguish a mass of red wires from a red battery charger, the erosion operator was used. For the mass of red wires, the erosion operator caused fragmentation. However, for the red battery charger, the erosion operator left the blob intact. Edge constraints were also used. For example, to distinguish a yellow screwdriver from a pair of yellow pliers or the top of a yellow jar, the extracted blob was required to have straight sides.

The second component in Connell's work, the robotic subsystem, performed the grasping actions using primitive robotic behaviors, such as *orient*, *contact*, or *push* to center the gripper and align it for grasping. These could be used to create behavior-based controllers for different tasks which allowed the system to compensate for small perturbations in the environment without requiring re-planning of a given interaction sequence. As an example, in Connell's system to find a pair of pliers, the system would:

1. Find the edge of the table, by finding a long edge in the lower third of the image.
2. Find connected components of areas of the image with a hue in the blue range, with the constraint that it must be on the table, i.e. "close to" edge of table.
3. Look for areas that are "plier-like," (elongated pixel areas, occurring in pairs).
4. Perform dilation, erosion and size thresholding based on expected pair size.

5. Rank the matches found in the image, using characteristics such as distance and reachability.
6. Generate a stereo pair of images to get the depth of object, using the centroids of regions for the correspondence problem.
7. Transfer control to the Robot Component.

Subsequently, the Grasp Tool Controller would be initiated, with the following sequence:

1. Position above object (location provided by the Vision Component).
2. Rotate gripper to center object.
3. Pitch fingers for gripping.
4. Position down upon object and grasp.
5. Raise object a small distance.

Bishay, *et al.* used a Bridgestone pneumatically-actuated manipulator and monochrome CCD video camera mounted on the arm to locate objects for a black-board architecture-based system called ISAC (Intelligent Soft Arm Control), designed to help feed the disabled [8]. By tracking the gripper and object in the acquired image, an error measure was determined to correct for the difference in position of features belonging to each (object and gripper). This error measure initiated additional control points for the robot arm to lead to the grasp point on the object. The gripper was located in the image by using a matched filter. The image of the object was located by using a binary object recognition algorithm based on an artificial neural network, trained with a circular histogram of the object. Two sets of 100 trials were conducted to test the system by placing a spoon at random locations and orientations in the workspace of the arm. A success rate of 81-91% was determined for finding and grasping the spoon. Additional work in this area has focussed on developing a low-cost active vision system for this robotic application [5] and addressing the practical issues surrounding the incorporation of humanoid robots in home and industrial applications [51].

Lapierre, *et al.* worked in the area of computer-aided driving using artificial intelligence and computer vision [60]. They developed a blackboard architecture-based system which could interpret and react to traffic scenes using multisensorial data fusion. Objects such as roads were modeled with static information such as road curvature and lane markings. Traffic objects were modeled with dynamic information such as the location of other moving cars and obstacles on the road.

2.4.4 Function from Manipulation

Function from manipulation addresses recovering the function of an object by manipulating it. Stansfield investigated the types of information a gripper could extract just from exploring an arbitrary object placed in front of it [79, 80]. This information was then used to perform recognition by matching the recovered characteristics to prototypical exemplars. This use of robotics and vision emphasized the modeling of generic object categories from the kitchen domain of objects. The concept of a six-sided *spatial polyhedron* was introduced to represent each class of objects in terms of the spatial configuration of required parts and function-based features that must be able to be sensed along six corresponding (fixed orientation) directions by the end effector. The model provided a means of classifying objects with wide structural variations of a particular feature, provided that the feature was sensed in one of six spatial locations indicated by the model. In addition, the model allowed the system to make predictions about unsensed views, or missing parts. In this work, interaction could be invoked to determine such aspects as which surfaces were non-elastic or non-compliant. The goal was therefore to combine the results of *independent* haptic exploration and feature recognition.

In Stansfield's system, basic level categories of objects were used. A basic category of objects was represented by a prototypical model in its canonical position

(upright orientation). The prototype described each of the parts and their configuration. The prototype captured structural variations in the category by specifying *a priori* allowable changes of contact positions among the parts. Each part was modeled by superquadric volumetric primitives (with deformation parameters specified in terms of allowable directions and amounts of deformation). Finally, each part was described by a set of features that could be used to recognize it in an image. Recognition was based on prototype matching. As an example, to find and recognize a cup using Stansfield's system, the following steps would be initiated:

1. Threshold the background; find connected edges and regions.
2. Perform feature detection: look for holes, cavities, handles, parallel surfaces, parallel edges, rotational symmetry, etc.
3. Determine where objects are and describe in terms of position, orientation, number of parts, and object dimensions.
4. Hypothesize prototype models that match the features detected.
5. Since features alone are not sufficient (different arrangements of the same features result in different objects), determine the arrangement of parts in the object.
6. Deform each part of the hypothesized prototypes to match the appropriate part of the object.
7. Verify the shape of individual parts in the superquadric prototype with the 2-D contour and 3-D points of object parts.
8. Estimate the direction and amount of deformation to determine closest match.

Finally, Stark has explored using function-based reasoning with the **GRUFF** system and analyzed simulations of object interaction using **ThingWorld**, a physically-based solid modeling system [81]. These simulations were carried out for the category *chair* and emphasized the use of shape-based reasoning to generate a plan for interaction. A set of objects composed of oak and/or rubber was defined, and deformation of object structure was observed as forces were applied at various points. Deformation was measured by the system designer manually creating a 3-D model of the object at

different points during the interaction, and providing this model to the **GRUFF** system for analysis of the functionality of its new (possibly deformed) 3-D shape.

Results were presented for a simple chair composed entirely of oak which did not deform and a chair composed of an oak seat and rubber legs which did deform under applied weight. Additionally, a set of straight back chairs composed entirely of oak, of an oak seat and a rubber back, and one composed entirely of rubber was tested. The set of objects was required to pass interaction tests for providing a sittable surface and providing back support. The solid oak and the oak/rubber straight back chairs were able to pass the first test, whereas the rubber chair failed. In the second test of confirming back support, only the solid oak object did not deform.

2.4.5 Function from Navigation

Function from navigation addresses recovering the function of an object by navigating around it. Rivlin, *et al.* [73] discussed the issue of such classes of navigational functionalities, where objects were classified as threats or obstacles, prey or food, or landmarks. Categorization was based on characteristics such as object motion, size, or comparison to a reference view. In the context of performing the task of cleaning the corridor floors of a building, their navigating agent, Sisyphus, was designed to recognize independently moving objects as potential threats, large object masses as obstacles (to be avoided), small object masses as litter (to be intercepted as prey or food), and specific places (to be used as references for navigation/landmarks, such as the corridor walls and ends of the main cleaning area). Each of these recognition tasks corresponded to a different class of object functionalities for the navigating agent, depending on how the class of objects impeded or facilitated its motion in the environment.

2.5 Motivation for Dissertation

From this literature review, it should be evident that the ideas behind grasping objects and reasoning about function for the purpose of recognition are not new. A number of researchers are actively studying different aspects of these areas. In addition, there are many applications where there is still a clear need for autonomous agents to be able to interact efficiently and usefully with the environment. The goal of the work described in this dissertation is *not* material property identification, such as that used in sensor guided robotic interaction (e.g., [20, 57, 58]). The objective is instead to determine if an object has the *sufficient* material property characteristics to function as a member of a category. In other words, whether a chair is composed of wood or metal is not evaluated by the system. Rather, the system determines if in either case the object would be able to support some amount of weight on its “sittable surface.” Consequently, of the research described above, the work which is most closely related to this dissertation includes that which performs model recovery and recognition based on functional categories. In addition, there must be an intention to perform interaction, using this derived information.

Recall that in Connell’s system [24], the following steps were used:

1. Perform feature extraction (blob detection).
2. Perform filtering based on expected region shape/size/position (recognition).
3. Use attentional selection to key in on object.
4. Perform interaction.

The work described in this dissertation is similar to Connell’s in the sense that we also use versatile operators in the vision and robot components. However, Connell’s system does *not* explicitly look for functionally relevant information, but rather, for color-based or shape-based features such as red handles of a certain dimension, in pairs, etc. In addition, Connell maintained that 2-D analysis is advantageous because

only a minimal “model” description was needed for operation. Further, he believed it was difficult to acquire a geometric model of the target object and for a vision system to build metrically accurate representations of its surroundings. Alternatively, this dissertation demonstrates that sufficiently accurate 3-D representations of the surroundings are possible and more readily applicable to a greater variety of more complex domains of objects, compared to a 2-D blob analysis approach to recognition. In addition, the 2-D data in Connell’s system used the assumption that all objects were of negligible height, such that 2-D edge data acquired from the image could be easily used to determine graspability. Alternatively, graspability in the **GRUFF-I** system relies on using the 3-D locations of points on the object.

Also, recall that in Stansfield’s work [79, 80], these steps were employed:

1. Acquire visual-spatial features.
2. Acquire haptic features.
3. Perform prototype matching.

One similarity the body of work in this dissertation shares with Stansfield’s work is the use of dimensional constraints to distinguish across basic level categories. However, the work described in this dissertation differs in a number of fundamental ways. First, Stansfield believed that each category of objects could be modeled by an n -sided prototype (the most representative shape in the category) and that a set of shape deformations could be specified *a priori* of that prototype to account for all variations inside that category. With this methodology, functionally similar but structurally different objects were organized as members of separate categories of objects. Therefore each object in a given category was assumed to have the same set of parts. This meant all “cups” must have one handle. “Two handled cups” would call for the formation of a new category, as would “no handled cups.”

A drawback of such part-based approaches is that they require the definition of a finite and inclusive set of primitives or sufficient vocabulary of parts into which

all objects can be decomposed. Alternatively, the **GRUFF-I** system is *not* part-based. Rather, shape-based analysis allows us to identify functionally significant *areas*. These areas are determined based on applying general inference rules about shape, as opposed to rules for partitioning based on pre-specified shape properties. Therefore, whereas the n-sided prototype representation could permit fewer faces for simpler objects, or may need more faces for manipulating complex objects, the **GRUFF-I** system has no such constraints. Once a functionally significant area is found, regardless of its location, the robot can be sent to approach that specific area, along any orientation, constrained only by manipulator positioning limits.

Finally, in Stansfield's approach, exploration of the object was *not* model driven, and no *a priori* knowledge about the object was desired or assumed by the system. Exploration of the object meant approaching the object along preset sensing planes, and subsequently invoking exploratory procedures based on local tactile exploration. The result of these two independent steps was the determination of a set of 3-D points for the object and a set of volumes for each visible part using the vision component. Tactile features (e.g., smoothness, noncompliance, nonelasticity, etc.) were acquired for each part of the object along each of six planes, using the robotic component.

The **GRUFF-I** system approach differs from this in that the output of the vision component is used to recognize the object, *prior to* any interaction. In this sense, the **GRUFF-I** system approach resembles active vision systems which seek to interpret the current state of the scene and react in a meaningful manner. In other words, the system seeks to understand not only *what* and *where* the objects are, but also *how* to interact with them efficiently. In this way, less energy is expended whenever objects are outside the domain of the system, or truly non-functional in relation to a given category.

Finally, the work described here is a direct extension of the system designed by Stark [81]. Recall that this system used function-based category definitions, performed shape-based reasoning and simulated interaction plans using the **Thing-World** solid modeling package. The **GRUFF-I** system differs from this in that it contains physical visual and robotic components, operating on-line, with real objects. In addition, the **GRUFF-I** system has been designed to handle interactions with objects from both the furniture and dishes categories. Preliminary results of the system described in this dissertation have been presented in [96, 97].

CHAPTER 3

EXPERIMENTAL PLATFORM

An overview of the flow of control in the **GRUFF-I** system is provided in Figure

2. The system is composed of three separate subsystems, as follows:

- Model Building - the subsystem which uses the range image acquired with the vision component to build a 3-D model of the object in the scene.
- Shape-based Reasoning - the subsystem which performs static shape-based labeling of the 3-D model.
- Interaction-based Reasoning - the subsystem which uses the vision and robot components to perform dynamic confirmation of the 3-D model, by interacting with the object in the scene and analyzing subsequent 2-D intensity images and robotic sensor feedback.

The following sections describe the hardware components, the flow of control and the data formats for each of these subsystems.

3.1 System Setup and Coordination

The **GRUFF-I** system runs on a SunSPARC Ultra 1 using the Solaris operating system (SunOS 5.5), with RS-232 connections to a robot arm and the vision component (see Figure 3). The vision component can acquire both intensity and range images, and is composed of a K2T Gray-Code Range Finder and Sony CCD Monochrome Video Camera with a K2T V300 RGB Digitizer and Display Board (i.e. a structured light scanner). With the V300, video data at a resolution of 480 rows by

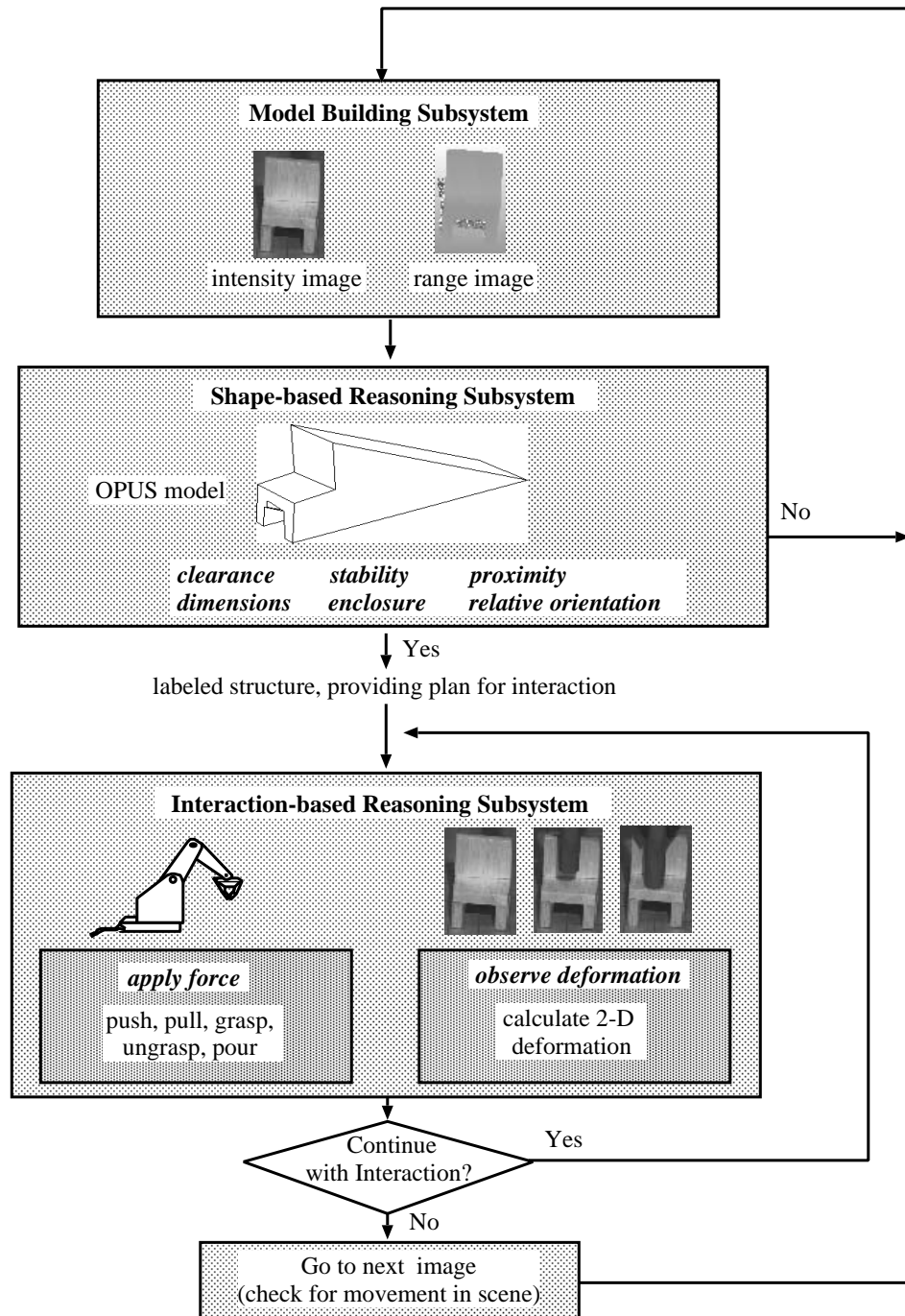


Figure 2. Recognizing and interacting with generic classes of objects. The methodology described above involves integrating information from visual and robotic sensors to achieve recognition of categories of objects.

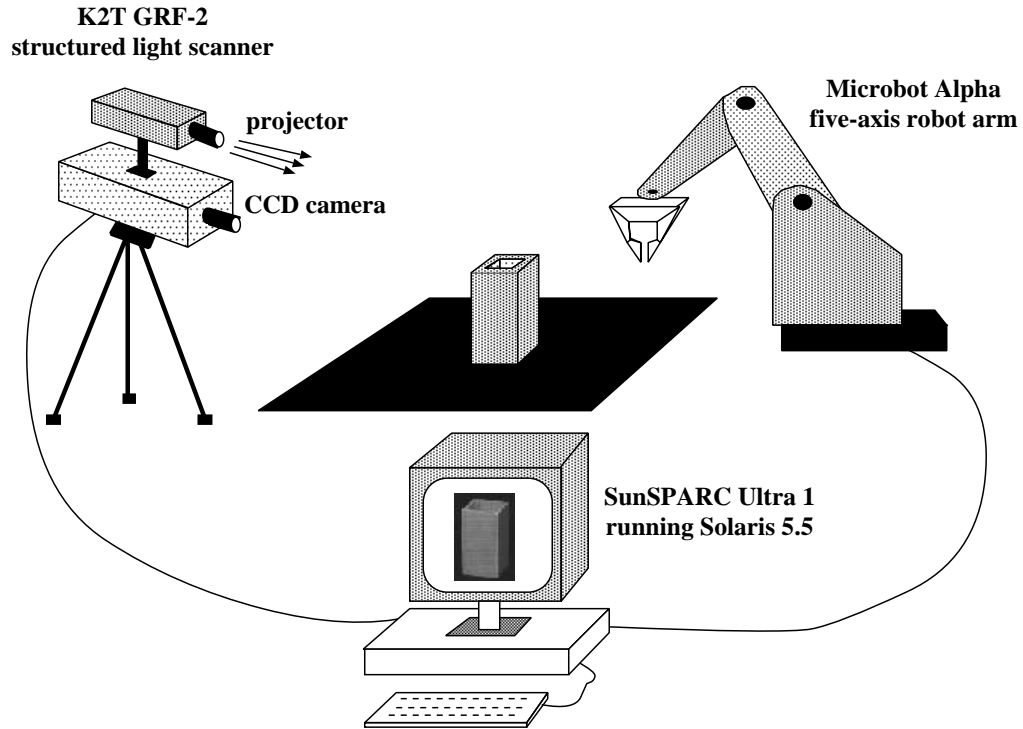


Figure 3. Experimental setup for the **GRUFF-I** system. Equipment includes a Sun workstation, a Microbot robot arm, a K2T GRF-2 structured light scanner and circuitry to transmit/receive sensor signals.

640 columns can be digitized into on-board memory at a maximum rate of 30 frames per second [49, 50].

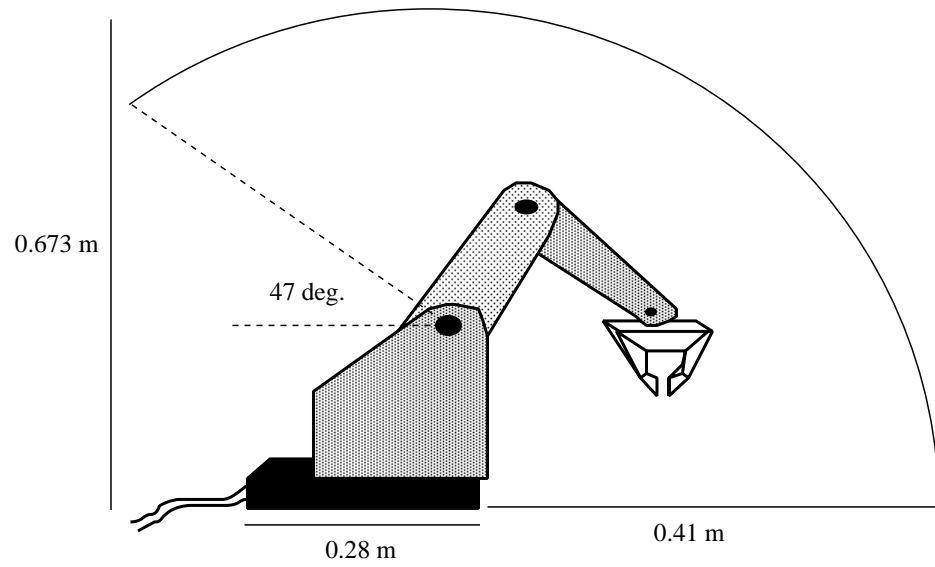
The scene area consists of a section of a table with a dark surface to diminish the effects of inter-reflectance. The camera is located roughly 1.0 m from the scene with the light pattern projector mounted approximately 0.30 m above it. A baseline of at least 0.25 m is typical to image 0.15 to 0.20 m objects. Increasing the baseline can improve range precision, but may lead to scene occlusion on some objects due to shadowing. Decreasing the baseline results in poor range data due to insufficient triangulation distance. The field-of-view limitations of this camera required some test objects to be scaled to roughly 1/10 of their normal size, such that the objects fit within a workspace centered by a 0.075 m cube.

The robot component uses a Microbot Alpha robot arm. This is a five-axis robotic arm with a cable-operated mechanical gripper. The arm has an open-loop

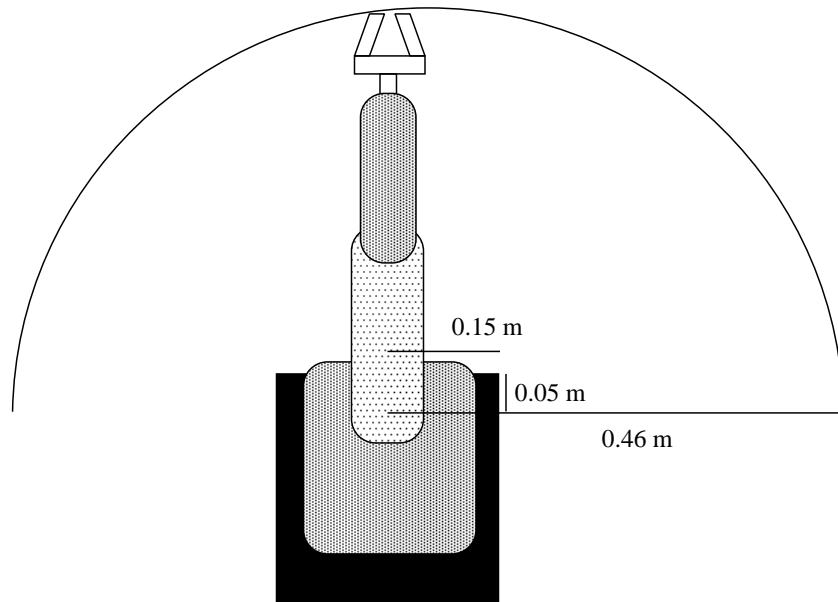
coordinated point-to-point control system with the capability of automatic homing. Communication between the Sun workstation and the robot arm is done via an RS-232 serial link running at 9600 baud. The robot can be moved with a maximum speed of 0.508 m/s and is capable of lifting payloads up to 0.681 kg. The gripper can grasp objects up to a width of 0.0762 m with a programmable force of 4.33 - 30 Newtons. The operating envelope constraints of the arm system are shown in Figure 4. The overall positioning accuracy of the arm is roughly 0.003-0.005 m.

3.2 Overview of GRUFF-I Subsystems

To initiate the **GRUFF-I** system, the vision component is invoked to snap intensity images of the scene every 10 seconds. This timing was chosen simply to give a user adequate time to place an object in the scene. The workspace is then monitored for movement, such as when an object is placed in the area. Movement is detected whenever there is a difference of 100 or more pixels between successive intensity images of the scene. Image subtraction is performed on the before and after images, using a threshold of 10 greyscale levels. If motion is detected, the vision component is invoked to acquire a range image. With a structured light scanner such as the one used in this setup, this actually involves taking a set of intensity images as a sequence of light patterns is projected on the scene area (see [103] for additional details). Depth (i.e. the range image) is then recovered via triangulation, with an accuracy of 0.001 - 0.002 m. Once the range image of the scene has been stored, control passes to the Model Building Subsystem.



(a) Side view



(b) Top view

Figure 4. Operating envelope constraints for the Microbot robot arm. These constraints limit the usable workspace area (adapted from [66]).

3.2.1 Model Building Subsystem

The Model Building Subsystem involves three stages, as shown in Figure 5-(a), and is based on two algorithms developed by Hoover, *et al.* [43, 45]. The first stage involves segmenting the range image into approximate face regions. The second stage fits planar surfaces to these regions and connects them together to obtain a 3-D polyhedral boundary representation model (defined in terms of faces, vertices and edges) of any object in the workspace. Finally, the third stage checks the validity of the recovered 3-D model.

An example of the input and output data formats used with this subsystem is shown in Figure 5-(b). The speckled areas in the range image of Figure 5 represent “shadow” or noise pixels. Shadow areas occur when using a structured light scanner, whenever the camera views a point which the projector doesn’t illuminate. Such areas tend to occur when the surface of the object obstructs the projector patterns. In this case, the seat of the chair obstructs the light patterns from being projected below it. The speckled noise areas on the seat itself are a result of the reflectance properties of the wood material. The model building algorithm does not require a perfect segmentation and incorporates various heuristics to handle both types of pixels.

The final model acquired from the above steps is what Hoover describes as an Object Plus Unseen Space (OPUS) model [43, 45]. The OPUS model captures the visible real faces and introduces faces along the line of sight referenced as occlusion faces. The resulting OPUS model of the object is a solid model enclosing the object and its surrounding occluded space. Defined in terms of faces and vertices, these models have an average space requirement of about 10 KB each. Processing time is approximately 5-6 minutes for range image segmentation, 1-3 minutes for creating the boundary representation and several seconds to determine model validity. If a valid

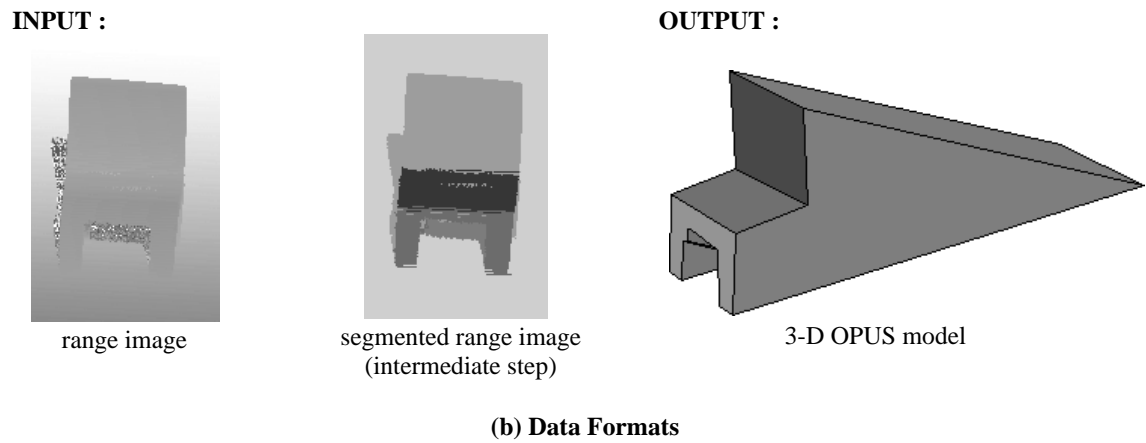
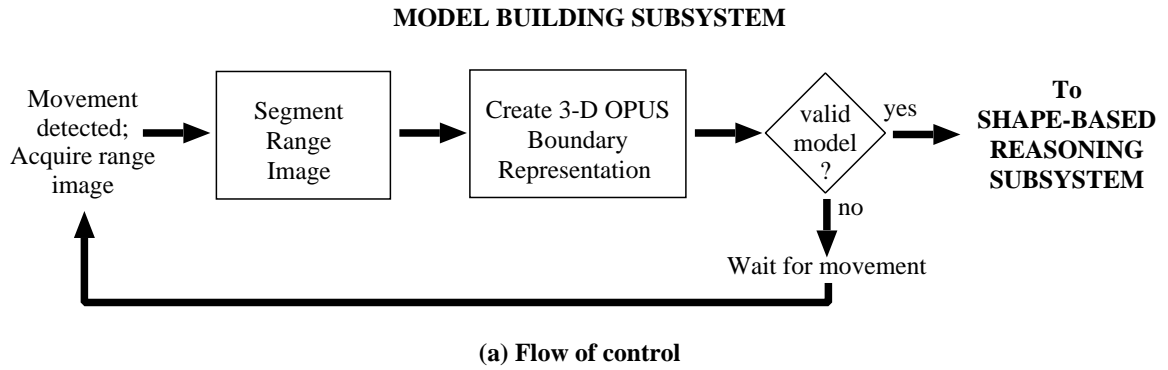


Figure 5. Flowchart for the Model Building Subsystem. In the processing for this subsystem, the range image is segmented into approximate face regions from which an OPUS model is created.

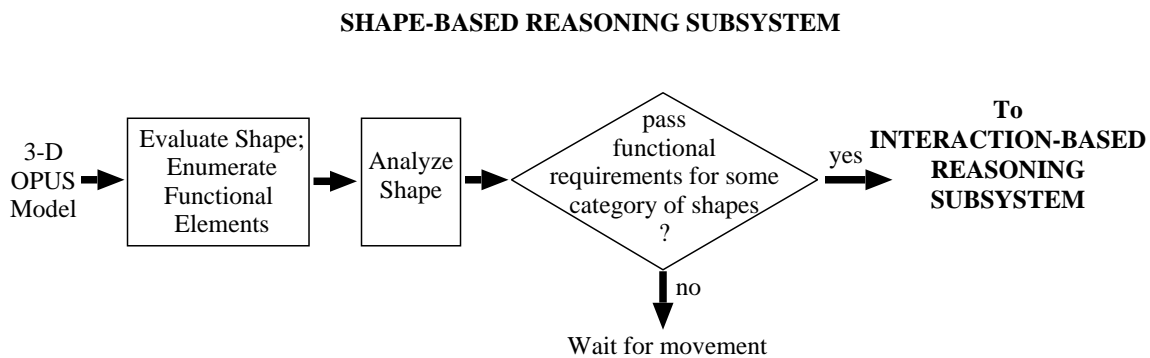
3-D model cannot be created, the system is re-initialized and the vision component is re-invoked to check for movement. If a valid model is formed, control passes to the Shape-based Reasoning Subsystem.

3.2.2 Shape-based Reasoning Subsystem

The Shape-based Reasoning Subsystem takes as input the 3-D boundary representation and applies concepts of physics and causation to determine, based on shape alone, if the object can tentatively function as a member of a particular generic category of objects, such as furniture or dishes. Each class of objects demands specific shape-based functional requirements be met, such as *provides sittability* or *provides stable support*, as shown in Figure 6. In order to use the functional requirements to recognize objects, the functional requirements must be converted into calls to appropriate operators which act on the shape to recover relevant information. There are six shape-based operators (termed *procedural knowledge primitives* or *PKPs*) which can be applied to determine functionality:

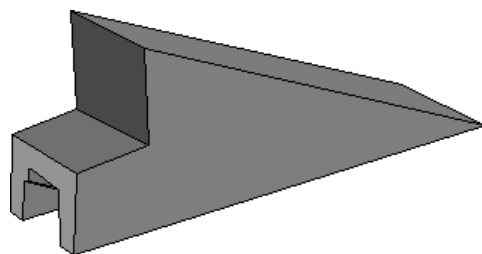
- **clearance** - determines the accessibility of some area within or around the shape.
- **dimensions** - determines the width, length, area, etc. of some surface.
- **enclosure** - determines locations of concavities in the shape.
- **proximity** - determines the relative location of two surfaces or characteristics of the shape.
- **relative orientation** - determines the orientation between two surfaces of the shape.
- **stability** - determines the stability of the shape in a given orientation.

As an example, the functional requirement of *provides cup/glass containment* includes the following PKP tests:

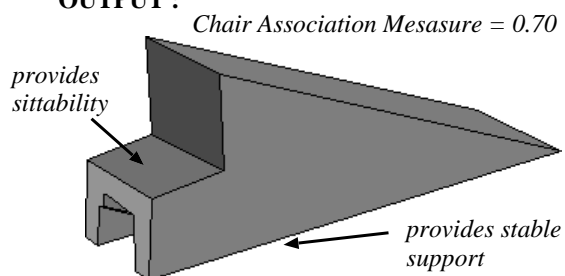


(a) Flow of control

INPUT :

3-D OPUS model
(uncategorized, unlabeled)

OUTPUT :

3-D OPUS model
(categorized, labeled)

(b) Data Formats

Figure 6. Flowchart for the Shape-based Reasoning Subsystem. This subsystem performs static function-based reasoning about purely geometric properties.

1. **stability** - check for stability of shape assuming no external forces except gravity and uniform density.
2. **enclosure** - check for a concavity in the current orientation.
3. **dimensions** - check for the appropriate volume of the concavity.
4. **dimensions** - check for the appropriate area of enclosing surface(s) for concavity.
5. **dimensions** - check for the appropriate width/depth of the concavity.
6. **dimensions** - check for the concavity being within the appropriate height range.
7. **clearance** - check for appropriate clearance above the enclosing area(s) of the concavity.

The final output of the Shape-based Reasoning Subsystem is a numerical value in the range 0.0 to 1.0 for category membership, and the attachment of labels to various 3-D surfaces, to indicate those areas of the model which are functionally significant for the specified category (termed *functional elements*). Processing time for shape-based reasoning is on the order of seconds to minutes, depending on the complexity of the 3-D model. Provided the shape satisfies the requirements of some category, control passes to the Interaction-based Reasoning Subsystem.

3.2.3 Interaction-based Reasoning Subsystem

The shape-based analysis in the previous subsystem assumes that the material properties of the object are sufficient. However, material property information is difficult to infer reliably from shape alone and is often necessary to confirm the usefulness of an object for a task. Therefore, a plan for interacting with the object is needed. The Interaction-based Reasoning Subsystem uses the labeled 3-D model from the Shape-based Reasoning Subsystem to produce a plan for interaction with the object, as shown in Figure 7. The robot arm is directed to absolute 3-D coordinates defined in terms of the robot frame of reference. In addition, analysis of subsequent

2-D intensity images taken during the interaction determines the success or failure of the interaction.

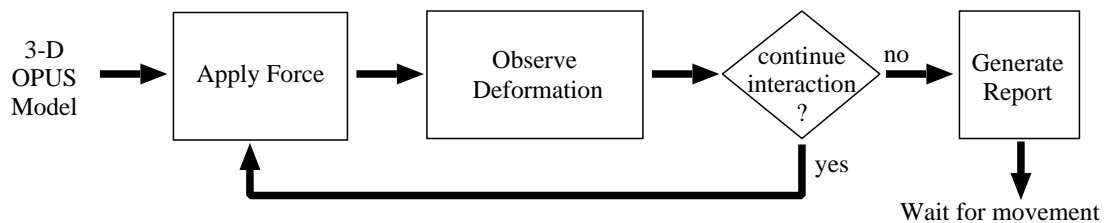
The creation of an interaction plan is accomplished by invoking a set of interaction-based primitive tests, as follows:

- **apply force** - guides the robot arm during the interaction, using operations such as:
 - GRASP/UNGRASP - grasp/ungrasp object at a given pair of points.
 - TRANSPORT - push, pull, raise, or lower the object.
 - POUR - pour a substance into the object at a specified point.
- **observe deformation** - determines if the object is deforming during the interaction, in comparison to an initial reference instance of the shape, using analysis of consecutive 2-D intensity images and robotic sensor feedback.

Just as the shape-based functional requirements (such as *provides sittability*) were tested by invoking a series of shape-based primitives, interaction-based functional requirements are confirmed for a given object using sequences of invocations of interaction-based primitives. For example, the requirement *confirm sittability* involves calls to **apply force** to position a weight on the potential seat and subsequent calls to **observe deformation** to determine if the interaction is successful.

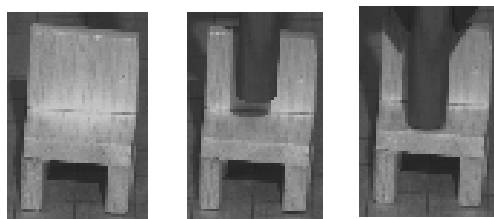
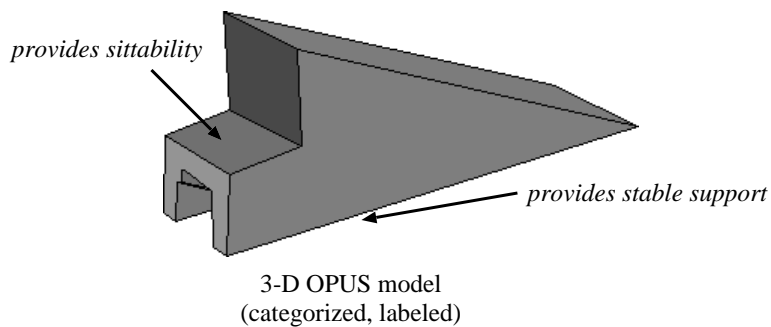
The goal of this interaction component of the system is therefore to determine the *suitability* or *sufficiency* of the material property requirements of the input shape, as opposed to the *identification* of its particular material composition. For example, based on interaction, the system would be able to determine that a paper maché chair is not functional due to its tendency to deform when weight is applied, as opposed to determining this by identifying that the object is composed of paper. Depending on the amount of interaction, completing an interaction plan requires 5 to 15 minutes. The final output of the system is a text report, which provides the object's numerical category membership value(s) and measures associated with how well it fulfilled each of the requirements for this category. The system then waits for movement, or the next object to be placed in the scene area.

INTERACTION-BASED REASONING SUBSYSTEM



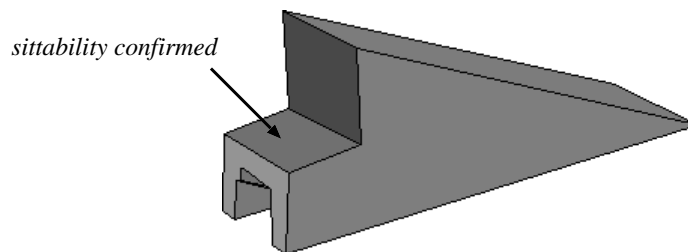
(a) Flow of control

INPUT : *Chair Association Mesasure = 0.70*



sequence of intensity images
(intermediate step)

OUTPUT :



(b) Data Formats

Figure 7. Flowchart for the Interaction-based Reasoning Subsystem. This subsystem performs dynamic function-based reasoning about the sufficiency of the material properties of the object for a specified functional requirement.

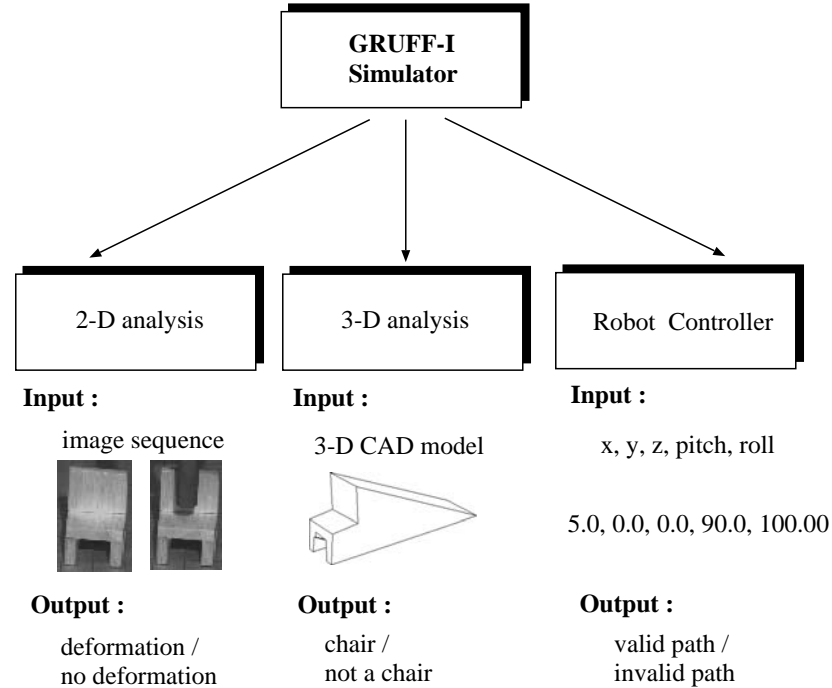


Figure 8. Overview of the simulation mode of the **GRUFF-I** system. This simulation mode can be used to test the individual subsystems.

3.3 Design of Simulation Mode

In the course of developing the overall system, a simulation mode was created, as shown in Figure 8. This mode was originally designed, and continues to be used, to test and improve the algorithms used for 2-D and 3-D processing and the robot controller. In addition, designing and developing these modules as independent units has allowed for running and testing the system with only partial components, and allowed for better handling of equipment substitutions, when necessary.

For example, when testing 2-D processing with the simulator, 3-D analysis and the robot controller are disabled. A sequence of intensity images with associated sensor data feedback derived from a previous experiment or acquired using a different system is then provided as input to the 2-D processing algorithm. The output indicates the success or failure of the interaction.

For testing 3-D processing, the simulator disables the 2-D analysis and the

robot controller. It then performs shape-based reasoning on a given model, just as the original **GRUFF** system would have [94, 95]. A user can select this option to test a new 3-D model of an arbitrary object against the categories known to the system. Alternatively, a user could modify an existing model, by changing its position, scale, orientation, etc. and then select this option to determine how this affects categorization of the shape.

Finally, for testing the robot controller, the simulator disables 2-D processing and tests the validity of any interaction paths generated by the 3-D algorithm. In this option, the output indicates if any coordinates in the path fall outside the operating constraints of the robot arm.

3.4 Dimensions Which Affected System Development

Using these hardware components and subsystems, three factors were considered important when designing, developing, and testing the **GRUFF-I** system, as shown in Figure 9. The axis for the **degree of interaction** for the system is based on the number of robot arms employed during the interaction. For simpler categories of rigid objects, one robot arm is sufficient. However, for more complex categories of articulated objects, a cooperating pair of robot arms is necessary. The **complexity of interaction** axis shows the number of different categories of objects possible for recognition (i.e. furniture, dishes and handtools), and includes categories of objects which are rigid and articulated.

Finally, the **feedback from interaction** axis shows examples of various sensing modalities which could be attached to each arm or could communicate directly to a workstation. As pointed out by Bogoni, *et al.*, while some functionalities may only be observable in one domain (e.g., self-stability is most readily observable using visual analysis), other functionalities lend themselves to combining evidence across

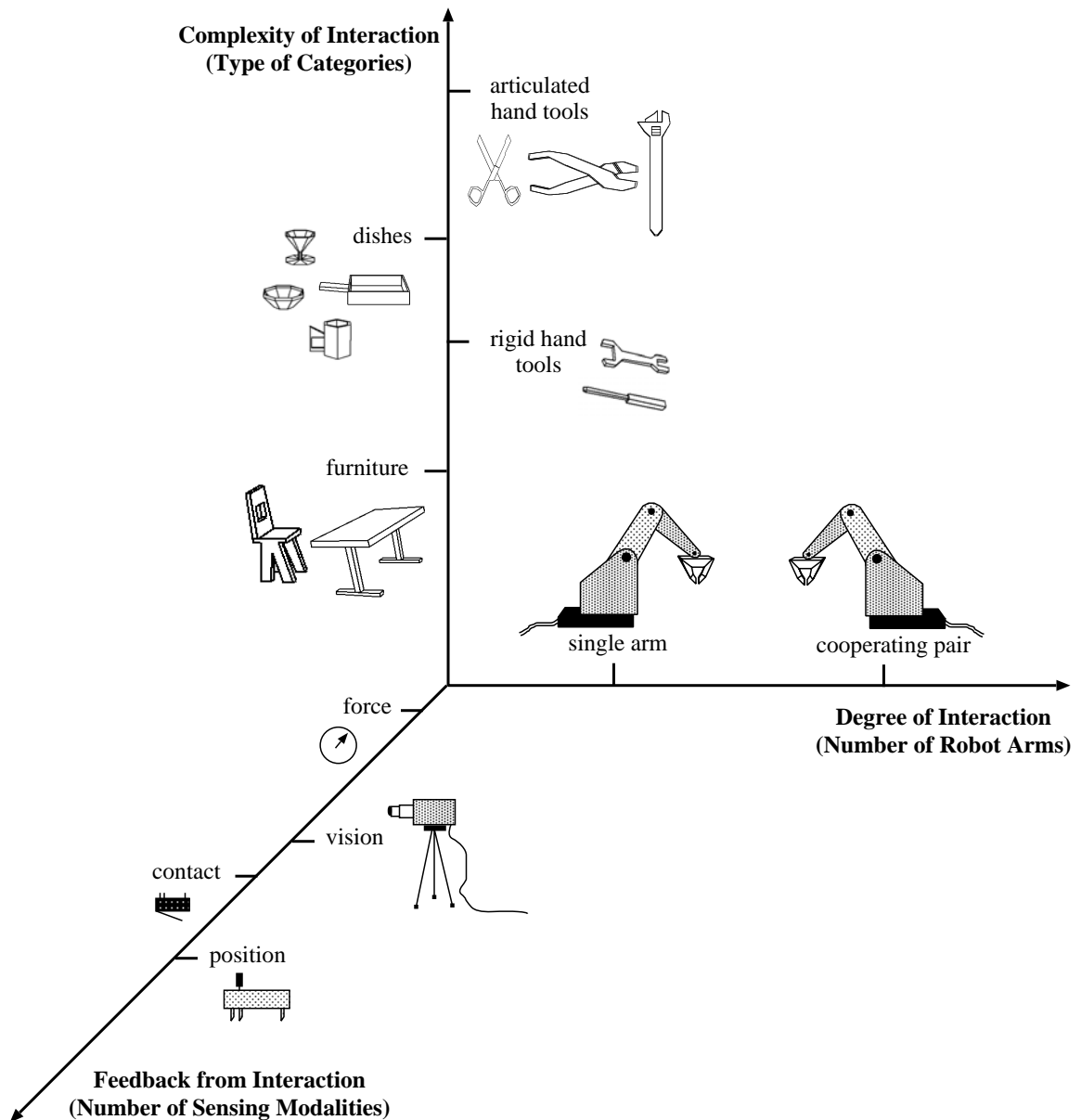


Figure 9. Factors which influence system complexity. The number of robot arms, the number of sensors and the number of categories considered influences the level of interaction possible and hence the overall complexity of the system.

sensing domains (e.g., the use of force and vision to confirm graspability) [10]. Additional tools required to confirm interaction also lie along this axis. For example, when testing *confirm sittability*, an external weight is needed to apply force to the potential “seat.” Similarly, when testing *confirm containment*, an amount of containment material is needed to be poured into the potential “concavity.”

For the work described in this dissertation, one robot arm is used. Sensor data is derived from the vision component and the built-in force sensors of the robot arm. Finally, one subcategory from the furniture domain (i.e. chairs) and one from the dishes domain of objects (i.e. cups) were selected for experimentation. As this is just a subset of the possibilities presented in Figure 9, the work in this dissertation can therefore be interpreted as a proof-of-concept to establish the feasibility of the framework upon which more complex systems could eventually be built. The following chapters examine each of the subsystems in detail and provide a set of experimental results from the integrated system.

CHAPTER 4

MODEL BUILDING

The Model Building Subsystem uses a structured light scanner and its associated software to acquire a range image of the scene. From this range image, a 3-D boundary representation of objects in the scene area is built. In the following sections, the step by step process leading to the final model is described.

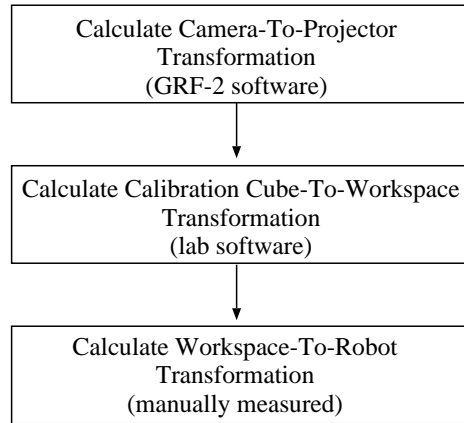
4.1 Calibration Procedure

The calibration procedure for the structured light scanner contains three separate stages, as indicated in Figure 10-(a). The objective of these stages is to determine the transformation from the cube-centered coordinate system to the robot arm-centered coordinate system, using the following equation:

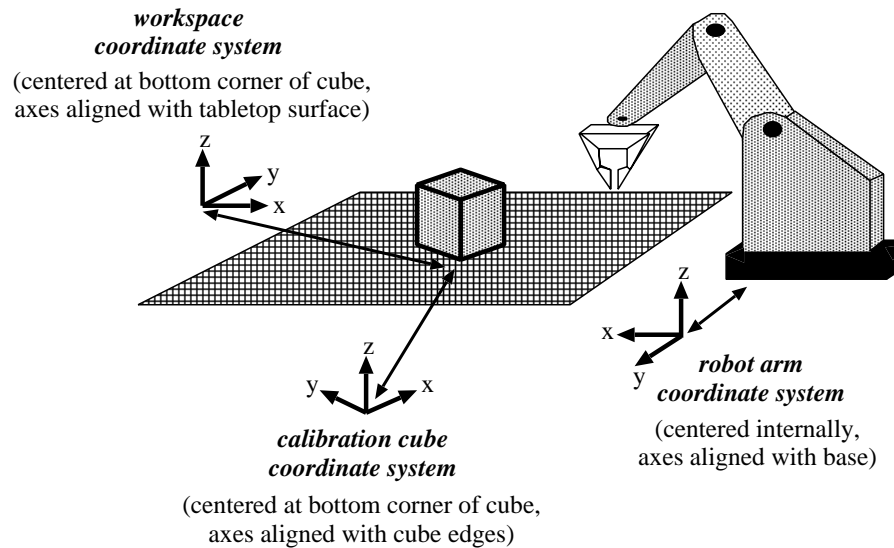
$$\begin{bmatrix} X_{robot} \\ Y_{robot} \\ Z_{robot} \end{bmatrix} = s\mathbf{R} \begin{bmatrix} X_{range} \\ Y_{range} \\ Z_{range} \end{bmatrix} + \mathbf{T} \quad (4.1)$$

where s is a scaling factor, \mathbf{R} is a rotation matrix and \mathbf{T} is a translation vector.

Each set of experiments begins by placing the GRF-2 calibration cube in the center of the scene, as shown in Figure 11-(a). This cube measures 0.075 m per side and contains seven circular inserts along each edge of each face. The cube is positioned in the scene such that three faces and their corresponding circles are visible to the camera and are covered completely by the projector light patterns.

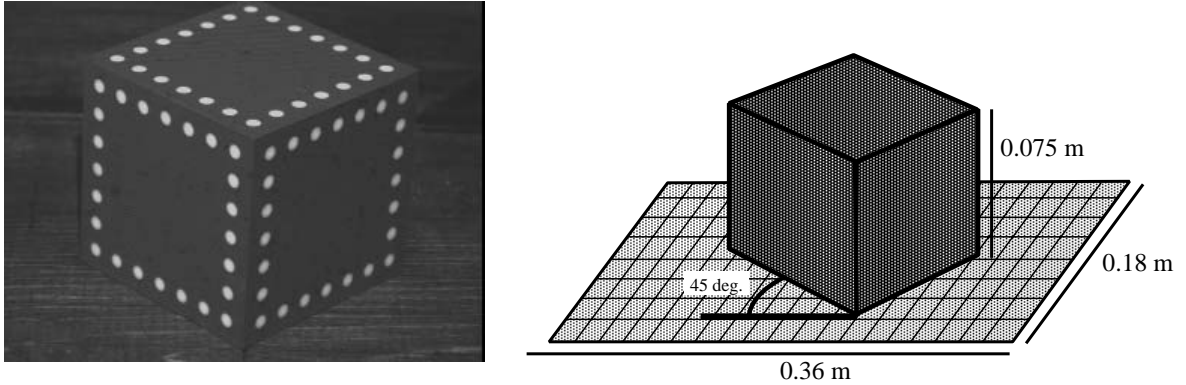


(a) Calibration procedure



(b) Coordinate systems

Figure 10. Calibration procedure. This procedure involves three steps in order to transform the 3-D data from the cube-centered coordinate system to the robot arm-centered coordinate system.



(a) Calibration cube placed in scene

(b) Dimensions of scene area (not to scale)

Figure 11. GRF-2 calibration cube. This cube contains circular inserts which are used to determine the camera-to-projector transformation.

The GRF-2 calibration software is then run [49, 50]. This software uses the circles on the calibration cube to calculate the camera-to-projector transformation. In order to threshold the cube from the scene, it is placed on a felt cloth, and the lighting in the room and camera aperture are adjusted until a single threshold can be used. For a typical calibration, a greyscale value in the range 30-40 was adequate to threshold the calibration cube, while a threshold of 60-75 was sufficient to threshold the circles on the cube. After this step, subsequently acquired range data is expressed with respect to an origin located at the cube's bottom corner, as indicated in Figure 10-(b).

Since the **GRUFF-I** system must ultimately use the 3-D data to direct the robot arm, two additional steps are taken in order to determine the transformation from the cube-centered coordinate system to the robot arm-centered coordinate system. In the second stage of the calibration, the felt cloth is removed from the scene and the cube is positioned in a specific 45 degree orientation along a rectangular grid in the scene, as shown in Figure 11-(b). Essentially, this step determines the rotation, scale, and translation parameters necessary to convert the range data from the cube-centered coordinate system to a workspace-centered coordinate system, as follows (adapted from [42]):

- rotation - The rotation matrix is determined based on the assumption that the the calibration cube is positioned at a known 45 degree angle on the grid.
- translation - The translation vector is determined as the zero-vector minus the 3-D position of the bottom front corner of the cube.
- scale - The *a priori* known length of one side of the calibration cube divided by the corresponding length in the intensity image provides a measure for the scaling factor.

The location of the origin and the orientation of the coordinate system of the workspace are also shown in Figure 10-(b).

In the last stage of calibration, the robot arm is placed at a predefined position on the workspace grid. The rotation and translation parameters necessary to convert from the workspace-centered coordinate system to the robot arm-centered coordinate system can be determined *a priori* (according to the positioning of the robot arm), and therefore in this stage, these parameters are manually entered into the system. Finally, the hand-drawn grid on the surface of the workspace which was used to align the cube as close as possible to the specified orientation is then used to test the calibration. This involves placing objects at specific locations on the grid and comparing the known location of the center of the object to the location determined from the range image data after transformation.

In summary, the structured light scanner and the robot arm each have their own independent coordinate systems. To be able to send the robot arm to pick up something recognized in the range image, the system must know how to be able to “go between” the coordinate systems. The necessary transformation equations are determined once, and thereafter 3-D points are in terms of the robot arm-centered coordinate system.

4.2 Range Segmentation Algorithm

Once these calibration steps have been completed, the camera, projector and robot arm are calibrated to work together. The **GRUFF-I** system is initiated, and intensity images are taken every 10 seconds until movement is detected. When motion is detected, a range image is acquired. A system call is then made to the YAR (**Y**et **A**nother **R**ange) segmentation algorithm, developed by Hoover, *et al.* [42]. Using the sequence of steps indicated in Figure 12, this algorithm analyzes the range image and attaches a label to groups of pixels which belong to the same surface. All experiments described in this dissertation used the parameters shown in Figure 13-(a) with the YAR segmentation algorithm. These parameters can be grouped into two categories with the following meanings:

- ImageTypeFlag, ReportFlag, SegTypeFlag - These parameters describe the data type for the K2T GRF-2 images and control the format of the generated output report and segmented image.
- MinRegionPixels, MaxPerpDist MaxPointDist - These parameters control the size of regions during segmentation.

A more detailed description of these parameters can be found in Appendix C.

4.3 OPUS Model Building Algorithm

Once the range image has been segmented, a system call is then made to the model building algorithm, also developed by Hoover, *et al.* [42]. A flowchart of this algorithm is provided in Figure 14. All experiments described in this dissertation used the parameters shown in Figure 13-(b) with this algorithm. These parameters can be grouped into four categories with the following meanings:

- ImageTypeFlag, ReportFlag, SegTypeFlag - These parameters describe the data type for the K2T GRF-2 images and control the format of the generated output report and segmented image.

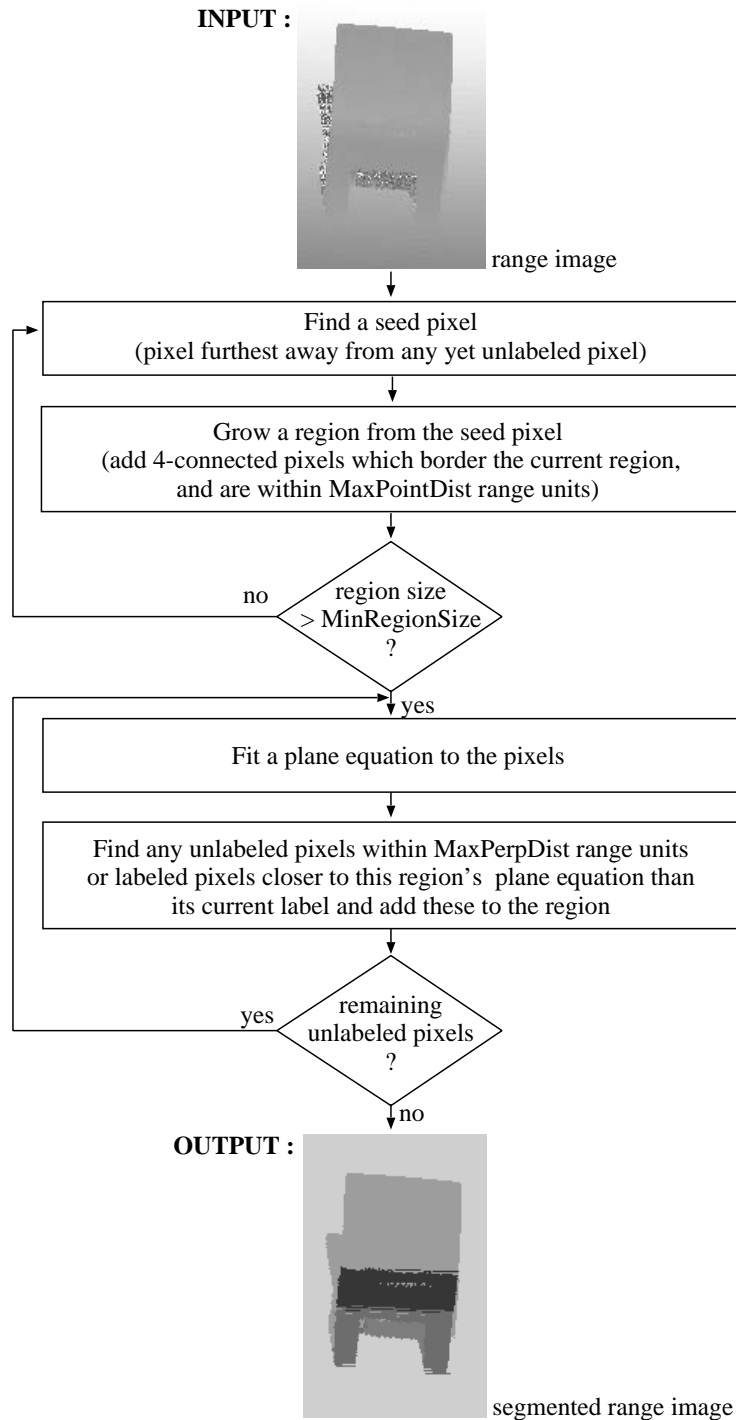


Figure 12. Flowchart for range image segmentation. The YAR segmentation algorithm follows a sequence of steps in order to label each pixel as belonging to a particular surface.

<p style="text-align: center;"><u>INPUT/OUTPUT FORMATS</u></p> <p>ImageTypeFlag = 9 (input format) ReportFlag = 2 (output format) SegTypeFlag = 0 (output format)</p> <p style="text-align: center;"><u>REGION SIZE</u></p> <p>MinRegionPixels = 100 pixels MaxPerpDist = 0.9 range units MaxPointDist = 0.9 range units</p> <p style="text-align: center;">(a) YAR parameters</p>	<p style="text-align: center;"><u>INPUT/OUTPUT FORMATS</u></p> <p>ImageTypeFlag = 9 (input format) ReportFlag = 2 (output format) SegTypeFlag = 0 (output format)</p> <p style="text-align: center;"><u>REAL SURFACE SIZE</u></p> <p>MinRegionPixels = 100 pixels Compactness = 0.5 (dimensionless) IncidentAngle = 85.0 degrees MaxPerpDist = 0.4 range units MaxPointDist = 0.8 range units MinEdgeLength = 15 pixel-sides</p> <p style="text-align: center;"><u>OCCLUSION PLANES</u></p> <p>LowEdgeDist = 0.5 range units HighEdgeDist = 1.0 range units JoinAngThresh = 10.0 degrees MinEdgeAngle = 160.0 degrees MaskFlag = 5 (dimensionless)</p> <p style="text-align: center;"><u>MODEL VALIDITY</u></p> <p>MinGlueDist = 0.06 range units ShortGlueEdge = 0.12 range units LongGlueEdge = 0.24 range units</p> <p style="text-align: center;">(b) OPUS parameters</p>
---	---

Figure 13. Parameters for the Model Building Subsystem. The parameters in (a) control range segmentation. Those in (b) control features of the final boundary representation.

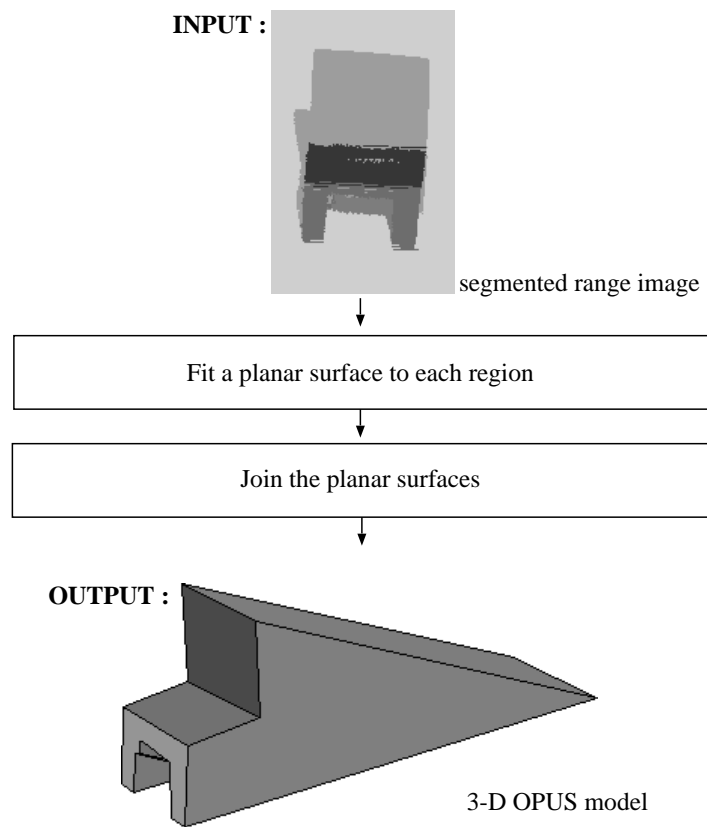


Figure 14. Flowchart for the OPUS model building algorithm. This algorithm creates the boundary representation from the segmented range image.

- MinRegionPixels, Compactness, IncidentAngle, MaxPerpDist, MaxPointDist, MinEdgeLength - These parameters control the size of real surfaces.
- LowEdgeDist, HighEdgeDist, JoinAngThresh, MinEdgeAngle, MaskFlag - These parameters control features of the occlusion planes.
- MinGlueDist, ShortGlueEdge, LongGlueEdge - These parameters are used to enforce planar boundary representation validity.

A more detailed description of these parameters can also be found in Appendix C.

Hoover described twenty different parameter sets for forty images taken with the K2T camera. These parameter sets were evaluated based on four criterion, as follows:

1. validity - This indicated the success of building a valid 3-D model.
2. average number of real surfaces - A smaller number of surfaces indicated the acquired model had a smaller ‘detail’ level.
3. average residual - This measure provided the average of the distances between the range points and the fitted surfaces. The average residual tended to increase with fewer surfaces.
4. average density - This measure was calculated by dividing the total number of modeled range points by the total area of real planes. This expressed the degree of ‘exaggeration’ in the reconstructed model.

The OPUS model building algorithm can therefore be run with various sets of parameters, in order to increase or decrease the level of “detail” or “exaggeration” in the final boundary representation. Detail in this sense is measured in terms of the size or number of surfaces in the constructed model. For the work in this dissertation, the parameter set selected resulted in a small average residual value and worked well on average, in terms of the number of valid models created. These criteria were considered important since having a small residual value indicated that the models would more closely match the actual range data, which would aid accuracy in path planning. A larger number of valid models would allow for more thorough testing of the remaining subsystems.

4.4 Model Validity Check

Provided that the segmentation and model building algorithms work on the range image, the final step for the Model Building Subsystem is to verify the validity of the 3-D model, using the following criteria (adapted from [62, 71, 75]):

1. Each vertex must be unique.
2. Each vertex must be used only once in a face.
3. Each face must have at least three vertices.
4. All faces must be planar.
5. Each edge must belong to exactly two faces.
6. Each inner loop of a face can intersect its corresponding face in at most one vertex. This test insures that an edge was not used once by the inner loop and the second time by its corresponding face.
7. Edges may only cross each other (intersect) at a point which is a vertex of the face which they define.
8. Two faces may only intersect each other at a common edge in the model.

Because the segmentation is not perfect (over-segmentation and under-segmentation are possible), the models are not always perfect, or clean. The model validity check ensures that despite appearances, the model meets standard geometric and topological constraints.

4.5 Post Processing of Acquired Model for the GRUFF-I System

Provided a valid model is created, there are several additional steps which must be taken, prior to passing control to the Shape-based Reasoning Subsystem. First, using the transformation equations obtained during calibration, the model is transformed into the robot arm-centered coordinate frame. In this frame of reference, the model's volume and center of mass are estimated. The center of mass is estimated

using the average of the vertices. The volume is estimated using the minimum and maximum extent of the object along each axis. In addition, the volume and center of mass of the concavities is measured.

4.6 Implementation Details

The YAR segmenter is written in C, with an executable size of approximately 108 kB. The OPUS model building algorithm is also written in C, with an executable size of about 145 kB. In the preliminary stages of system development, an alternative segmenter, the USF Segmenter (see [46]), was used as the front end to the Model Building Subsystem. However, this segmenter suffered from poor response time (on the order of tens of minutes per image), and therefore led to the selection of the YAR segmenter, which runs on the order of minutes per image.

4.7 Summary

In summary, the Model Building Subsystem (1) requires that the range scanner and robot arm use a common coordinate system, (2) produces a segmentation of the range image of the scene and (3) creates an OPUS 3-D shape model from the segmented range image. The final form of the OPUS model is in the robot arm-centered coordinate system. Provided this model is valid, control passes to the Shape-based Reasoning Subsystem.

CHAPTER 5

SHAPE-BASED REASONING

*...form ever
follows function...*

Louis Henri Sullivan (1896) [89]

The Shape-based Reasoning Subsystem analyzes 3-D models of objects from the categories furniture and dishes to determine if the shape satisfies the functional requirements of some category of objects. In the results described in this dissertation, this subsystem only considers a subset of its known categories. A more complete description of the entire set of categories, including the evolution of the system upon which this subsystem is based is provided in Appendix B.

The Shape-based Reasoning Subsystem outputs two important sources of information for each category evaluated. The first is an association measure between 0.0 and 1.0 for each object category. The subsystem also supplies a list of functional elements and evaluation measures. Functional elements are specific portions of the object shape which have explicit functional uses consistent with the target category. Evaluation measures are measures in the range 0.0 to 1.0 which reflect the strength of the association of a function label (functional property) to a functional element in the 3-D shape of an object. The following sections provide additional details on the structure of knowledge and processing used in this subsystem.

5.1 The Representation of Functional Knowledge

In order for the Shape-based Reasoning Subsystem to be able to recognize shapes in a particular category, the subsystem must hold a function-based definition of the object category. The *concept* of a set of shapes is therefore understood by the subsystem as a hierarchically organized tree of categories, as shown in Figure 15. This hierarchy uses Rosch's organization of object categories, as described in Chapters 1 and 2. Referring to Figure 15, at the superordinate category level, furniture and dishes are defined. At the basic level, examples include chair and cup/glass. Finally, at the subordinate level, examples include conventional chair and glass. Each category of objects has an associated set of function labels, or shape-based requirements that the shape must satisfy in order to be considered a member of a given category.

5.2 Static Shape-based Knowledge Primitives

In order to use functional requirements to recognize shapes, each functional requirement must be converted into calls to appropriate operators which act on the shape to recover relevant information. There are six operators (termed *procedural knowledge primitives or PKPs*) which can be applied to determine functionality: **clearance**, **dimensions**, **enclosure**, **proximity**, **relative orientation** and **stability**.

5.2.1 Primitive Clearance

The **clearance** primitive is invoked using the call:

$$\text{clearance}(\text{shape_element}, \text{clearance_volume}) \quad (5.1)$$

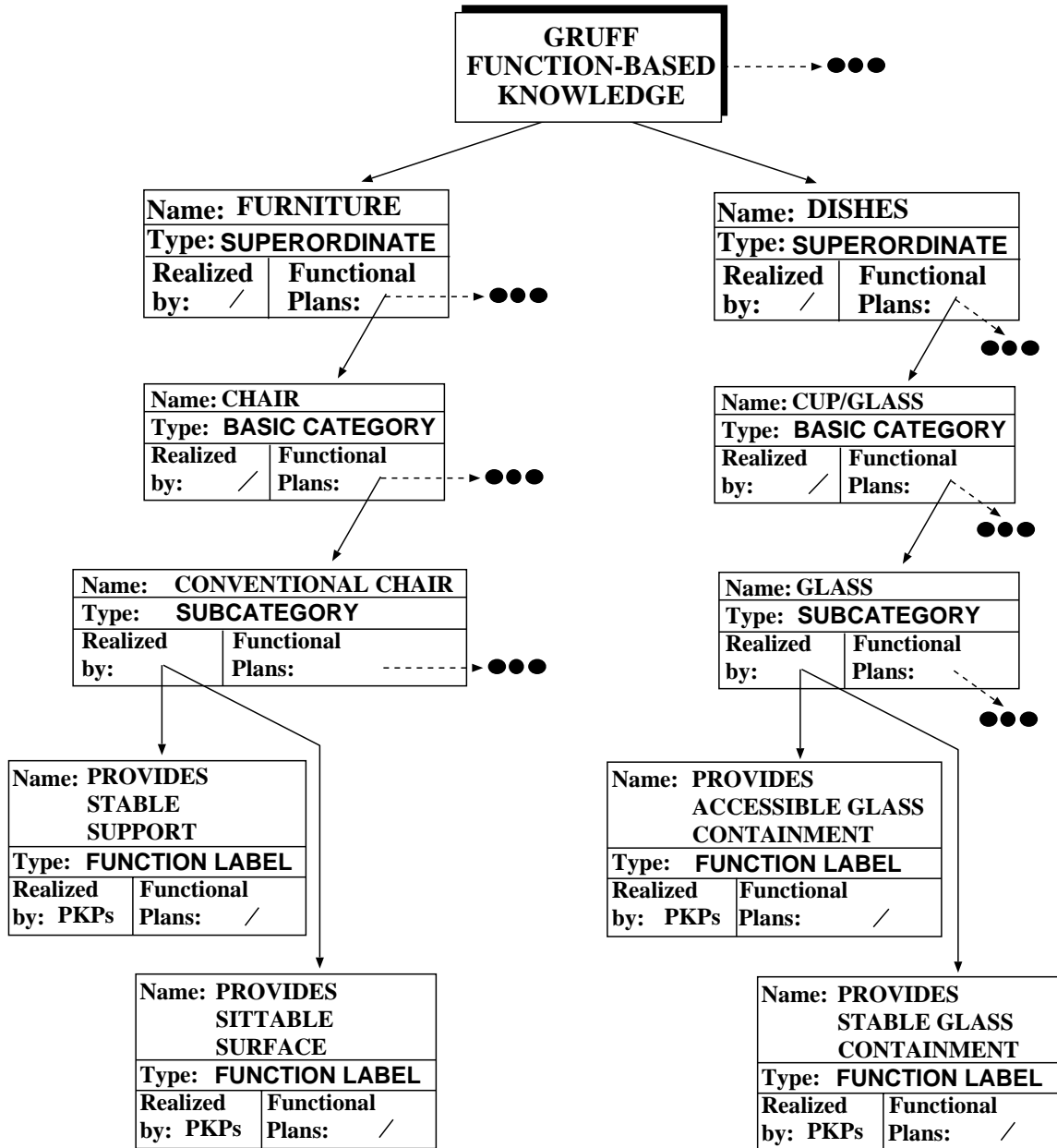


Figure 15. Example of functional requirements. Each (sub)category is represented by a set of functional requirements.

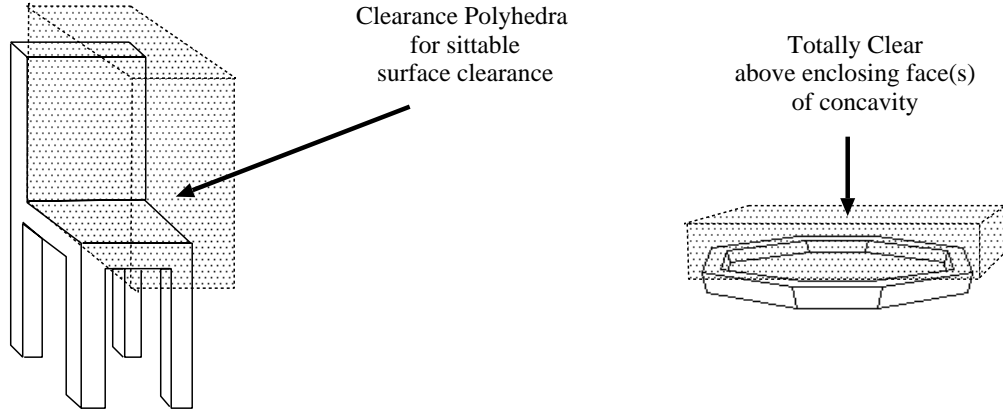


Figure 16. Knowledge primitive “clearance.”

as shown in Figure 16. This primitive can be used to check that there is a specified volume of unobstructed space in a particular location relative to a particular area of the shape. The *clearance volume* is represented by a clearance polyhedron which is specified by a set of faces and vertices. The *shape element* parameter is the 3-D shape of the object. The clearance test is implemented by checking for intersections between the object shape and the clearance polyhedron. The evaluation measure returned by this primitive is 1.0 if the volume specified is unobstructed, or 0.0 if it is obstructed.

5.2.2 Primitive Dimensions

The **dimensions** primitive is invoked using the call:

$$\text{dimensions}(\text{shape_element}, \text{dimension_type}, \text{range_params}) \quad (5.2)$$

This primitive can be used to determine, for example, if the width or depth of a surface lies within a specified range, as shown in Figure 17. The parameter *shape element* is an element of the object shape. The parameter *dimension type* specifies the type of measurement to be made, which can be one of the following:

- AREA, measured as units of square meters enclosed by the boundary of a face.
- CONTIGUOUS AREA, measured as the percentage of area enclosed by 2-D boundary that actually represents the object rather than “holes.”

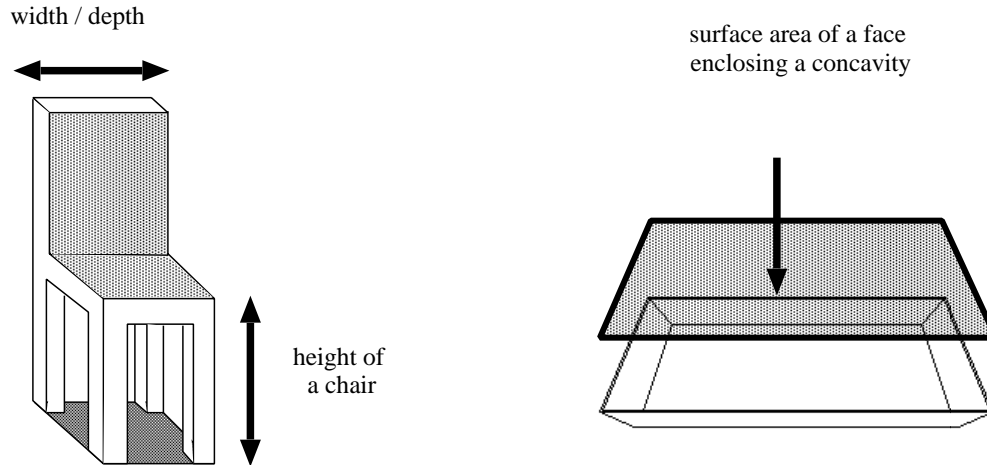


Figure 17. Knowledge primitive “dimensions.”

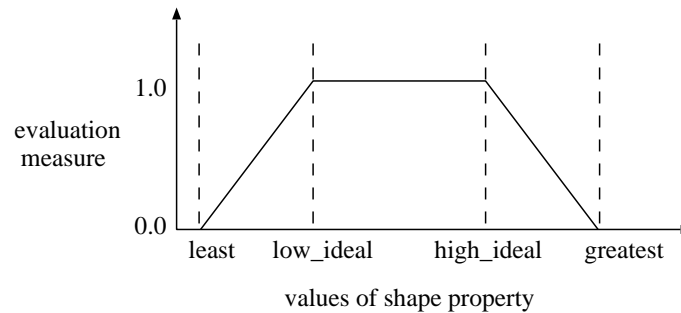


Figure 18. Using shape properties to calculate an evaluation measure.

- DEPTH, measured along the Z axis of the coordinate system.
- HEIGHT, measured along the Y axis of the coordinate system.
- WIDTH, measured along the X axis of the coordinate system.
- VOLUME, measured as the cubic meters of volume enclosed by a 3-D boundary.

For this primitive, the evaluation measure is calculated using four *range parameters*: *least*, *low ideal*, *high ideal* and *greatest*, which specify the desired range of values for the measurement. These parameters form a trapezoidal measure function as depicted in Figure 18. Any value between *low ideal* and *high ideal* results in a measure of 1.0. Values outside of this range but between *least* and *greatest* fall off linearly to 0.0.

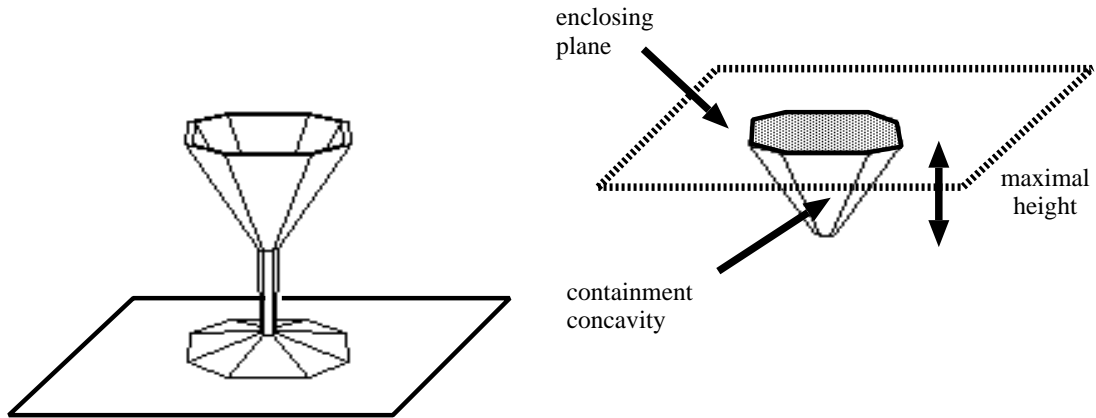


Figure 19. Knowledge primitive “enclosure.”

5.2.3 Primitive Enclosure

The **enclosure** primitive is invoked as follows:

$$\text{enclosure}(\textit{orientation}, \textit{concavity}, \textit{enclosing_plane}) \quad (5.3)$$

This primitive is used to determine if there exists a concavity in the shape which can be “closed” by a single plane introduced parallel to the support plane in a given orientation (see Figure 19). Because an infinite number of planes could be introduced at different levels, each enclosing a different volume, it is desired not only to confirm that a concavity can be enclosed but also to find the maximal volume that can be enclosed. The *concavity* parameter indicates a set of object faces which comprise a concavity of the object. The *orientation* parameter specifies the particular orientation to be tested. Finally, the *enclosing plane* parameter indicates the level at which the concavity is to be tested for enclosure. This primitive returns an evaluation measure of 0.0 if no maximal concavity can be found for the given orientation, or a measure of 1.0 along with the list of potential containment concavities and enclosing planes.

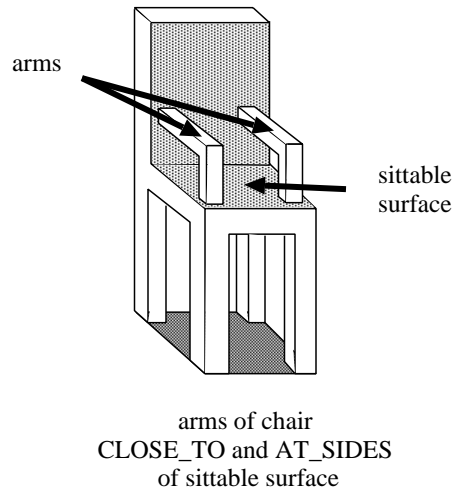


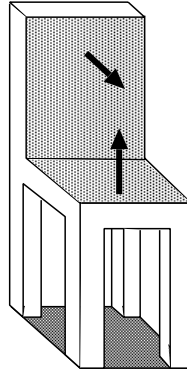
Figure 20. Knowledge primitive “proximity.”

5.2.4 Primitive Proximity

The **proximity** primitive is invoked as follows:

$$proximity(pair_type, relation, shape_element1, shape_element2, range_params) \quad (5.4)$$

This primitive can be used to check qualitative relations between faces or areas of the object, such as ABOVE, BELOW and CLOSE_TO, as shown in Figure 20. The parameter *pair type* indicates whether the comparison involves a point and a plane, a point and a line or two points. The parameter *relation* indicates the type of relation that is being evaluated. For example, a point may be evaluated for having a relation of ABOVE, BELOW or WITHIN, relative to a plane. The parameters *shape element1* and *shape element2* specify two elements of the object’s shape. As an example, this primitive can be used to check that the arms of a chair are close to the sittable surface.



back support approximately
orthogonal to seat support

Figure 21. Knowledge primitive “relative_orientation.”

5.2.5 Primitive Relative orientation

The **relative orientation** primitive is invoked as follows:

$$\textit{relative_orientation}(\textit{normal_one}, \textit{normal_two}, \textit{range_params}) \quad (5.5)$$

This primitive determines if the angle between the normals for two surfaces (*normal one* and *normal two*) falls within a desired range (see Figure 21). The desired relation is specified in terms of an appropriate range for the value of the difference between the surface normals of the two surfaces. The angle between the two normals is computed and this angle is checked against the specified range. The specified range may represent, for example, the requirement for the sittable surface of a chair to be approximately parallel to the support plane.

5.2.6 Primitive Stability

Finally, the **stability** primitive is invoked using:

$$\textit{stability}(\textit{shape}, \textit{orientation}, \textit{applied_force}) \quad (5.6)$$

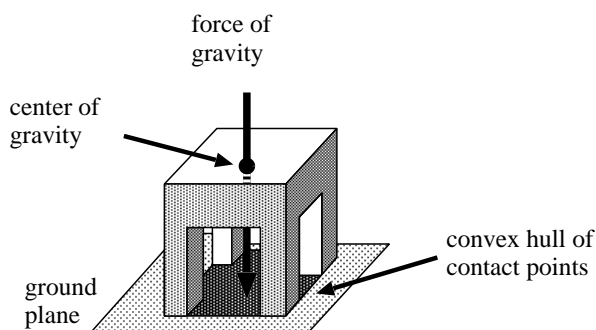


Figure 22. Knowledge primitive “stability.”

This primitive checks that a given shape is stable in the given orientation, with a (possibly zero) force applied to its center of mass (see Figure 22). The *shape* parameter is the 3-D shape of the object. The *orientation* parameter specifies the orientation to be tested. The *applied force* parameter specifies a force to be applied to a given point on the object. This primitive can be used to test two types of stability, self stability and function stability. *Self stability* for a shape and orientation assumes no forces are applied, except gravity. *Function stability* for a shape and orientation assumes forces are applied according to how the shape is to be used.

It is assumed that the object has homogeneous density, so that the center of mass may be calculated directly from the shape description. The check for stability begins by computing the 2-D convex hull of the points of contact between the object and the support plane. A test is then made for whether the projection of the center of mass orthogonal to the support plane falls inside the 2-D convex hull of the contact with the support plane. If the projection does fall within the 2-D convex hull, then the object is stable in that orientation. If it does not, then the object is not stable. This primitive returns an evaluation measure of 1.0 if the shape is stable in the specified orientation, or a value of 0.0 if it is not.

It is important to note that these primitives are domain-independent and are not “model-based.” For example, there is no mapping to a prototypical representation of a chair seat, but rather a series of geometrical tests to confirm areas of “sittability.”

5.3 Integrating Shape-based Functional Evidence

A series of procedural knowledge primitives (PKPs) are invoked to operate on selected portions of the shape to determine if requirements are met. For example, when analyzing a 3-D shape description to determine if it can satisfy one of the functional requirements for the chair category, *provides sittability*, the subsystem will determine if there exists some surface within the shape with an appropriate size (an invocation of **dimensions**) which is essentially parallel to the ground plane (an invocation of **relative orientation**) and accessible (an invocation of **clearance**) for sitting on (an invocation of **stability**).

The output of each primitive invocation is an evaluation measure between 0.0 and 1.0 indicating the suitability of some portion of the structure. These measurements are then combined for each requirement. In addition, the measures for a set of requirements are combined to obtain an overall measure for category membership. The final *association measure* for the shape is accumulated in a manner such that if a required shape analysis returns an evaluation measure lower than the current association measure, it lowers the overall evidence for that (sub)category. Representative instances of three families of aggregation calculi [12] were evaluated and compared for performance on a set of example shapes for the object category “chair” in the original **GRUFF** system [87]. The operation which provided the best performance at the category level was the T-norm or conjunctive operation:

$$T(a, b) = a * b \tag{5.7}$$

However, when the subcategory being investigated is a refinement of its parent subcategory (in the way that a mug is a refinement of a cup/glass), the evidence gathered at the parent subcategory node is used in a way which raises the association measure

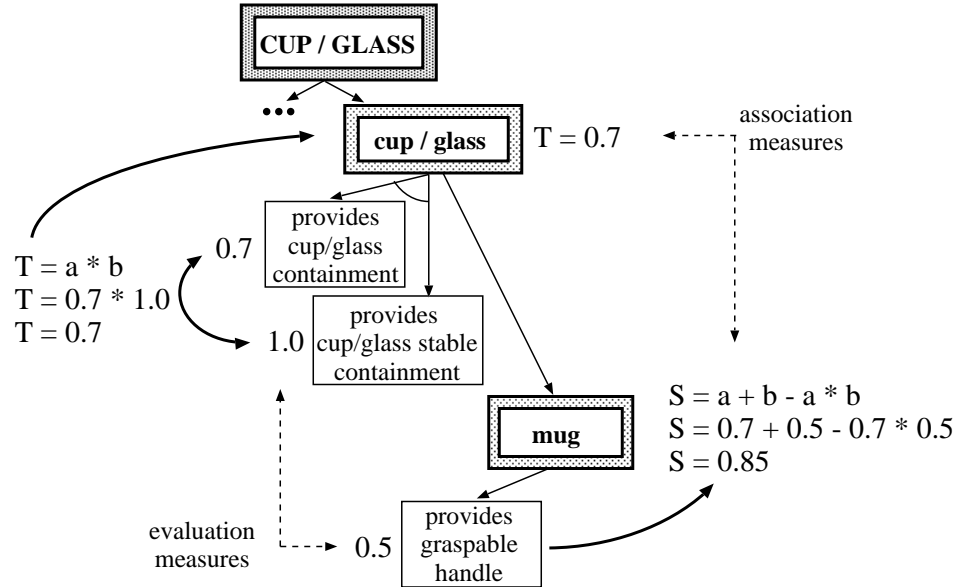


Figure 23. Example of combining measures. Aggregation calculi must be used to combine the *evaluation measures* of a series of individual primitive calls into a meaningful *association measure* representing the usefulness of the shape for a specific category.

calculated at the present subcategory node by some factor associated to the evidence gathered at the parent node, using a T-conorm operation:

$$S(a, b) = a + b - a * b \quad (5.8)$$

An example of these calculations is provided in Figure 23.

5.4 Deciding Categorization of the Shape

Passing shape-based reasoning suggests all shape-based required functional properties are confirmed for the object in the current view. In this case, provided the final measure is above a threshold, the subsystem passes control to the Interaction-based Reasoning Subsystem. An example of the results from this stage for a cup-like object is shown in Figure 24. Note that at this point in the processing, the subsystem has *hypothesized* how the shape may be used, assuming the material properties are

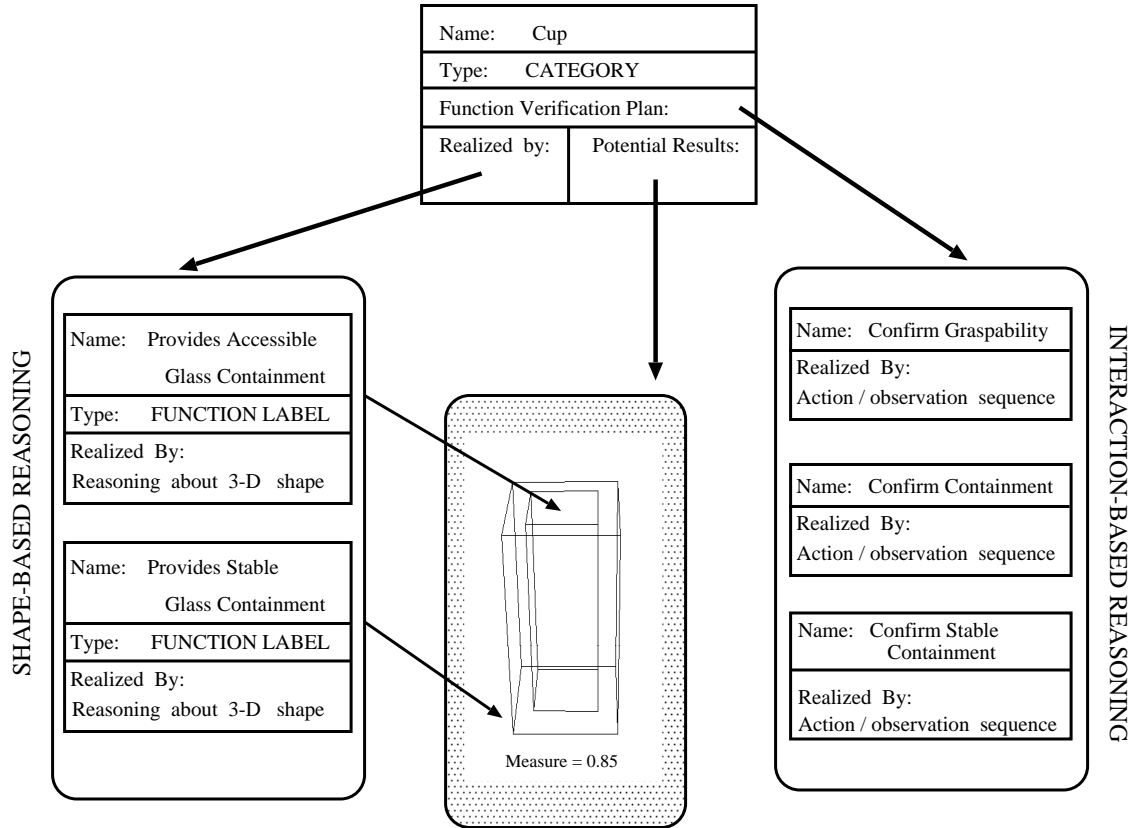


Figure 24. Functional properties for cup objects. Properties which involve only reasoning about abstract shape are shown on the left, as the *hypothesized functional plan*. Functional properties which reason about physical interaction are shown on the right, as the shape-suggested *function verification plan*.

sufficient. In the next stage, the Interaction-based Reasoning Subsystem uses the results in this *hypothesized functional plan* to initiate a *function verification plan* for interaction.

When analyzing OPUS models, objects may fail shape-based reasoning for several reasons. For example, if the current view of the object does not show a significant portion of its “functional” structure, then no functional property may succeed. Alternatively, some required shape-based functional properties may partially succeed, but with small measures, leading to an overall association measure close to 0.0. In both these cases, confirmation of function requires (1) a different orientation for the object in the current setup, (2) altering the setup, or (3) combining multiple

views of the same object, using the current setup. In the work described in this dissertation, the setup remains fixed, and thus only one view of the object is possible. In this case, if the Shape-based Reasoning Subsystem is unable to correctly categorize the object based on its current view in the scene, the user must manually reorient the object and allow the **GRUFF-I** system to acquire a new range image and model for analysis.

5.5 Implementation Details

The Shape-based Reasoning Subsystem is derived from the original **GRUFF** system reported in [83]. Written in C, this software has an executable program size of approximately 2.5 MB. Prior to integrating the **GRUFF** system into the Shape-based Reasoning Subsystem, it was tested with a total of 468 manually created complete 3-D shapes from the superordinate categories furniture, dishes and handtools, as reported in [86, 94, 95]. Appendix B provides a more complete description of this evolution.

Specific changes in the design which were necessary for the use of this subsystem with OPUS models were: (1) not considering occlusion surfaces as potentially functional, (2) not re-orienting the object model prior to shape-based analysis, and (3) altering the dimensional requirements of objects to be members of the conventional chair and glass categories to meet the constraints imposed by the workspace environment. Example PKP parameter values used in this subsystem are provided in Appendix C.

5.6 Summary

The Shape-based Reasoning Subsystem analyzes the 3-D shape of an object against a subcategory of objects (conventional chairs or glasses) and supplies two

sources of information. First, an association measure between 0.0 and 1.0 is provided to indicate the level of confidence that the object is a member of the specified category. Second, a set of labels for functionally significant areas of the object consistent with the target category is provided. These are the areas which the Interaction-based Reasoning Subsystem will attempt to interact with.

CHAPTER 6

INTERACTION-BASED REASONING

The Interaction-based Reasoning Subsystem uses information derived from the vision and robot components in order to direct interaction with the object in the scene to confirm functionality. This subsystem begins by instantiating a function verification plan, which contains a representation of how reasoning about physical interaction should occur, as shown in Figure 25. The goal of this subsystem is to use the 3-D information derived from the structured light scanner, along with the output of the shape-based analysis, to direct the robot arm to interact with particular locations on the object, defined in terms of the robot arm-centered coordinate system. The success or failure of this interaction is subsequently determined through analysis of 2-D intensity images and robotic sensor feedback obtained during the interaction. This chapter describes the primitives and the implementation of function verification plans for two test categories of objects, cups and chairs.

6.1 Dynamic Interaction-based Knowledge Primitives

Recall that in the Shape-based Reasoning Subsystem, a sequence of invocations of shape-based primitives (such as **clearance**, **dimensions**, etc.) is invoked to confirm static shape-based functional requirements such as *provides sittability*. Similarly, in the Interaction-based Reasoning Subsystem, a sequence of invocations of interaction-based primitives is needed to confirm dynamic functional requirements

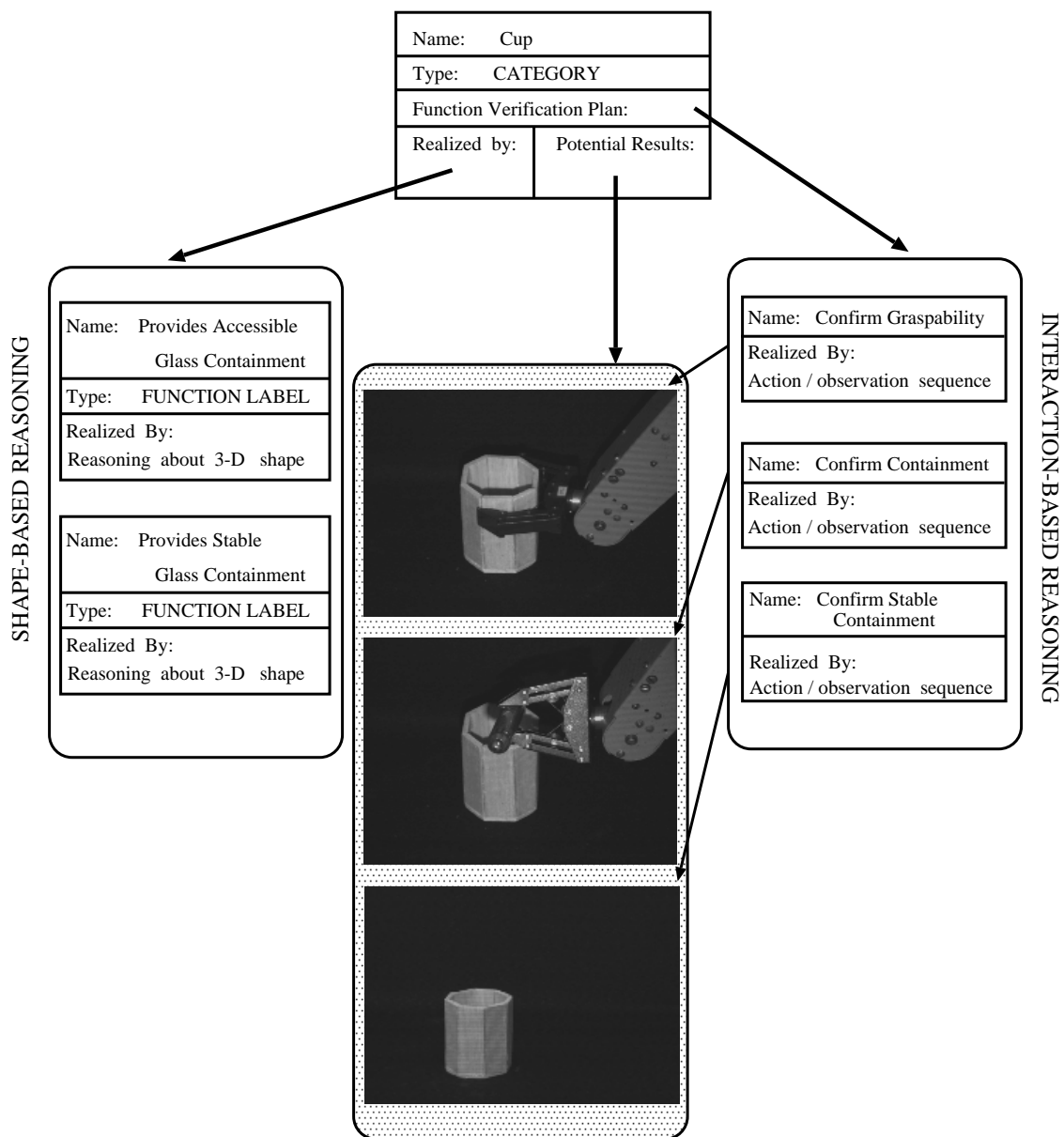


Figure 25. Verifying the functional plan for a cup. This involves instructing the robot arm to interact with the object in a manner consistent with its shape-suggested functionality.

such as *confirm sittability* or *confirm containment*. Two new primitives were created for this purpose, **apply force** and **observe deformation**. The primitive **apply force** is used to guide the robot arm during the interaction. This involves establishing a set of low level control operations for the Microbot robot arm. The primitive **observe deformation** is used to determine if the object is deforming during the interaction, in comparison to the reference instance of the shape. This involves acquiring a sequence of 2-D intensity images with the CCD camera and analyzing them with a set of low level vision operations, in addition to comparing robotic sensor feedback before and after the interaction. The following sections describe each of these primitives in detail and discuss how sequences of them can be combined to confirm interaction-based functional requirements.

6.1.1 Primitive apply force

The first step in designing the **apply force** primitive involved coding and/or testing a set of robot control operations for the Microbot Alpha robot arm, as follows:

- **open port** - This operation opens the serial port on the workstation and sets the parameters necessary to communicate with the robot arm (e.g., baud rate).
- **read location** - This operation reads the robot arm's current position, which is expressed as the number of motor steps for the base, shoulder, elbow, right and left wrist joints of the robot arm. These measurements are subsequently converted into Cartesian coordinates and expressed as an x, y, z position and a gripper separation distance (in meters). In addition, the pitch and roll angles of the manipulator (in degrees) are provided. These parameters are illustrated in Figure 26.
- **move arm** - This operation moves the robot arm to an absolute location or to a position relative to the arm's current position. The operation receives the position as Cartesian coordinates in the robot arm-centered coordinate system. These coordinates are converted to motor steps when sending the command to the robot arm.

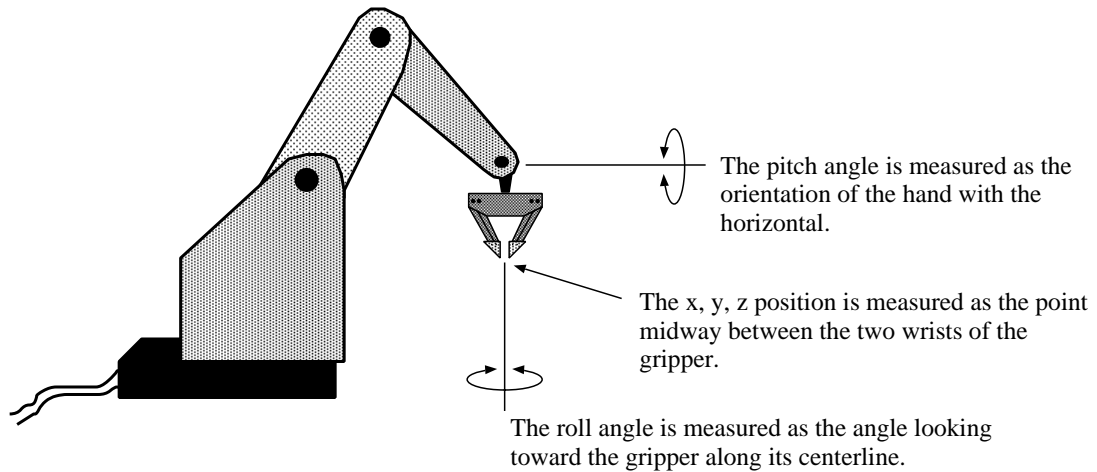


Figure 26. Positioning of the Microbot robot arm. The positioning is based on specifying parameters for the (x, y, z) coordinate, the pitch and roll angles of the gripper, and the gripper separation distance.

- **close gripper** - This operation closes the manipulator fingers. The gripper can be closed up to 160 motor steps after the two fingers of the gripper make contact, to build up a maximum force of approximately 30 Newtons, as shown in Figure 27. In practice, it was found that in order to grasp and pick up an arbitrary object without slipping, the gripper needed to be closed approximately 50 steps after contact. At this level, rigid objects (e.g., those composed of wood or Styrofoam) and most medium-weight material (sponge) objects would not deform, but non-rigid objects composed of lightweight materials such as paper showed deformation. At 160 steps after contact, medium-weight materials such as sponge showed deformation, while most rigid objects were unaffected. In the experiments described in this dissertation, a threshold of 125 steps after contact was used to grasp objects.
- **open gripper** - This operation is used to open the gripper a specified distance expressed in meters.
- **home robot** - This operation returns the robot arm to its home position. It is necessary to home the robot after approximately 4000 steps, in order to account for motor slippage. In addition, homing the robot arm allows the system to initialize it to a known position, from which other paths can be initiated. In the **GRUFF-I** system, it was sufficient to home the robot arm after confirming each functional requirement.

Just as a shape-based primitive like **stability** is built up by applying concepts of physics and causation to analyze the 3-D shape, the **apply force** primitive is

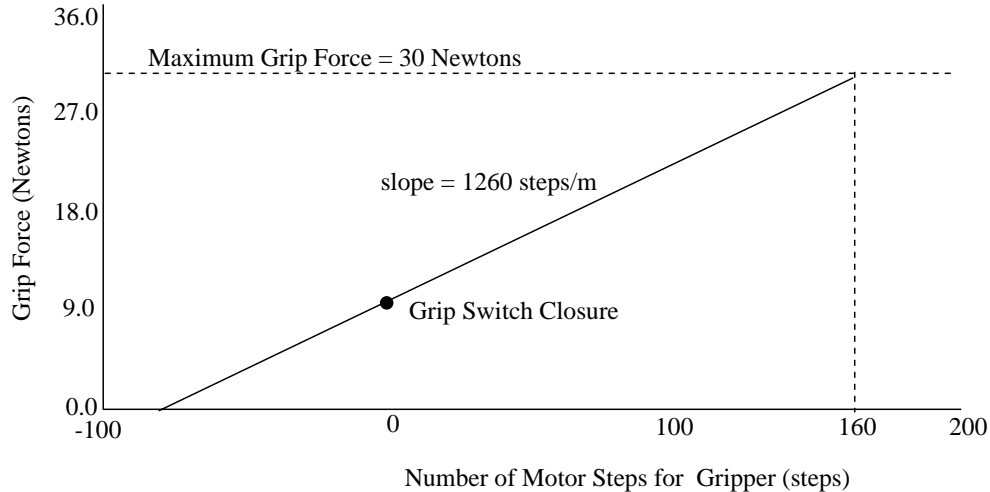


Figure 27. Force profile of the Microbot robot arm. After the left and right grippers make contact, a programmable amount of force can be built up to grasp the objects and avoid slippage (adapted from [66]).

built up by applying these robot control operations to the object in the scene. The primitive is invoked with a call such as the following:

$$\text{apply_force}(\text{path}, \text{force_type}, \text{force_amount}) \quad (6.1)$$

The parameter *path* is a linked list of points (x, y, z, pitch, roll and gripper separation distance) expressed in the robot-arm centered coordinate system. The parameter *force type* has four types, as follows:

- GRASP - When this type is used, the system opens the grippers and traverses the indicated *path*. At the final point, the grippers are closed with a force indicated by *force amount*.
- UNGRASP - With this type, the grippers are opened the maximum distance and the *path* is traversed.
- TRANSPORT - This type assumes the grippers have been placed somewhere relative to the object or around the object and is used to push, pull, raise, or lower the object by traversing the indicated *path*.
- POUR - This type is used to pour a containment material at some point in the workspace. This type assumes a container of pouring material has been retrieved from a pre-defined position, and the grippers are positioned relative to the destination point where the material is to be released. The parameter *path* indicates the sequence of points to traverse in order to release the material.

6.1.2 Primitive observe deformation

In order to monitor the results of the interaction, the additional primitive **observe deformation** is needed. The structured light scanner can supply intensity and range images, so the option exists to analyze the deformation in 2-D, 3-D, or both. As the previous chapters indicated, extra time is required to obtain a range image of the scene, since this actually involves taking a sequence of intensity images. In addition, the time to segment the range image and build a model is significant. Consequently, since reasonable response time is an important factor, 2-D analysis was selected for determining deformation. This subsequently involved designing and implementing a set of image processing operations, which could be used on successive intensity images, as follows:

- **snap image** - Snap an intensity image of the scene using the CCD camera of the structured light scanner component.
- **shrink image** - Reduce the image from a size of 480x640 pixels to a size of 240x320 pixels. This reduction allowed for a significantly faster execution of the remaining image processing operations.
- **median filter** - Smooth the image using median filtering. This type of filtering is applicable due to the “salt and pepper” nature of the noise in these images.
- **region growing** - Iteratively find the connected components of the given image, by labeling groups of connected pixels.
- **erosion** - Perform the morphological operation of erosion on an intensity image.
- **dilation** - Perform the morphological operation of dilation on an intensity image.
- **thresholding** - Threshold the image based on an intensity value.
- **centroid** - Calculate the centroid of a region by averaging the locations of its member pixels.
- **get boundary** - Calculate the boundary of a region.
- **find principal axes** - Determine the principal axes of a region.
- **image subtraction** - Determine the difference between two binary images.

The **observe deformation** primitive is built up from these image processing operations and is invoked as follows:

$$\textit{observe_deformation}(\textit{image1}, \textit{image2}, \textit{deformation_type}) \quad (6.2)$$

The parameters *image1* and *image2* are pointers to successively acquired intensity images. The parameter *deformation type* can be one of two types, as follows:

- POSITION - This type is used to obtain feedback from the gripper in terms of its current position and gripper separation distance. This can be compared to previous measurements to determine, for example, if the gripper is crushing the object.
- INTENSITY-STRUCTURE - This type performs analysis of intensity images taken during the interaction to locate object regions and determine corresponding object centroid movement, axis rotation and area differences.

For the analysis with the INTENSITY-STRUCTURE type, a predefined sequence of image processing steps is run on the set of images, using the low level image processing operations described earlier. This sequence can be broken into four stages: Preprocessing, Object Extraction, Feature Calculation, and Deformation Analysis, as shown in Figure 28.

During Preprocessing, smoothing the image helps to eliminate small noise spikes. Thresholding is then used based on the expectation that the object will be one of the brightest regions in the image. This constraint was imposed by the use of the structured light scanner which does not image black or highly reflective surfaces well [49]. During Object Extraction, the object region must be segmented from the background, from noise spike areas (e.g., points of light reflection on other objects, such as the robot arm), and finally from areas where additional pouring material has been added. This involves applying thresholding, region growing and a series of morphological operations to the image. For example, an erosion-dilation sequence with different-sized structuring elements is used to remove isolated noise pixels and to determine the location of any pouring material areas. This is followed

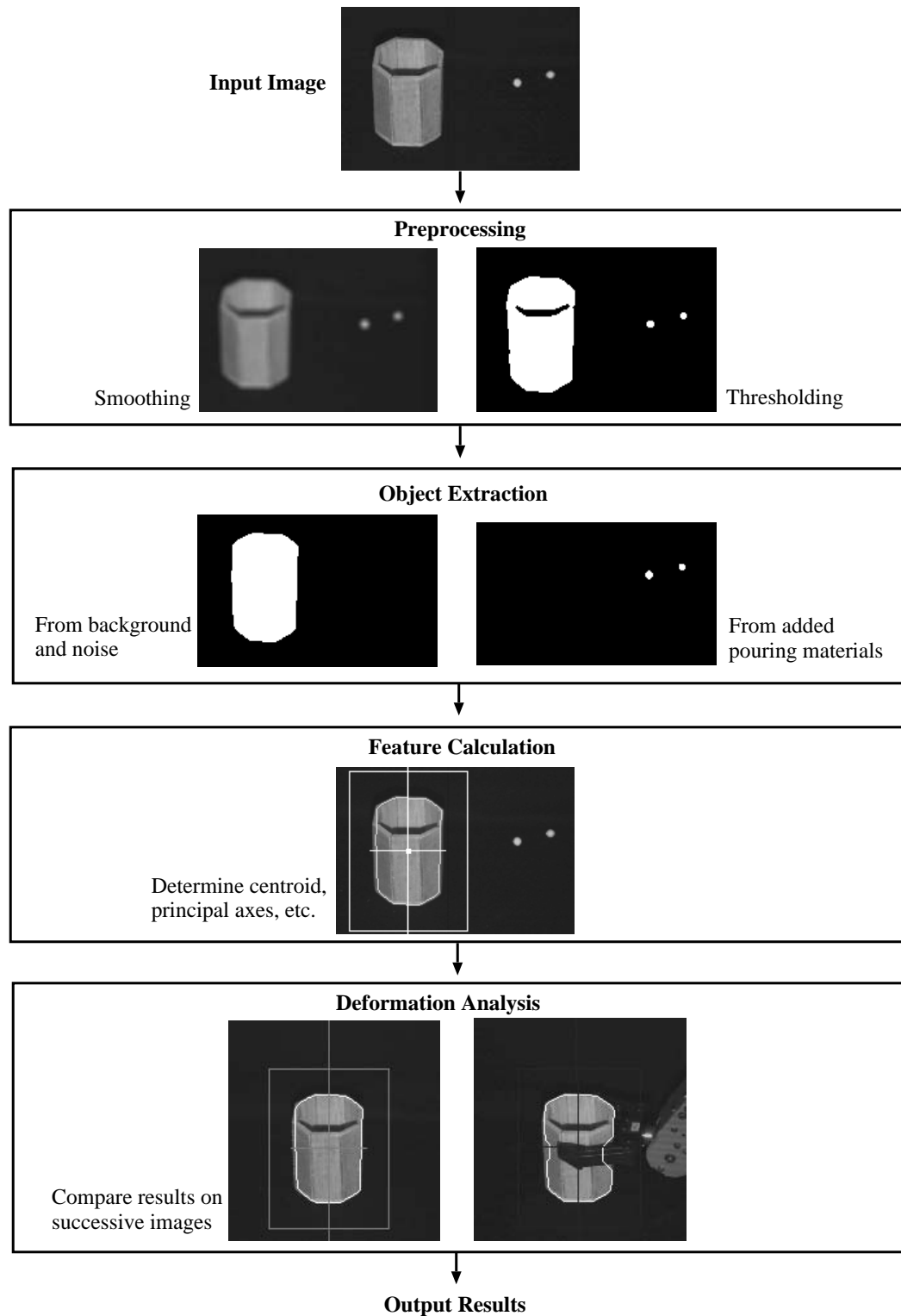


Figure 28. Analysis of deformation using intensity images. The image of the scene must be processed to extract the object and determine if deformation has occurred.

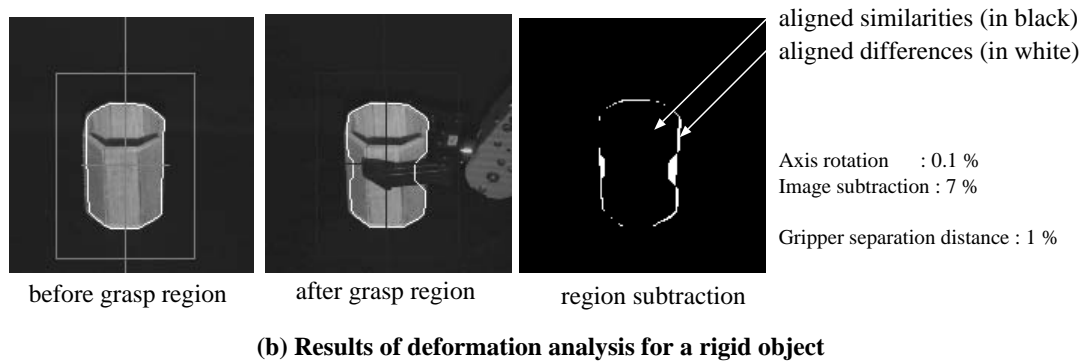
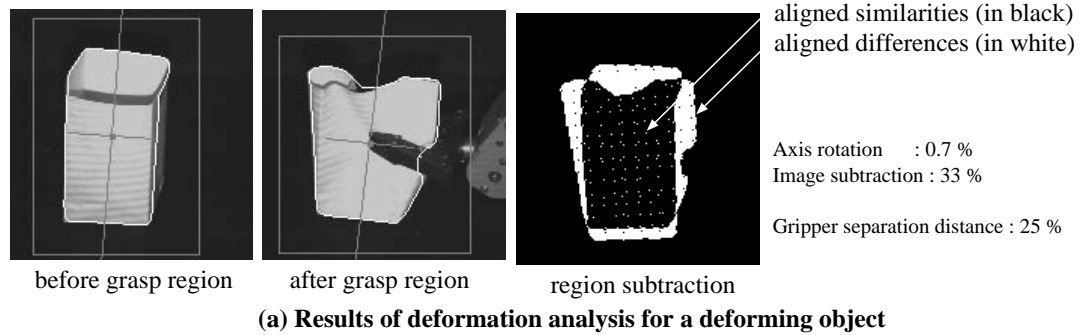


Figure 29. An example of image processing results. Once the object regions and principal axes are determined for each image, the *after grasp* region can be aligned to the *before grasp* region for image subtraction and comparison. Additionally, the expected and actual gripper separation distance and position before and after grasping the object can be compared.

by a dilation-erosion sequence which helps to fill back in object areas which become occluded as a result of interaction.

During Feature Calculation, the location and size of object regions, the number of regions, the orientation of the principal axes of object regions and locations of pouring substance regions is determined. Finally, in the Deformation Analysis phase, an analysis of the results of these steps compared to the results from the previously taken intensity image is made. As an example, Figure 29 shows the results of this analysis on intensity images taken during interaction with a rigid wooden cup and a non-rigid paper cup.

6.2 Interaction-based Functional Requirements for Category Cup

Given the implementation of these two primitives for monitoring interaction, the next step involves building the function verification plans (interaction-based requirements) for each of the categories of objects for which the system is to be tested. For the cup category, continuation into the Interaction-based Reasoning Subsystem implies that shape-based analysis of the current view has determined that the following shape-based functional requirements have been satisfied (shown in Figure 24):

- *provides accessible glass containment* - the shape has a concavity and is accessible for grasping. Labeled functional elements in the shape include the faces which define the concavity and the enclosing surface for the concavity.
- *provides stable glass containment* - the shape can provide stable containment, as determined by applying a point force at the center of mass of the concavity.

To confirm these requirements for the object in the scene, it is necessary to design the explicit interaction tests with the object which need to be confirmed in order for the object to remain a member of the cup category. These interaction requirements were designed as follows (shown in Figure 25):

- *confirm graspability* - the object can be grasped and lifted.
- *confirm containment* - the object can contain some substance.
- *confirm stable containment* - the object can continue to provide containment when lifted.

An example of a sequence of images acquired during confirmation of these requirements is shown in Figure 30.

6.2.1 Confirm graspability

The implementation of each interaction-based requirement requires defining it in terms of calls to the PKPs **apply force** and **observe deformation**. To confirm

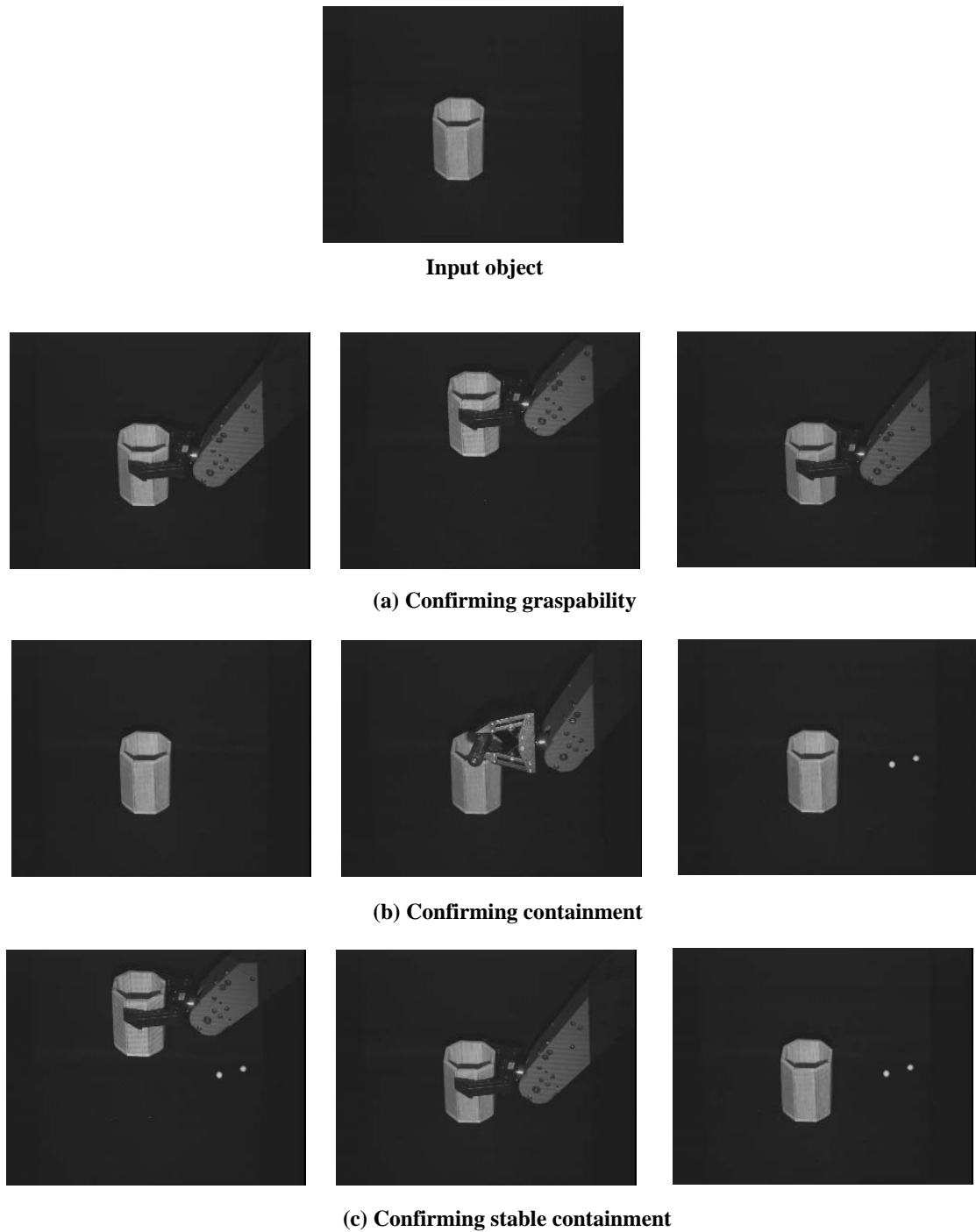


Figure 30. Interaction tests for a cup object. Confirmation of the object as a member of the cup category of objects involves three interaction tests: (a) confirm graspability, (b) confirm containment and (c) confirm stable containment.

graspable containment, the grippers must make contact with the object, pick up the object while maintaining contact and then put the object down. This requirement is implemented as follows:

- *apply force*(GRASP, ...) - Move to grasp the object in the scene.
- *observe deformation*(POSITION, ...) - Compare results of gripper location and separation distance before and after grasp.
- *observe deformation*(INTENSITY-STRUCTURE, ...) - Compare object areas before and after grasp.
- *apply force*(TRANSPORT, ...) - Traverse a path directed upwards from the current location.
- *observe deformation*(INTENSITY-STRUCTURE, ...) - Compare object areas before and after lifting.
- *apply force*(TRANSPORT, ...) - Traverse a path to return to starting location.
- *observe deformation*(INTENSITY-STRUCTURE, ...) - Compare object areas before and after setting the object back down.
- *apply force*(UNGRASP, ...) - Ungrasp the object.
- *observe deformation*(INTENSITY-STRUCTURE, ...) - Compare object areas before and after ungrasping.

For this requirement, the critical features of the object which define the grasp site are available from the identified functional elements labeled in the Shape-based Reasoning Subsystem. From these functional elements and the constraints imposed by using a two-fingered gripper, the system can determine the best path along which the approach for grasping can be made, as shown in Figure 31. The first step in the process of determining where to grasp the object is to form a list of all non-concavity defining faces in the 3-D model. This list is then searched to find those sets of faces whose normals are essentially anti-parallel to each other, meet perpendicular to the stability vector (0,0,1) and are also oriented perpendicular to robot arm vector (1, 0, 0). For each set of parallel faces which passes these constraints, the average point

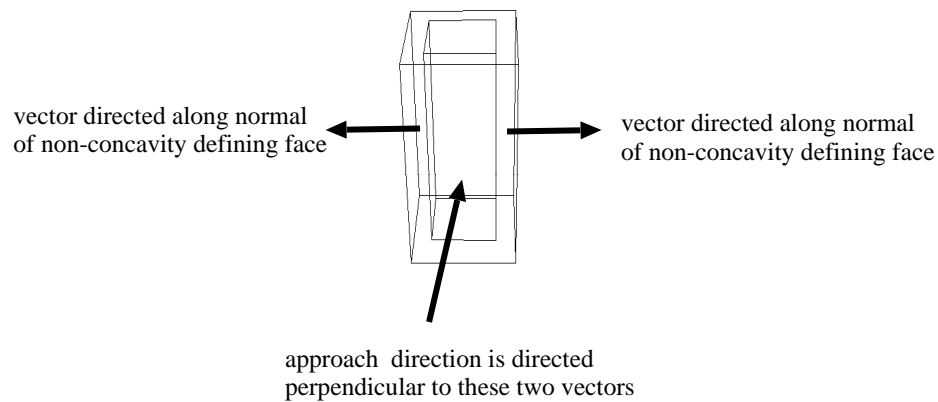


Figure 31. Path planning for cup objects. The approach path for these objects is constrained by the location of graspable surfaces and the operating envelope constraints of the two-fingered gripper.

of each surface is calculated to determine if the surfaces are separated by a distance attainable by the gripper.

If the surfaces are within the gripper distance threshold, the final step is to calculate the path to take to approach the sides of the object. From the surviving set of parallel faces, the system determines which edge of the two faces is closest for the robot to use as a guide for approaching. This is accomplished by searching for the closest edge which is parallel or anti-parallel to the stability vector.

6.2.2 Confirm containment

To confirm that the object can provide containment, approximately 0.002 L of distinctly colored beads (0.008 m in diameter) are poured into the concavity area using the center of the enclosing surface of the concavity (identified in the Shape-based Reasoning Subsystem) as the destination point. The sequence of calls to accomplish this is as follows:

- *apply force(GRASP, ...)* - Traverse a predetermined path to get a beaker containing the portion of beads.

- *apply force(TRANSPORT, ...)* - Position beaker relative to the destination point.
- *apply force(POUR, ...)* - Release material.
- *observe deformation(INTENSITY-STRUCTURE, ...)* - Compare images before and after pouring.
- *apply force(TRANSPORT, ...)* - Return beaker to its starting position.
- *apply force(UNGRASP, ...)* - Release the beaker.
- *observe deformation(INTENSITY-STRUCTURE, ...)* - Compare images before and after beaker removed from scene.

6.2.3 Confirm stable containment

To confirm stable graspable containment, the grippers must again make contact, pick up the object (maintaining contact), and then put the object down. This requirement is implemented with the same sequence of paired calls to **apply force** and **observe deformation** as was used with the *confirm containment* requirement. The objective in repeating this sequence of events is to ensure that the object will not fail to provide containment when lifted.

6.3 Interaction-based Functional Requirements for Category Chair

When an object has passed through the Shape-based Reasoning Subsystem for the category chair, the following requirements have been met (shown in Figure 32):

- *provides sittable surface* - some surface has been found which is accessible for sitting. Labeled functional elements include each surface which can provide sittability, along with an associated set of edges from these faces which have been determined to be clear for approaching to sit.
- *provides stable support* - the shape can provide sittability with a force applied to points on the potential seat.

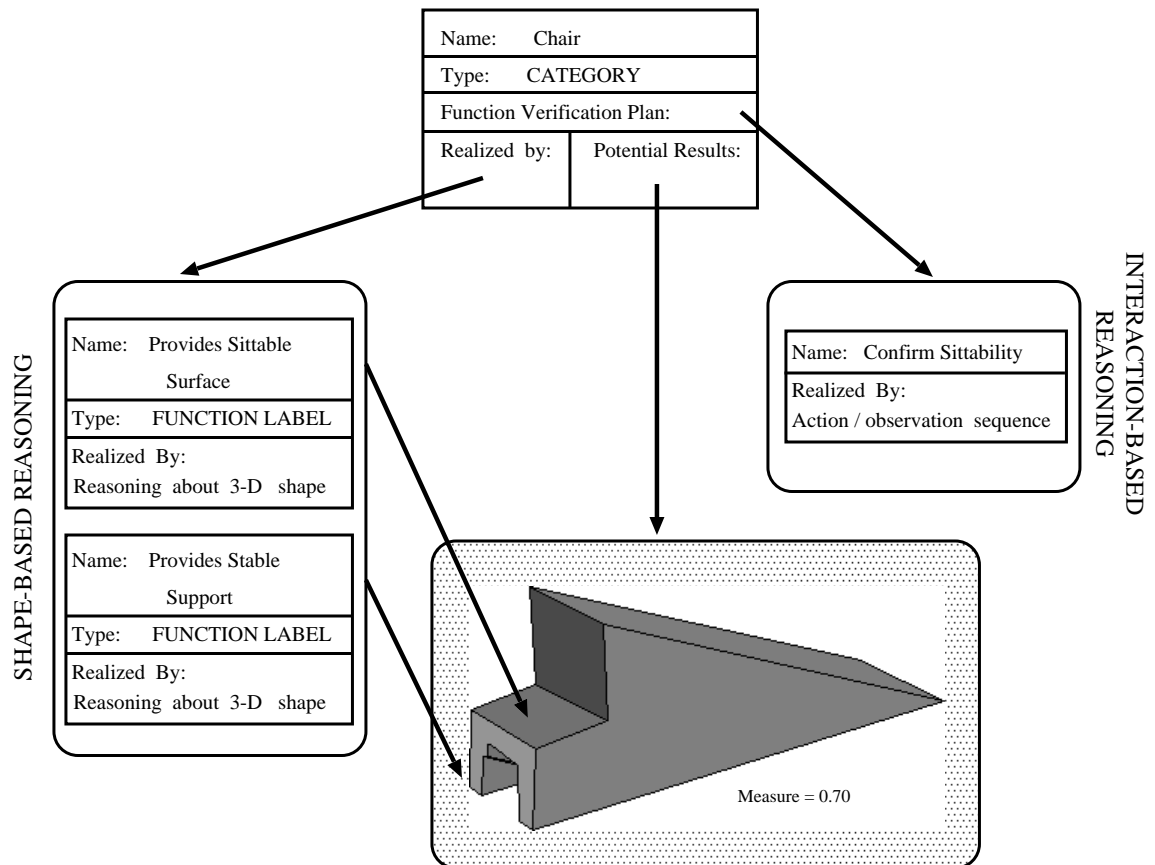


Figure 32. Functional properties for chair objects. Functional properties that involve only reasoning about abstract shape are shown on the left, as the *hypothesized functional plan*. Functional properties which reason about physical interaction are shown on the right, as the shape-suggested *function verification plan*.

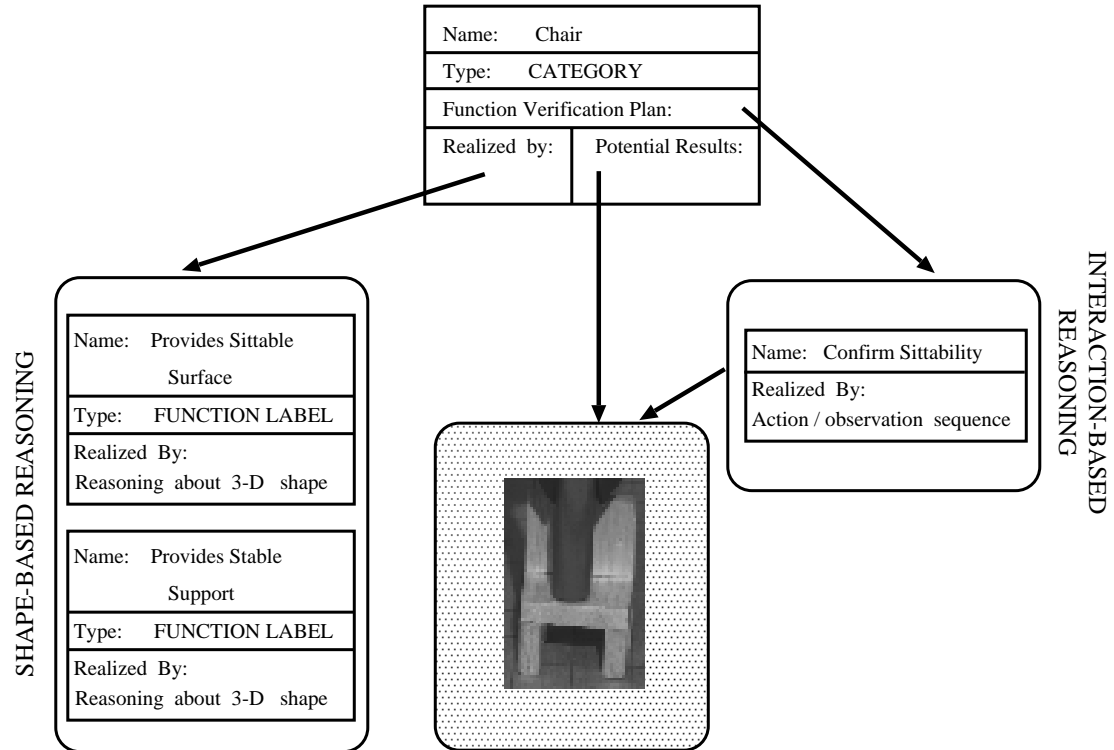


Figure 33. Verifying the functional plan for a chair. Verification of the functional plan involves instructing the robot arm to interact with the object in a manner consistent with its shape-suggested functionality.

To confirm these requirements for the object in the scene, it is again necessary to design the explicit interaction tests needed to confirm that the object can remain a member of the chair category. One interaction requirement was designed for this purpose (shown in Figure 33):

- *confirm sittability* - the object can support weight applied at the labeled sittable surface.

An example of a sequence of images acquired during confirmation of this requirement is shown in Figure 34.

6.3.1 Confirm sittability

To confirm sittability, the system must apply a weight to the seat, using the following calls:

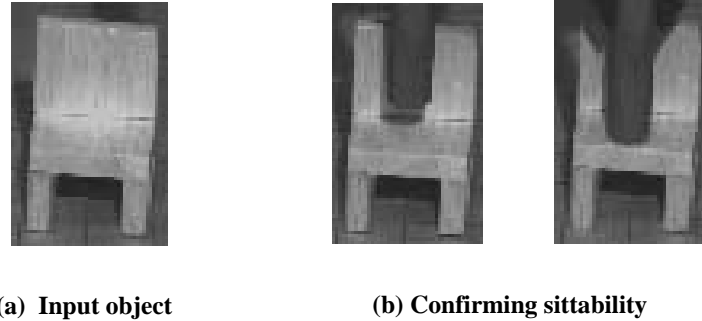


Figure 34. Interaction test for a chair object. Confirmation of the object as a member of the chair category of objects involves the interaction test confirm sittability.

- *apply force(GRASP, ...)* - Traverse a predetermined path to get a weight.
- *apply force(TRANSPORT, ...)* - Position weight relative to the destination point.
- *observe deformation(INTENSITY-STRUCTURE, ...)* - Compare images before and after weight positioning.
- *apply force(UNGRASP, ...)* - Release the weight.
- *observe deformation(INTENSITY-STRUCTURE, ...)* - Compare images before and after weight placement.
- *apply force(GRASP, ...)* - Re-grasp the weight.
- *observe deformation(POSITION, ...)* - Compare results of gripper location and separation distance before and after grasp.
- *observe deformation(INTENSITY-STRUCTURE, ...)* - Compare images before and after weight recovery.
- *apply force(TRANSPORT, ...)* - Return weight to its starting position.
- *apply force(UNGRASP, ...)* - Release the weight.
- *observe deformation(INTENSITY-STRUCTURE, ...)* - Compare images before and after weight removed from scene.

For the chair category, the approach path is based on the path necessary to transport the weight to the center of the seat. The Shape-based Reasoning Subsystem provides the information to approach the seat in a direction which is perpendicular to an edge which has been determined to be clear. The cross product of a candidate

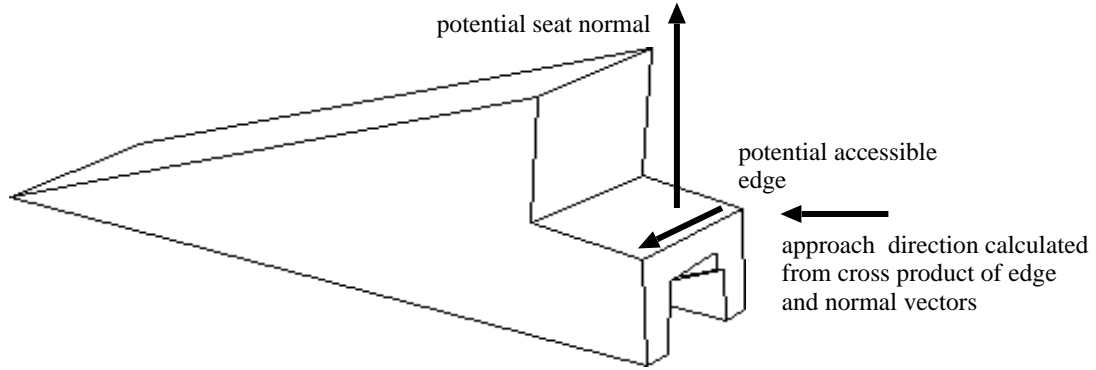


Figure 35. Path planning for chair objects. The approach path for these objects is determined by the location of the sittable surface and the edges which have been determined to be accessible for sitting.

approach edge on the seat and the normal vector directed from the sittable surface is used for path planning, as shown in Figure 35. After approaching along this direction, the robot arm places the weight in the center of the sittable surface, as estimated by the average of the vertices of this face.

6.4 Implementation Details

An outline for the code to control the robot arm was provided with the Microbot Alpha reference and programming guide manuals [66, 67]. In addition, a previous robotics course at USF provided an implementation of these operations in C. Using these two sources as a base, the software was redesigned and recoded to run and control both the Microbot Alpha robot arms and the Microbot TeachMover arms. The Microbot TeachMover arms were an earlier set of robot arms, used during preliminary development of the **GRUFF-I** system. Furthermore, the initial version of this software was written to run under System V-like versions of UNIX. The implementation described in this dissertation has been made POSIX compatible, in order to be portable to run under BSD-like versions of UNIX as well.

In addition, this set of operations and those written for performing image processing are stand-alone systems. In other words, the **GRUFF-I** system actually performs a system call to the executable files which implement these operations. This modular approach to implementing the system has allowed these operations to be used to solve other problems outside the scope of this dissertation. Furthermore, this approach has allowed for easier expansion and integration of new operations.

Finally, in an initial version of this system, more rigorous deformation analysis incorporating hyperquadric model fitting to the object in the 2-D intensity images was used, as reported in [59]. However, this method yielded poor response time (on the order of minutes), and thus the simpler features described in this chapter were utilized instead.

6.5 Summary

This chapter concludes the discussion of each of the subsystems upon which the **GRUFF-I** system is based. Figure 36 provides a summary of this information and provides additional details on how these subsystems communicate. As noted by the numbering, the first step in the process is to invoke the Model Building Subsystem to build a 3-D model and report information about the faces and vertices in the recovered model. The Shape-based Reasoning Subsystem is then invoked to use this information and apply concepts of physics and causation to the 3-D shape, using operations such as clearance and stability. The results of this analysis are reported in a text file. Finally, the Interaction-based Reasoning Subsystem reads this text file and is subsequently invoked multiple times, using operations to control the robot arm and the CCD camera and to perform image processing, in order to confirm the shape-suggested functionality of the 3-D model. The final results of this subsystem are also reported in a text file.

Such a design is similar in several respects to a blackboard architecture-based system, such as that used in the the road scene understanding system developed by Lapierre, *et al.* [60]. For example, in the **GRUFF-I** system, a single “supervisor” program invokes independent model building, shape-based and interaction-based “specialist” programs, each of which provides a partial interpretation of the current scene, using different sensors and data formats. The controlling program focuses the activities of the sensors according to the success or failure of the interaction. As Lapierre pointed out, there are a number of advantages to this modular approach, such as the simplicity of adding new sensors and specialists, the ease of program maintenance and the support of software re-usability, in that well-designed specialists can be used to solve other problems.

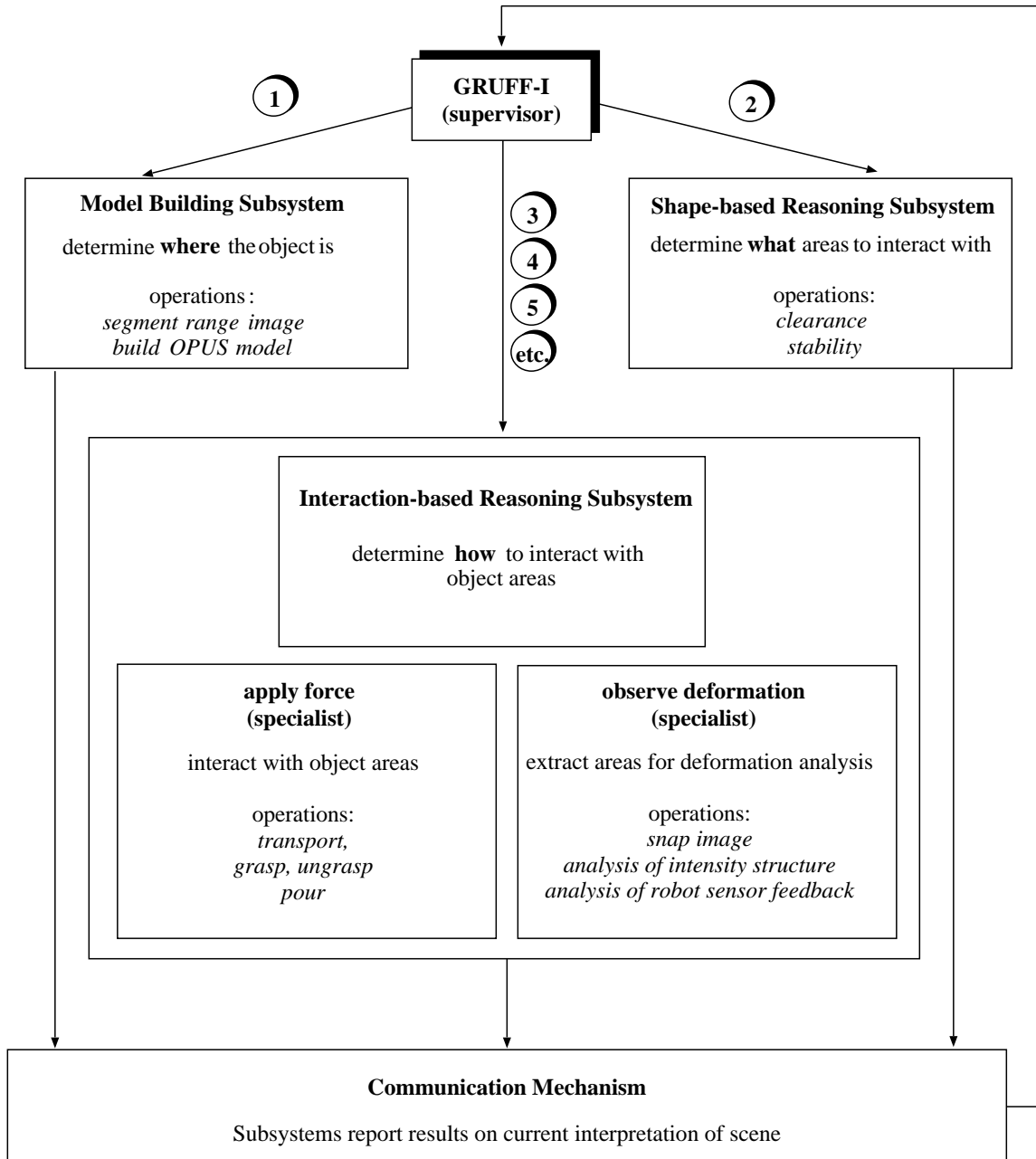


Figure 36. Communication in the **GRUFF-I** system. The subsystems communicate via shared files which contain information about how each subsystem interprets the scene.

CHAPTER 7

EXPERIMENTAL RESULTS

7.1 Evaluation Techniques

In order to evaluate the competence of the **GRUFF-I** system, three factors were considered significant. First, a set of test objects must be designed which meets the demands/constraints imposed by the experimental setup and data acquisition components. This provides a means to test the robustness of each of the individual subsystems, as well as the overall system. Second, what constitutes a repeatable “experiment” must be defined. Third, the methodology for analyzing the results of each experiment must be established. This chapter provides these details along with the results of testing the system with a set of physical objects.

7.1.1 Design of Test Objects

When designing the set of test objects, three characteristics were considered important, as follows:

1. **Category/Shape Variation** - The shape of the objects should be varied within a given category of objects. In addition, there should be objects from categories outside of the domain of objects known to the system.
2. **Placement/Orientation Variation** - The orientation of the object in the scene should be varied.
3. **Material Composition Variation** - The material composition of the objects should be varied to include different materials, such as paper, balsa wood, sponge and Styrofoam.

Each of these characteristics provides the opportunity to test different aspects of each of the subsystems. For example, shape/category variation provides a means to test the Model Building Subsystem for its performance on creating both simple and complex models. More complex models would be those where there is possibility of object self-occlusion, or those where shadow areas affect the heuristics used for surface fitting and/or joining. Shadow areas are an inherent problem when using a structured light scanner. Whenever some surface of the object obstructs the view of the light projector, the resulting range pixels are marked as invalid. Such invalid pixels contribute to the overall complexity of the acquired model, and demonstrate the ability of the system to handle “noise.” Shape/category variation can also be used to test the dimensional constraints imposed in the Shape-based Reasoning Subsystem, which had to be scaled to meet the workspace limitations. Finally, this variation also permits testing of the robustness of the path and grasp planning strategies in the Interaction-based Reasoning Subsystem.

Varying the placement and orientation of the object when it is positioned in the scene provides additional testing of the success of the Model Building Subsystem and determines if placement affects heuristics used for range segmentation or model building. It also allows for testing the Shape-based Reasoning Subsystem with objects showing varying levels of functionality for a given category. This is a result of the constraint that model recovery is performed from a single view, and if this view provides insufficient information about functional surfaces, the object will not pass shape-based reasoning. Finally, placement/orientation variation provides additional testing for the path and grasp planning strategies in the Interaction-based Reasoning Subsystem.

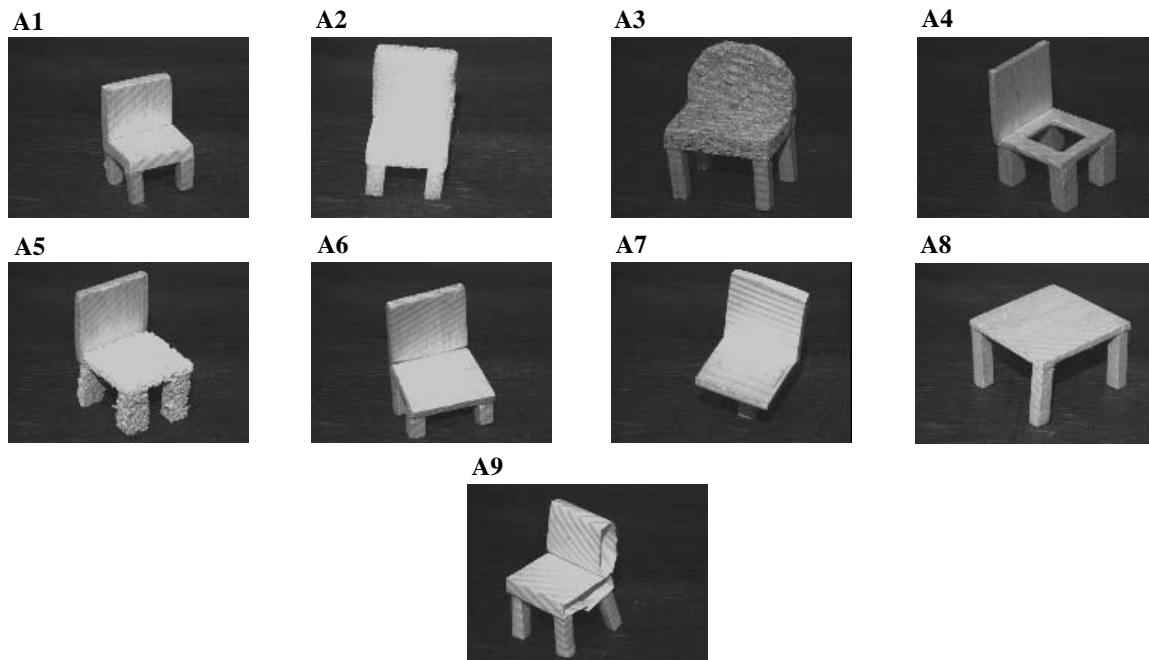
Variation of material composition allows the capability to test the effect of materials with different reflectivity properties on the range acquisition component of the Model Building Subsystem. This in turn affects the complexity of the recovered

model, which provides additional testing for the Shape-based Reasoning Subsystem. Finally, this variation provides the opportunity to test the dynamic knowledge primitives of the Interaction-based Reasoning Subsystem. Since the appearance of the different materials varies in the intensity image, varying the material composition of objects elucidates any limitations of the deformation analysis used in the primitive **observe deformation**. For the primitive **apply force**, since different materials tend to have markedly different structural properties, this variation allows testing of the sufficiency of applications of forces to the object during interaction.

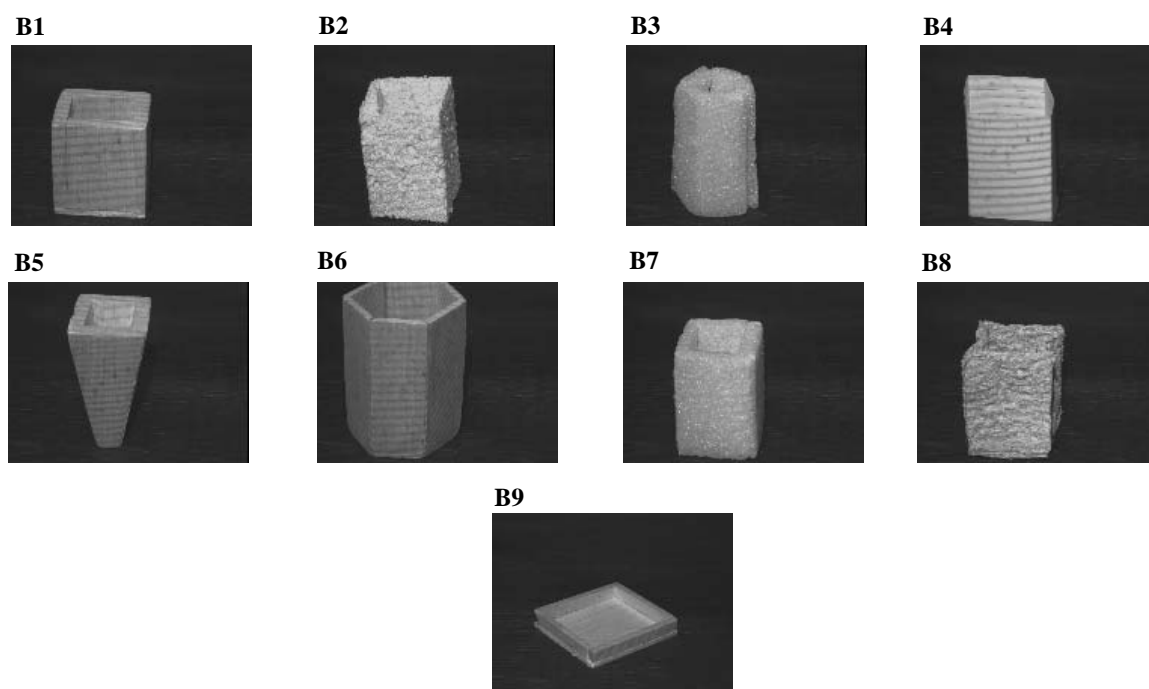
With these factors in mind, eighteen object models were created from a variety of materials (Styrofoam, balsa wood, paper and sponge) for experimentation. An intensity image of each of these objects is provided in Figure 37. In addition, a description of the general shape characteristics and material composition of each is provided in Table 1.

7.1.2 Recognition Experiments

In the **GRUFF-I** system, an experiment or set of experiments is initiated after the system calibration procedures have been completed. For the results presented in this dissertation, two sets of experiments were run. For furniture-like objects, each object was placed in front of the scene in five different orientations and a 0.09 kg weight composed of steel was used during the interaction. For dish-like objects, each object was placed in front of the scene in three different orientations. It is felt that this set of objects and orientations provide a broad range of test conditions to determine the feasibility of the **GRUFF-I** system approach. The total processing time per object is on the order of tens of minutes and is a combination of the CPU time to execute the various subsystems and the number of interaction sequences for the object category. Recall that for furniture-like objects, one interaction-based requirement must be met



(a) furniture-like objects



(b) dish-like objects

Figure 37. Functional and non-functional object models.

Table 1. Summary of Test Object Characteristics

Number	Category	Material Composition	Shape Description
A1	chair	balsa wood	small-sized
A2	chair	Styrofoam	medium-sized
A3	chair	sponge, balsa wood	large-sized; seat and back are sponge; legs are balsa wood
A4	chair	balsa wood	object has hole in seat
A5	chair	sponge, balsa wood	seat and legs are sponge; back is balsa wood
A6	chair	balsa wood	seat is slanted downward, toward the floor
A7	chair	balsa wood	object has one leg
A8	table	balsa wood	object is not a chair
A9	chair	paper	small-sized
B1	cup	balsa wood	square-shaped
B2	cup	sponge	object is non-rigid
B3	cup	Styrofoam	object pieces are placed next to each other, but not held together
B4	cup	paper	object is non-rigid
B5	cup	balsa wood	object sits on a small base
B6	cup	balsa wood	object has hole in one side
B7	cup	Styrofoam	object has no bottom
B8	cup	sponge	object is non-rigid
B9	plate	balsa wood	object is not a cup

(*confirm sittability*), while for dish-like objects, three interaction-based requirements (*confirm graspability*, *confirm containment* and *confirm stable containment*) must be satisfied.

The first set of experiments examined the effect of interaction with the furniture-like objects in Figure 37-(a), using all three subsystems (Model Building, Shape-based Reasoning and Interaction-based Reasoning). The second set of experiments examined the effect of interaction with the dish-like objects in Figure 37-(b), using the final two subsystems (Shape-based Reasoning and Interaction-based Reasoning). This latter set of experiments assumed a complete 3-D model as input for the shape description, defined in terms of faces and vertices in the robot arm-centered coordinate system. This assumption was made as a result of problems with the added complexity of surface descriptions of concavities within the 3-D shape in the Model Building Subsystem. In these cases, although OPUS models could be created, the topology of objects with concavities could not be handled properly by the post-processing routines used after model building. Systems which combine multiple views could form a close-to-complete model, as we are using here. However, combining views is a research topic in itself (see [21, 22, 64]) and is not in the scope of this dissertation.

7.1.3 Verification and Validation Methodology

Before initiating an experiment with a set of objects, the methodology for validating the results must be established. There are at least two levels at which this can occur. At the highest level, the user can enumerate *a priori* whether each object should or should not pass through each of the subsystems. The object can then be placed in the workspace, and the output of the system can be recorded. Alternatively, the user could enumerate the exact shape-based or interaction-based functional requirement the object should fail, if it should fail one. In this case, further

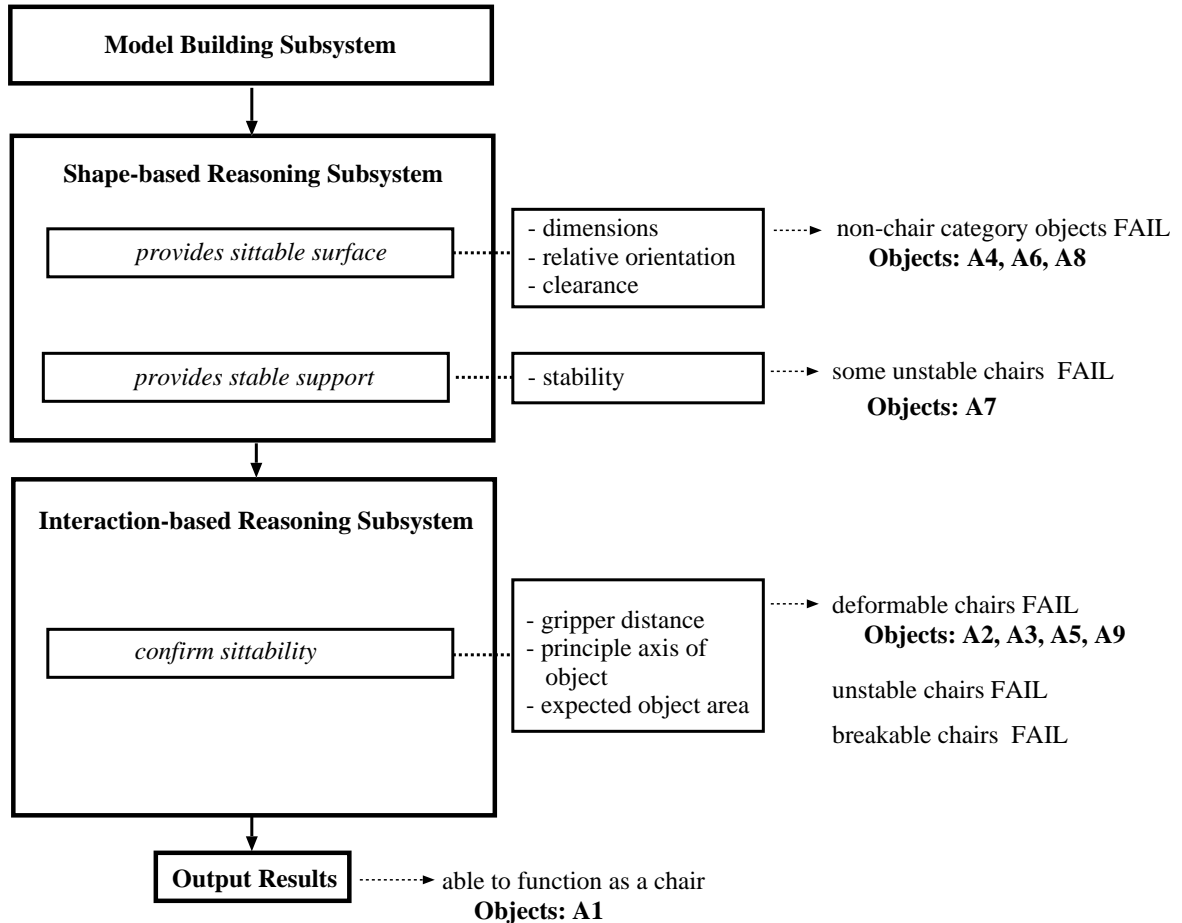


Figure 38. Projected failure points for chair objects.

details can be analyzed from the system to see if these more rigorous expectations are met. With these factors in mind, for each of the test objects shown in Figure 37, the various points in the processing where each of these objects was expected to fail were enumerated, as summarized graphically in Figures 38 and 39.

Inside the Model Building Subsystem, an object may fail for legitimate reasons, such as when it is unable to satisfy all of the requirements for a category of objects. For example, non-cup-like objects such as bowls and plates or objects without concavities are expected to fail the requirement *provides accessible glass containment* due to the dimensional constraints imposed for concavities (including volume, height and depth). For the furniture category, objects such as tables, or chairs with non-functional sittable surfaces (i.e. those which have a non-parallel orientation or contain

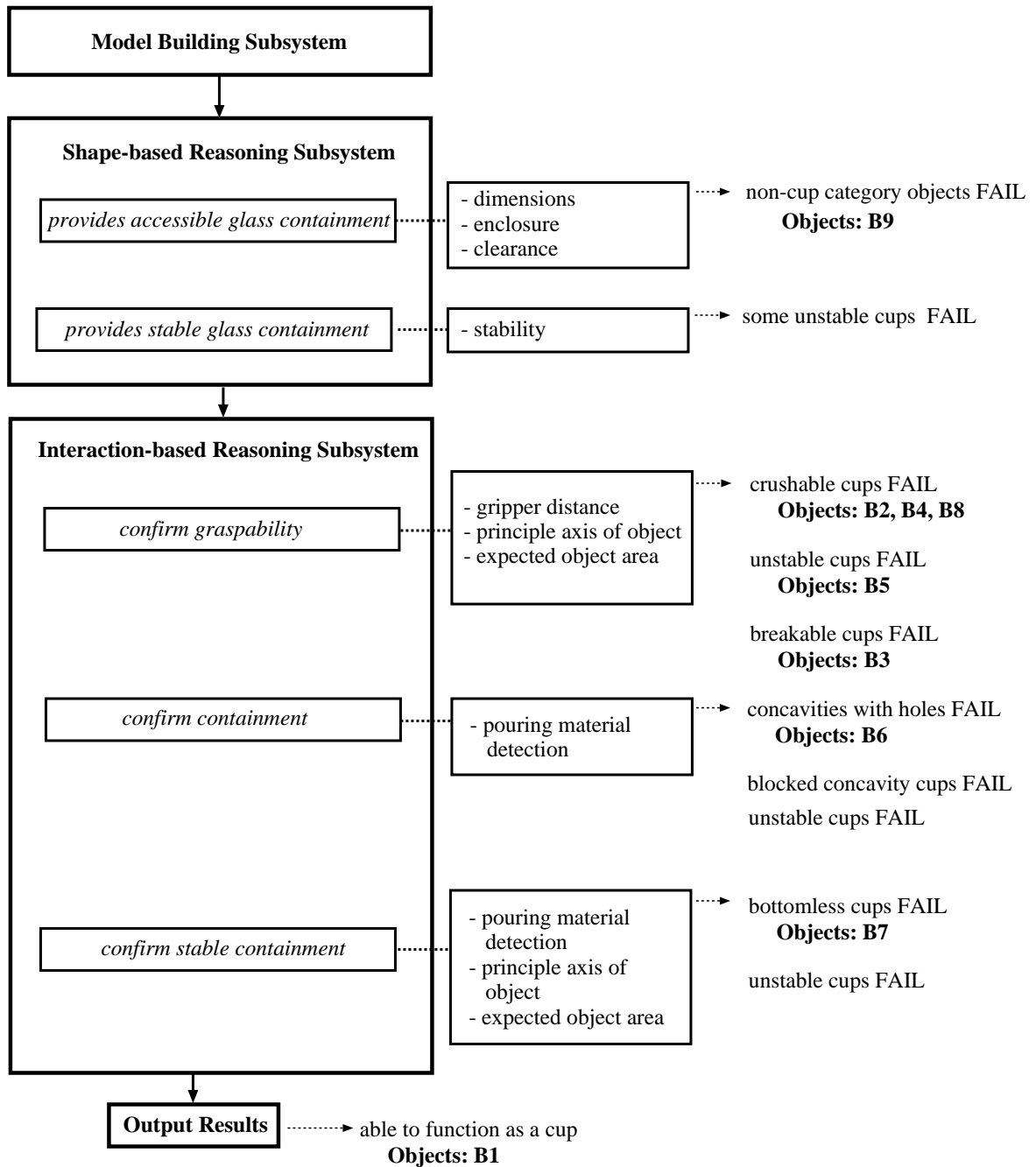


Figure 39. Projected failure points for cup objects.

holes in the potential seat) are expected to fail the shape-based requirement *provides sittable surface*.

Expectations of failures in the Interaction-based Reasoning Subsystem are largely based on the material composition of the object and corresponding assumptions about how the structure will hold up during the interaction. Cup-like objects which are crushable, breakable or unstable are expected to fail the *confirm graspability* requirement, since the gripper is expected to deform the object when grasping it. Objects with holes in the concavities, or blocked concavities, are expected to be unable to hold the containment material and therefore fail the requirement of *confirm containment*. In addition, somewhat unsteady cups should also fail this requirement, when the pouring material comes into contact with points on the object. Finally, bottomless cups are expected to fail the final requirement *confirm stable containment*, since the pouring material below the object may only be detected when the object is transported upwards. For furniture-like objects, those which are deformable when weight is applied to the seat are expected to fail the requirement *confirm sittability*.

7.2 Demonstration of System Competence

The various points in the system evaluation where each of the experimental objects actually did fail during the trials is shown in Tables 2 and 3. In these tables, the notation “pass / pass” or “pass / fail” indicates the results of the *shape-based / interaction-based* analysis. The designation “fail / –” indicates that shape-based reasoning has not been successful, and as such, no interaction plan is generated or executed.

Table 4 provides a summary of the results of recognition processing, in terms of where each of the physical objects unexpectedly failed during the trials. As noted previously, the Model Building Subsystem is disabled for cup-like objects (denoted

Table 2. Experimental Results (furniture-like objects)

Object	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5
A1	pass / pass	pass / pass	model error	no model	fail / -
A2	pass / pass	model error	fail / -	pass / pass	pass / pass
A3	fail / -	pass / fail	pass / fail	pass / fail	pass / fail
A4	no model	no model	no model	no model	no model
A5	pass / fail	pass / fail	pass / fail	pass / fail	pass / fail
A6	pass / pass	pass / fail	pass / fail	fail / -	fail / -
A7	fail / -	pass / pass	no model	model error	no model
A8	fail / -	fail / -	fail / -	no model	no model
A9	pass / fail	pass / fail	pass / fail	fail / -	pass / fail

Table 3. Experimental Results (dish-like objects)

Object	Trial 1	Trial 2	Trial 3
B1	pass / fail	pass / pass	pass / pass
B2	pass / fail	pass / fail	pass / fail
B3	pass / fail	pass / fail	pass / fail
B4	pass / fail	pass / fail	pass / fail
B5	pass / fail	pass / fail	pass / fail
B6	pass / pass	pass / pass	pass / pass
B7	pass / pass	pass / pass	pass / pass
B8	pass / fail	pass / fail	pass / fail
B9	fail / -	fail / -	fail / -

Table 4. Summary of Unpredicted Subsystem Failures

Category	Model Building Subsystem	Shape-based Reasoning Subsystem	Interaction-based Reasoning Subsystem
Chairs	13 / 45 (29%)	8 / 32 (25%)	3 / 18 (17%)
Cups	—	0 / 27 (0%)	7 / 27 (26%)

by the “—” in Table 4) due to difficulties in post-processing objects with concavities. For these orientations, a complete 3-D model defined in terms of the robot arm-centered coordinate system is used. Figures 40 and 41 show a subset of the example OPUS models acquired when testing furniture-like objects. In addition, Figure 42 provides a subset of the intensity images acquired during testing dish-like objects. The following sections summarize specific difficulties or discrepancies within each of these subsystems.

7.2.1 Model Building

For this subsystem, for the furniture category of objects, the system was unable to create a valid 3-D model for 13 / 45 orientations (29 %). In 10 / 45 orientations, an OPUS could not be created from the segmented range image. An example of these types of errors is shown in Figure 43. The range segmentation stage of the Model Building Subsystem tended to be sensitive to noise and the reflective properties of the surface on which the objects sit, which contributed to these problems. The final column of Figure 43 demonstrates how, in many cases, pieces of the objects may not be recovered during segmentation due to being joined to the background region.

In 3 / 45 orientations, an OPUS model could be created, but was determined to be an invalid boundary representation, due to the constraint that two faces may only intersect each other at a common edge in the model. Figure 44 demonstrates the three orientations in which such erroneous face intersections were encountered in the OPUS models.

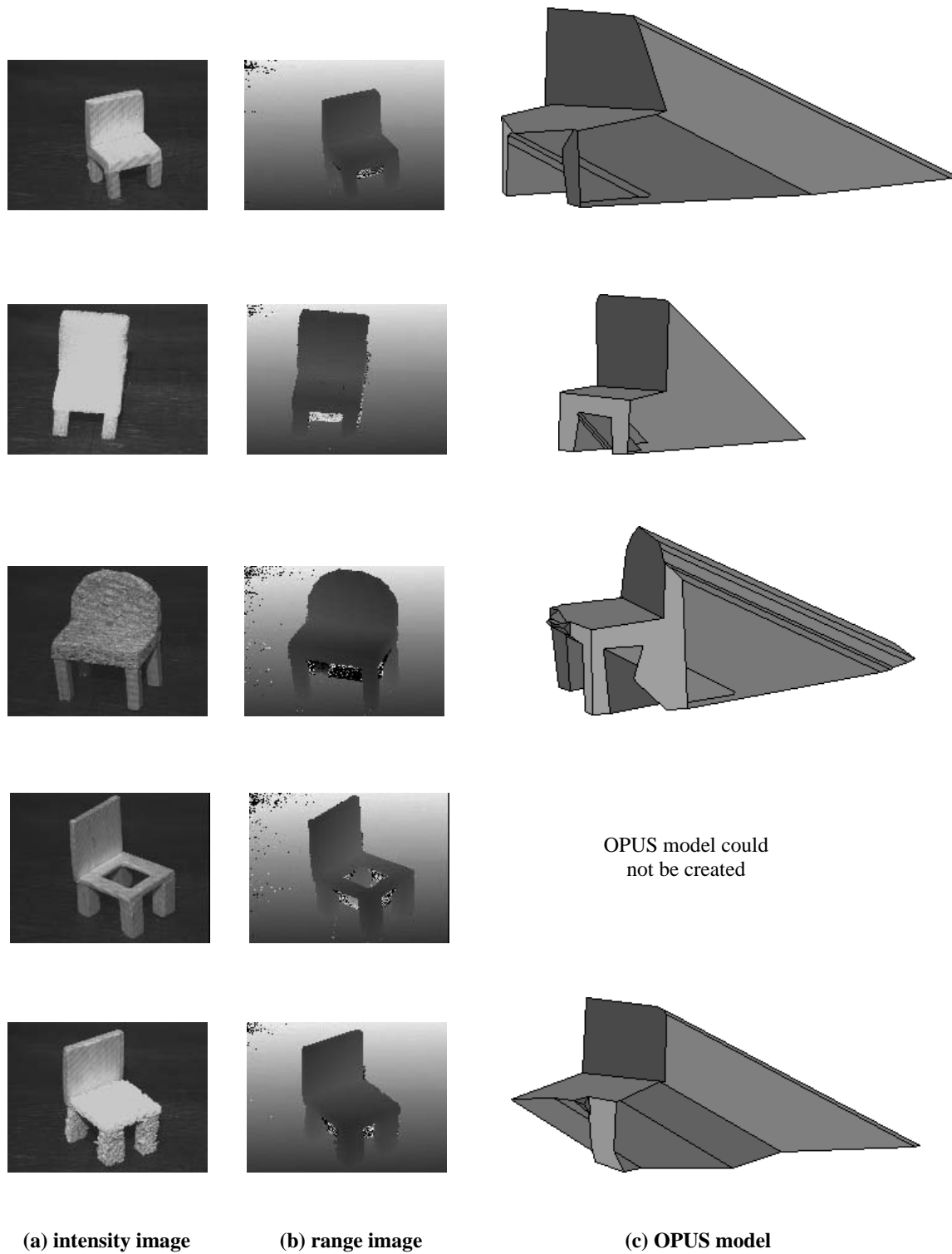


Figure 40. Representative OPUS models for chair objects A1-A5.

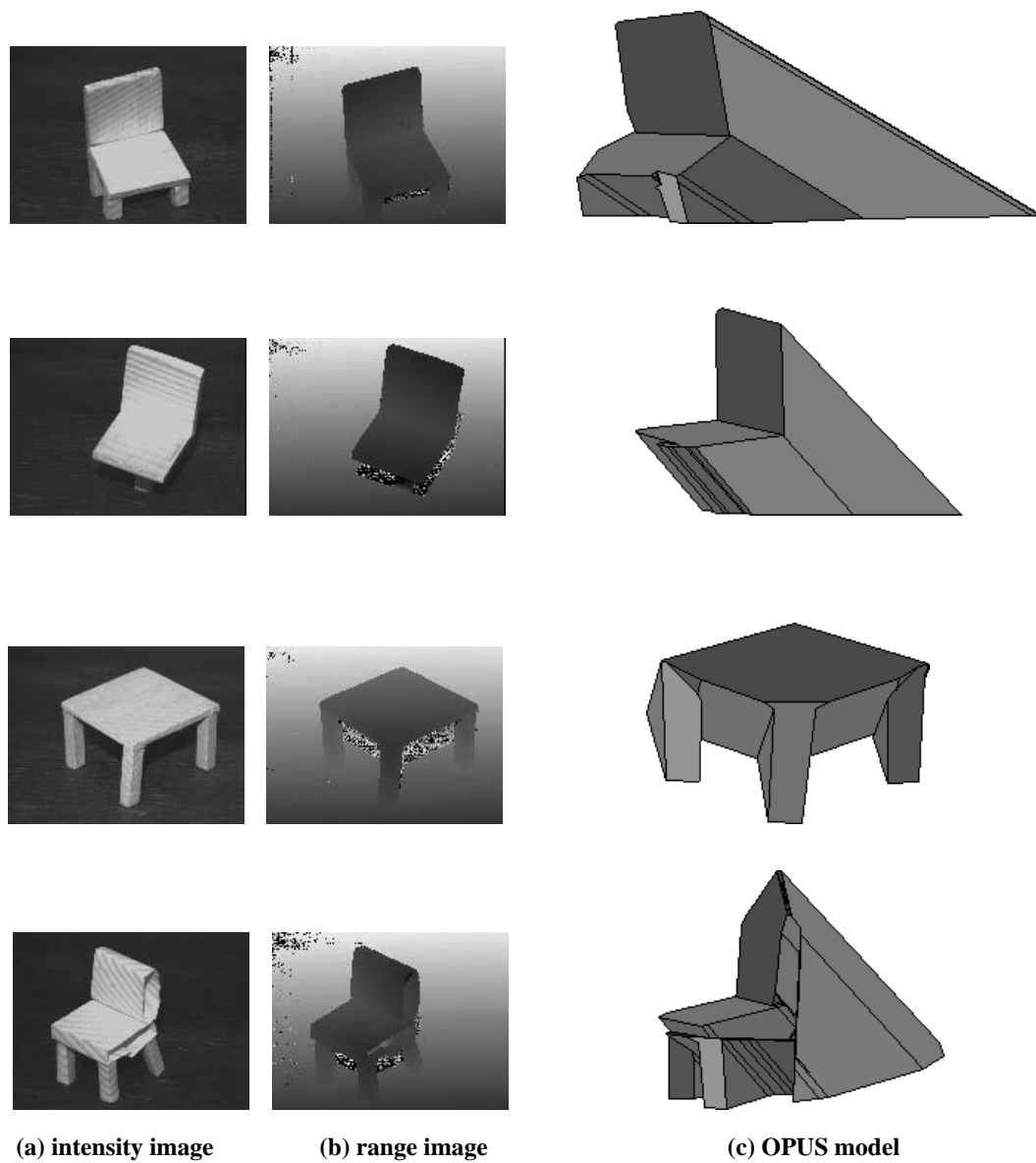


Figure 41. Representative OPUS models for chair objects A6-A9.

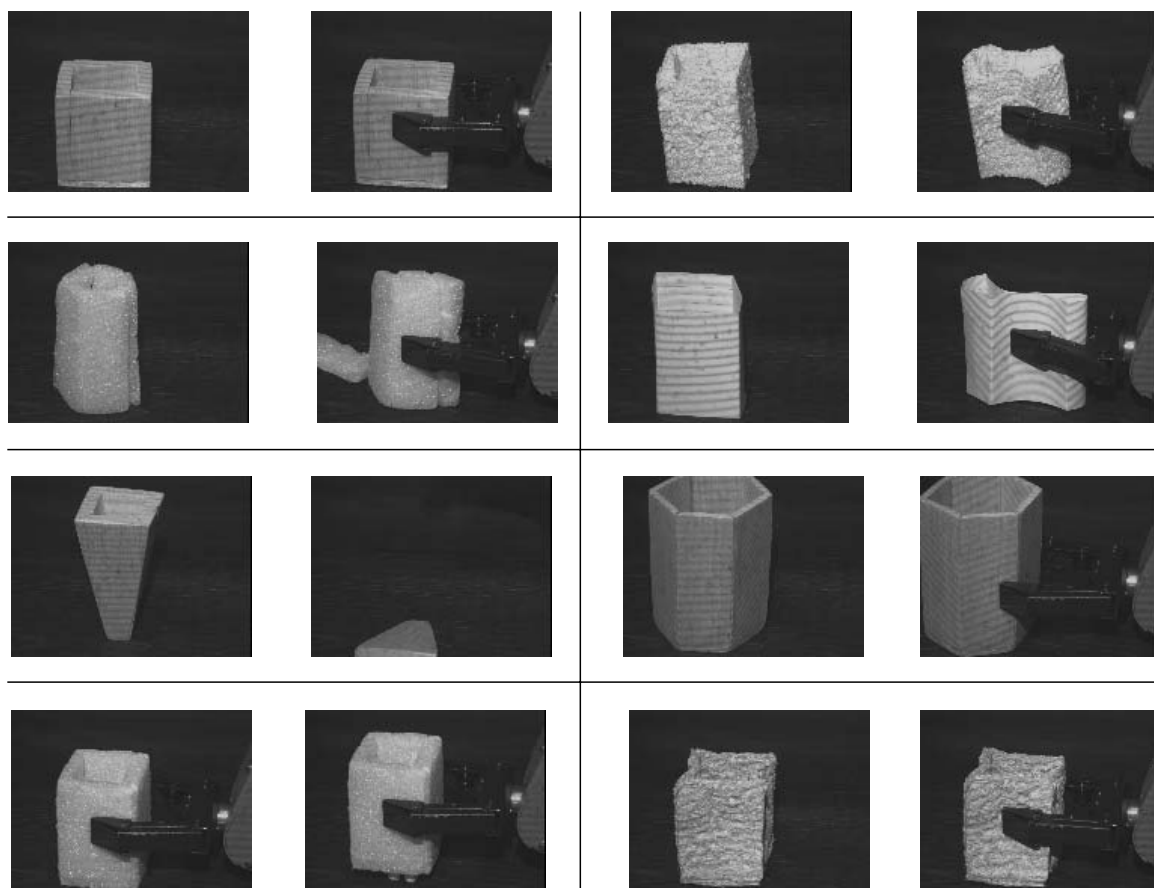


Figure 42. Representative image sequences for cup-like objects B1-B8.

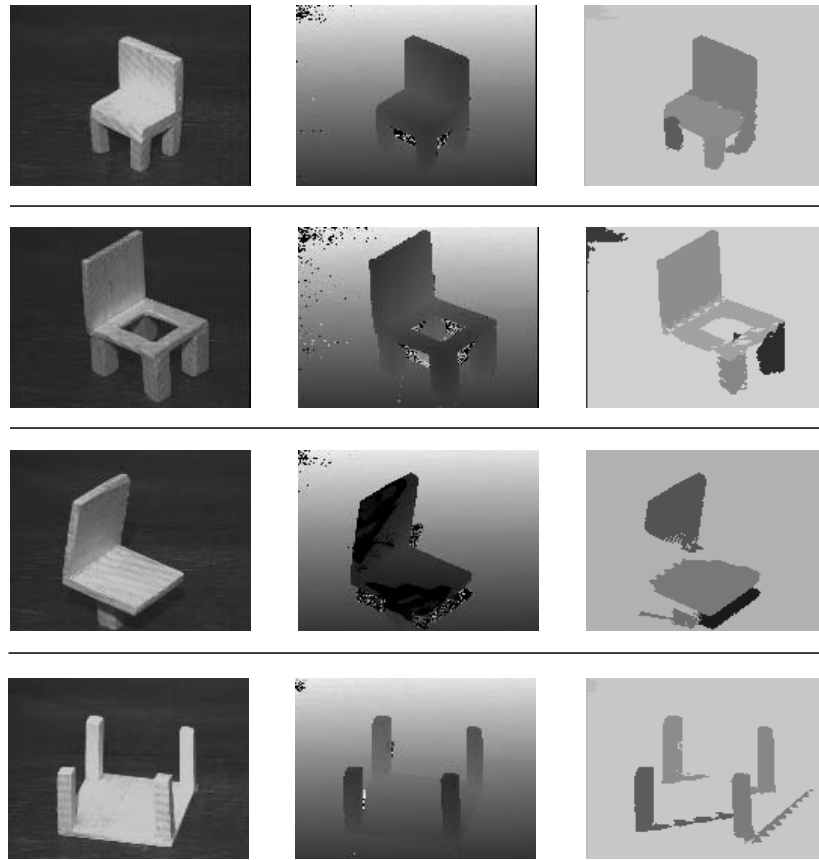


Figure 43. Object orientations in which no OPUS model could be created.

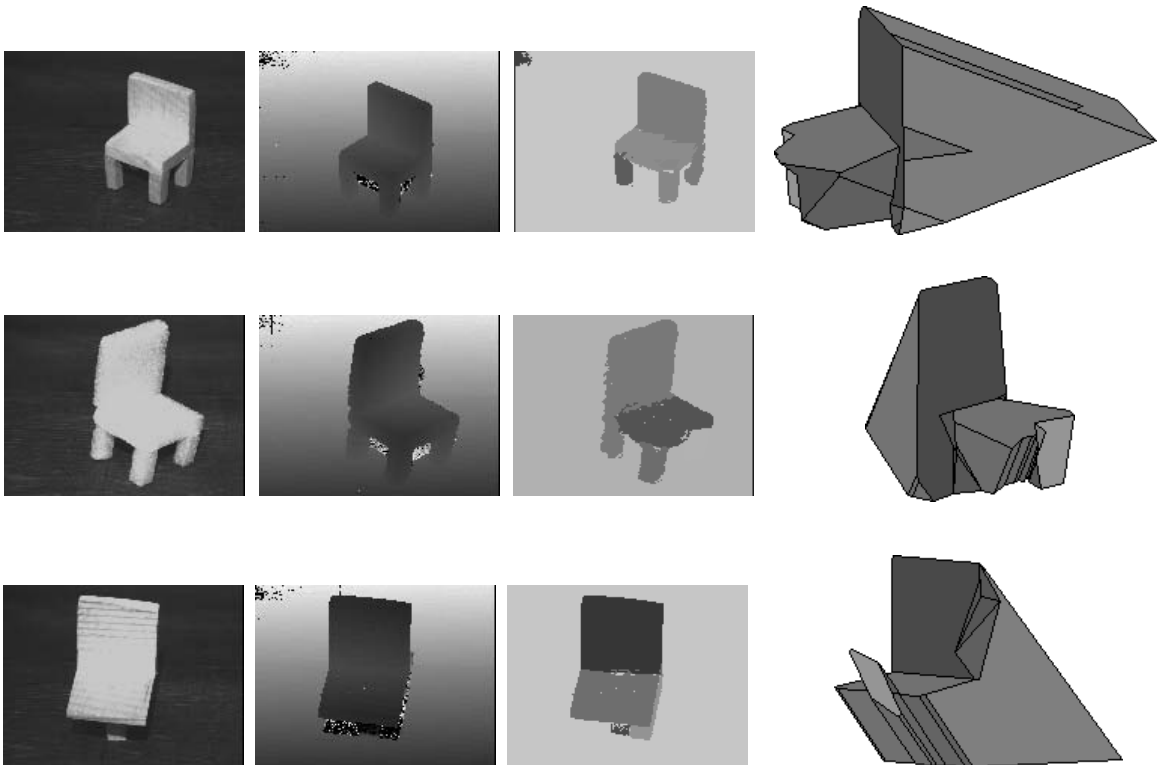


Figure 44. Object orientations in which resulting OPUS model was invalid. The boundary representations were invalid due to face intersections in the recovered model.

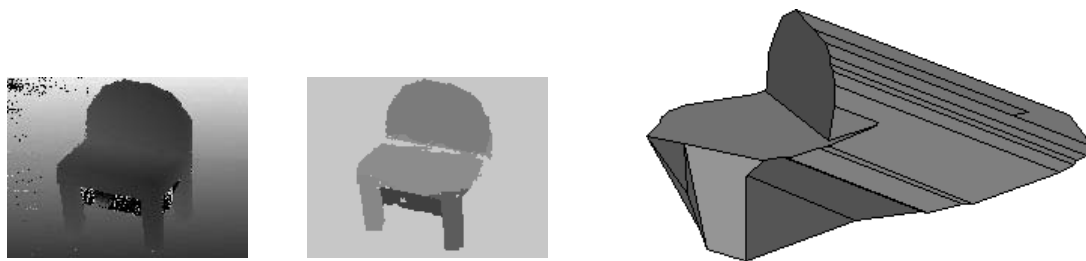


Figure 45. Object orientation which failed shape-based reasoning.

7.2.2 Shape-based Reasoning

Unexpected results for shape-based analysis occurred for 8 / 32 orientations (25 %) for furniture-like objects and 0 % of the time for dish-like objects. In 4 / 32 orientations for the furniture category, the object failed shape-based reasoning when it was expected to pass. These failures were attributable to the dimensional constraints imposed for the category chair. For example, for the object shown in Figure 45 (object A3), the dimensions of the sittable surface exceed the system thresholds for this surface. Recall that in such cases, when shape-based reasoning fails, no interaction plan is created or executed. This is one of the advantages of the **GRUFF-I** approach, in terms of the efficient use of system resources. There is no point to interact with an object which does not meet the minimal shape-based requirements to function as a potential member of a category of objects.

In 4 / 32 orientations for the furniture category, the object passed shape-based reasoning when it was expected to fail. This occurred for objects A6 (three orientations) and A7 (one orientation). For object A6, the problem is attributable to the fact that the sittable surface orientation was not significantly off from being parallel with the ground plane. For object A7, the physical model appeared unstable. However, the recovered 3-D model did not give this impression. This chair, which is composed of only one leg (see Figure 37), has a corresponding OPUS model (shown

in Figure 41) which gives an impression of stability, allowing it to pass successfully through the 3-D shape-based analysis.

7.2.3 Interaction-based Reasoning

For the furniture category of objects, five different objects (A1, A2, A3, A5 and A9) were expected to pass shape-based reasoning (twenty-five object orientations). From this set, eighteen orientations actually passed through the Model Building and Shape-based Reasoning Subsystems to make it to the Interaction-based Reasoning Subsystem. Using a 0.09 kg weight composed on steel, 3 / 18 (17 %) of the orientations which were predicted to fail actually passed interaction-based analysis for the requirement *confirm sittability*. These orientations occurred for object A2, indicating that the system considered a chair composed of thick Styrofoam to be as functional as one composed of balsa wood. In a training set of tests, 16 / 18 orientations passed through the final subsystem, using a weight of 0.01 kg. In these cases, objects A2, A3, A5 and A9 were expected to fail interaction-based reasoning, based on insufficient material composition (these are composed of Styrofoam, sponge and paper). However, they were able to support small amounts of weight without deformation.

For the cup category of objects, the objects failed to be processed as expected in the Interaction-based Reasoning Subsystem 26 % of the time. Similar to the results for furniture-like objects, in 6 / 27 orientations, the object passed interaction when it was expected to fail. This occurred for objects B6 and B7. For this category, the error could be attributed to the method used for the interaction-based requirement *confirm containment*. For these orientations, the containment material fell outside the workspace area and could therefore not be detected. In 1 / 27 orientations (object B1), the object failed interaction when it was expected to pass. This occurred as a result of a positioning error in the final location of the pouring material beaker before

release. In this case, the pouring substance was detected outside the object area, when none was expected.

7.3 Discussion of Results

It is difficult to compare this system quantitatively to other systems, since there is no ground truth set of objects from which a number of systems have been tested. However, we can comment on the results of the system in terms of sensitivity to sensor noise and in terms of the repeatability of the experiments. In this case, the major trouble spots occurred in the Model Building and Interaction-based Reasoning subsystems.

As mentioned in Chapter 4, it is possible to run the Model Building Subsystem with various sets of parameters. For the **GRUFF-I** system, one parameter set was selected for all test objects, with the primary objective of minimizing the average residual value in the constructed valid model. A smaller residual value was sought to ensure the models would more closely match the actual range data. However, due to the number of model building errors observed, it may be appropriate to consider the effect of using scale space processing instead of a single set of parameters [42, 61]. This would mean that different sets of parameters would be applied to the range image until a valid model was formed, leading to a larger set of object orientations which could ultimately be tested with the final two subsystems.

In addition, Hoover outlines a number of strategies for handling shadow areas, including allowing pixels on the border of shadow regions to bleed across foreground and background boundaries [42]. Experimenting with these strategies for both categories of objects also has the potential to increase the success of the Model Building Subsystem. However, the cost of incorporating such bleed-through strategies would be a potentially decreased accuracy of the constructed model. For example, such

heuristics would allow real object faces to extend where none exist, which could have detrimental results in the Interaction-based Reasoning Subsystem, when path planning for specific points on the object.

Upon reaching the Interaction-based Reasoning Subsystem, the accumulated error of all the systems became a factor to the success or failure of the robot arm's path execution, since the scale of the objects had to be reduced to meet the workspace constraints. Recall that sources of error which contribute to the overall error include:

- Calibration Procedure - approximations are used for the parameters for the relationship between the coordinate systems for the projector, the CCD camera, the calibration cube, the workspace and the robot arm.
- Model Building Subsystem - various heuristics are used in both the segmentation and surface fitting stages.
- Interaction-based Reasoning Subsystem - the robot control routines use an approximation for the kinematics of the arm and are also affected by the mechanical problems associated with the arm, such as motor slippage.

Since the **GRUFF-I** system is an endpoint open loop system (vision is not used in path planning), this accumulated error can cause the gripper to be positioned incorrectly for a specific interaction test. Most often, the positioning discrepancy did not affect the final results. However, for smaller objects, the effect would obviously be more noticeable. For example, during interaction with object A1, which is a valid small-sized chair composed of balsa wood, the object was moved slightly by the gripper, when no movement should have occurred. This problem can be corrected in the future by calibrating with the larger calibration cube available with the structured light scanner, which will double the workspace area and allow the objects to be sized to more realistic dimensions. With this extended setup, it may also be useful to consider different sets of model building parameters for furniture-like and dish-like objects, since the relative scale between the two would also be more realistic.

Finally, the errors within the Interaction-based Reasoning Subsystem can be attributed to the limited amount of sensor feedback when testing the requirements

confirm sittability and *confirm containment*. For example, the experiments with furniture indicated the dependence of the *confirm sittability* requirement on a reasonable amount of applied weight. A more rigorous test for *confirm sittability* could test objects for degrees of functionality, using a range of weights. Alternatively, the incorporation of a force sensor at the end of the weight could provide feedback about the object's compliance. Along these lines, to handle the cases for the requirement *confirm containment*, when the pouring material was not detected within the camera's view, a weight sensor could be added to provide feedback on the weight of the object before and after testing this requirement. In either case, the added observational powers of the system would be at the cost of increased circuitry, hardware and processing time to acquire and record the measurements.

CHAPTER 8

CONCLUSION

In a similar vein, trying to understand perception by studying only neurons is like trying to understand bird flight by studying only feathers: It just cannot be done. In order to understand bird flight, we have to understand aerodynamics; only then do the structure of feathers and the different shapes of birds' wings make sense.

David Marr (1982) [63]

8.1 Summary of Dissertation

Previous versions of the **GRUFF** system could only indicate the *potential* sufficiency of a given 3-D shape for the functional requirements of some category. The work described in this dissertation is differentiated from this earlier work by the incorporation of actual visual and robotic components, operating on-line, to perform interaction with physical objects. The **GRUFF-I** system can distinguish between functional and non-functional objects, based either on shape alone, or on the results of interactions which are suggested by successful shape analysis. This system therefore operates under what might be termed an *expectancy* paradigm. An initial hypothesis is formed for recognition using bottom-up visual and shaped-based reasoning alone, expecting that the material properties of an object are sufficient. The results of this preliminary analysis (yielding locations of functionally significant areas on the object) provides the guidance to instantiate further (more expensive) top-down exploratory modules to confirm the shape-suggested functionality.

8.2 Contributions

The original contribution of this work is that it is the first *non-part-based approach* to use function-based reasoning to symbolically label *and* plan for interaction with objects. The existing research in generic object recognition includes a variety of input formats (including 2-D and 3-D data) and constraints on input orientations. Alternatively, the **GRUFF-I** system is implemented to evaluate a “static” *uninterpreted* rigid 3-D shape obtained from real image data. A set of physical artifacts was created and used to evaluate the prototype system, by placing each object in the workspace in a number of orientations. Additional contributions of this work include:

Metrically accurate representations of the world can be built and used for higher level reasoning. Valid OPUS models were obtained approximately 70 % of the time for furniture-like objects. Based on a set of transformation equations determined during calibration, these models were ultimately defined in terms of the robot arm-centered coordinate system. Path planning with a Microbot robot arm using 3-D points in this coordinate system resulted in an overall accuracy of approximately 0.01 m.

Shape-based reasoning prior to interaction-based reasoning provides an efficient methodology for object recognition, in terms of the judicious use of system resources. In the **GRUFF-I** system, shape-based functional object recognition provides the means to visually explore (pre-process) the object as much as possible to develop a list of intelligent locations for subsequent higher level top-down processing. If the object does not have the minimal shape-based characteristics to function as a member of a class of objects, no interaction plan is created. In addition, objects in non-functional orientations will not yield interaction plans.

Interaction-based reasoning can be used to confirm the functionality of a categorized object without explicitly determining the object’s material composition. It may be entirely possible to code algorithms to distinguish between the image properties of

various materials such as wood, sponge and paper, using reflective properties, texture, etc. However, such systems can ultimately be fooled if the object is covered with an external surface having favorable “rigid” properties. In the **GRUFF-I** system, shape-based analysis operates under the assumption that the material properties of the object are sufficient. Interaction-based analysis confirms or disproves this assumption by interacting with the object in a task consistent its shape-predicted functionality. This determines if, regardless of composition, the object will retain its functionality under use.

8.3 Implications in the Field of Computer Vision

The goal of this work was not to solve the whole problem of machine perception, nor to develop a perfect system for a special pre-defined visual mini-world. Rather, the objective was to develop a set algorithms for three subsystems, integrate the results, and demonstrate the competence of the resulting prototype system. Although there were a number of constraints on the workspace, researchers can make use of the results of this work in richer visual environments in the following ways:

In terms of the *sufficiency of the representation/organization* within the **GRUFF-I** system, the required knowledge of the world is organized hierarchically by a series of shape-based and interaction-based requirements for category membership. Research in human perception by Rosch, *et al.* supports the use of such hierarchical category classes [77].

In terms of *breadth*, the system is extendible to include both multiple domains of objects and multiple sensors. In addition, the system demonstrates that a small knowledge base of shape-based and interaction-based knowledge primitives is sufficient to test and confirm a variety of functional requirements. The **GRUFF-I** system

is also *generative*, in the sense of [14], in that it is capable of recognizing artifacts which have never been encountered before.

In terms of *practicality*, the **GRUFF-I** system approach can be differentiated from the work of Stansfield, who used a six-sided spatial polyhedral for designing interaction plans [79]. The determination of the locations of *functionally significant areas* by the **GRUFF-I** system does not limit the approach or grasp strategies. Furthermore, analyzing the entire shape for functionally significant areas is more useful than basing recognition on matching to prototypical part descriptions or color-based blob characteristics, in the sense of Rivlin, *et al.* [74] and Connell [24].

In terms of *efficiency*, several researchers, such as Bogoni, *et al.* [10] and Krotkov [57] have focused on the recognition of the material properties of objects, without considering shape. However, as mentioned previously, the **GRUFF-I** system is predicated on the idea that due to the volume of data an autonomous agent may encounter in any real-world scene, judicious use of resources is important. Shape-based reasoning prior to manipulation provides the ability to suggest possible areas of functional significance, where robotic manipulation can then be directed. Since robotic control can be expensive in terms of positioning, re-calibration and time, pre-recognition of the shape and design of an interaction plan to confirm those areas with functional significance provides an efficient methodology. We therefore expect to visually explore the object's 3-D model as much as possible, to develop a *plan of minimal work* for confirming object functionality. The cost of such shape-based analysis is directly proportional to the complexity of the model. This analysis is therefore not computationally cheap in an absolute sense, but appears cheaper relative to the costs involved with interacting independent of object shape.

8.4 Future Work

In addition to the changes suggested in the discussion of results, such as adding force and weight sensors to provide additional feedback when testing requirements such as *provides sittability* and *confirm containment*, there are a number of future directions for this work. One possibility would be the incorporation of building and analyzing 3-D models during each stage of the interaction and the design of a corresponding set of 3-D deformation analysis routines. This suggests the possibility of integrating and weighing different sources of information from different subsystems and the possibility of exploiting parallel programming to increase efficiency.

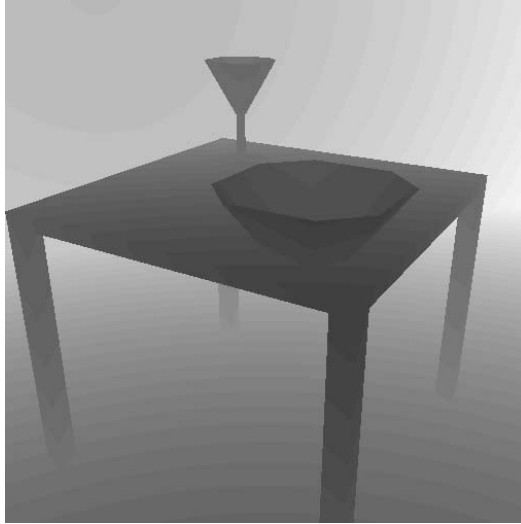
An additional robot arm would allow interaction with articulated objects. In addition, the use of an alternative 3-D data acquisition system could overcome the object material/appearance restrictions of the structured light scanner components. It may also be possible to design heuristics guided by information from registered range and intensity images for object detection and deformation analysis. In this case, during region growing, we need only consider the areas of the intensity image corresponding to the labeled 3-D model.

A more extensive project would involve integrating multiple views of an object in the scene to explore the shape as much as possible, prior to any interaction. This would allow the system to postulate the view or views which contain the “maximal function information” for model building. This suggests the need for the development of another interaction-based PKP which could be used to re-orient the object to acquire these additional views.

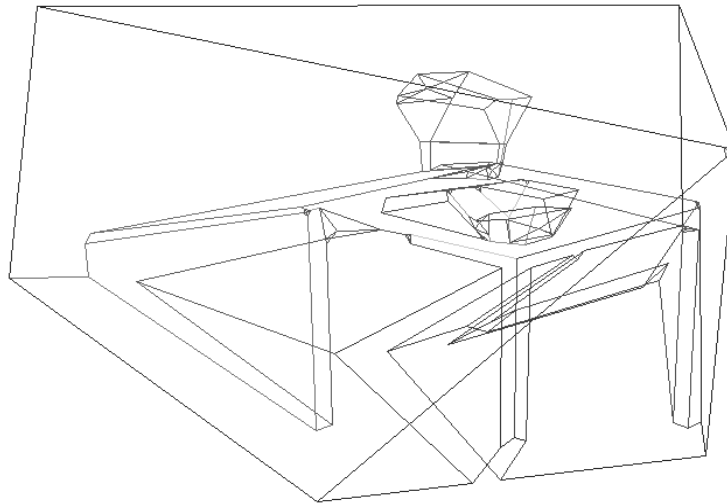
Finally, in the current implementation, isolated objects are assumed. However, more interesting applications are possible with scenes of objects. Toward this end, Hoover has designed a system for acquiring what he calls a *space envelope representation* of the world [42]. An example of this representation is shown in Figure

46. This offers an exciting opportunity to consider the implications of the “super-functionality,” or context, of a collection of objects, and suggests the possibility of performing object segmentation based on functional properties. Such “functionality in the large” applications also offer the opportunity to explore higher level processing mechanisms for scene analysis. For example, from the functional processing of a local neighborhood of objects, the system may determine the location is a “kitchen.” This information could then be used to further index into, or more appropriately rank, subsequent processing mechanisms.

The development of representations which will support generic object recognition is clearly of fundamental importance if the goal of autonomous real-world systems is to be achieved. Just as clearly, this area of research is still in its infancy. Object representations which are eventually developed to support this goal are likely to incorporate elements of all the types of models and sensor data described here, as well as information from additional modalities.



(a) Range image



(b) 3-D “space envelope”

Figure 46. Functionality in the ‘large.’

LIST OF REFERENCES

- [1] F. Ade, M. Rutishauser, and M. Trobina. Grasping unknown objects. In H. Bunke, editor, *Proceedings of Dagstuhl Seminar: Environment Modeling and Motion Planning for Autonomous Robots*, pages 445–459. World Scientific, 1995.
- [2] F. Ade, M. Rutishauser, M. Trobina, and A. Yla-Jaaski. A 3-D vision system for deriving gripping information for a robot. In *Proceedings of 8th Scandinavian Conference on Image Analysis*, pages 329–336, 1993.
- [3] P.K. Allen, A.T. Miller, P.Y. Oh, and B.S. Leibowitz. Using tactile and visual sensing with a robotic hand. In *Proceedings of IEEE International Conference on Robotics and Automation*, 1997 (to appear).
- [4] F. Arman and J.K. Aggarwal. Model-based object recognition in dense-range images – a review. *ACM Computing Surveys*, 25(1):5–43, March 1993.
- [5] S. Asaad, M. Bishay, D.M. Wilkes, and K. Kawamura. A low-cost, DSP-based, intelligent vision system for robotic applications. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1651–1661, 1996.
- [6] A. Bendiksen and G. Hager. A vision-based grasping system for unfamiliar planar objects. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 2844–2849, 1994.
- [7] P.J. Besl and R.C. Jain. Three-dimensional object recognition. *Computing Surveys*, 17(1):75–145, 1985.
- [8] M. Bishay, M. Cambron, K. Negishi, R.A. Peters II, and K. Kawamura. Visual servoing in ISAC - a decentralized robotic system for the disabled. In *Proceedings of IEEE Computer Vision Symposium*, pages 335–340, Coral Gables, Florida, 1995.
- [9] L. Bogoni. *Identification of Functional Features Through Observations and Interactions*. PhD thesis, University of Pennsylvania, 1995.
- [10] L. Bogoni and R. Bajcsy. An active approach to characterization and recognition of functionality and functional properties. In *AAAI Workshop: Reasoning About Function*, pages 9–16, 1993.
- [11] L. Bogoni and R. Bajcsy. Interactive recognition and representation of functionality. *Computer Vision and Image Understanding*, 62(2):194–214, September 1995.

- [12] P.P. Bonissone and K.S. Decker. Selecting uncertainty calculi and granularity: An experiment in trading-off precision and complexity. In L. Kanal and J. Lemmer, editors, *Uncertainty in Artificial Intelligence*, pages 217–247. North-Holland Publishing Company, 1985.
- [13] M. Brady, P.E. Agre, D.J. Braunegg, and J.H. Connell. The mechanics mate. In T. O’Shea, editor, *Advances in Artificial Intelligence: European Conference of Artificial Intelligence*, pages 79–94. Elsevier Science Publishers, 1985.
- [14] M. Brand. An eye for design: Why, where and how to look for causal structure in visual scenes. In *Proceedings of SPIE Workshop on Intelligent Robots and Computer Vision XI: Algorithms, Techniques and Active Vision*, pages 180–188, 1992.
- [15] M. Brand. Vision systems that see in terms of function. In *AAAI-93 Workshop on Reasoning about Function*, pages 17–22, Washington, D.C., July 1993.
- [16] M. Brand. Physics-based visual understanding. *Computer Vision and Image Understanding*, 65(2):192–205, February 1997.
- [17] R.A. Brooks. Symbolic reasoning among 3-D models and 2-D images. *Artificial Intelligence*, 17:285–348, 1981.
- [18] R.A. Brooks. *Model-Based Computer Vision*, volume 85. UMI Research Press, Ann Arbor, Michigan, 1984.
- [19] R.A. Brooks. Model-based three-dimensional interpretations of two-dimensional images. In M.A. Fischler and O. Firschein, editors, *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*, pages 360–370. Morgan Kaufmann Publishers, Inc., 1987.
- [20] M. Campos, R. Bajcsy, and V. Kumar. Exploratory procedures for material properties: The temperature perception. In *Proceedings of the International Conference on Advanced Robotics*, pages 205–210, June 1991.
- [21] Y. Chen and G. Medioni. Surface descriptions of complex objects from multiple range images. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 153–158, 1994.
- [22] Z. Chen and S.Y. Ho. Incremental model building of polyhedral objects using structured light. *Pattern Recognition*, 26(1):33–46, 1993.
- [23] R.T. Chin and C.R. Dyer. Model-based recognition in robot vision. *ACM Computing Surveys*, 18(1):67–108, 1986.
- [24] J.H. Connell. Get me that screwdriver! Developing a sensory-action vocabulary for fetch-and-carry tasks. *IBM CyberJournal Research Report RC 19473*, April 1994.

- [25] J.H. Connell and M. Brady. Generating and generalizing models of visual objects. *Artificial Intelligence*, 31:159–183, 1987.
- [26] P.R. Cooper, L.A. Birnbaum, and M.E. Brand. Causal scene understanding. *Computer Vision and Image Understanding*, 62(2):215–231, September 1995.
- [27] E. Davis. Shape and function of solid objects: Some examples. Technical Report 137, New York University, October 1984.
- [28] L. Dey, P.P. Das, and S. Chaudhury. Recognition and learning of unknown objects in a hierarchical knowledge-base. In *AAAI Fall Symposium Series - Technical Report FS-93-04*, pages 15–19, 1993.
- [29] M. DiManzo, E. Trucco, F. Giunchiglia, and F. Ricci. FUR: Understanding FUnctional Reasoning. *International Journal of Intelligent Systems*, 4:431–457, 1989.
- [30] Z. Duric, J. Fayman, and E. Rivlin. Recognizing functionality. In *Proceedings of International Symposium on Computer Vision*, pages 247–252, 1995.
- [31] Z. Duric, J.A. Fayman, and E. Rivlin. Function from motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(6):579–591, June 1996.
- [32] A.K. Goel. Integrating case-based and model-based reasoning. *AI Magazine*, pages 50–54, 1992.
- [33] K. Green. *Generic Recognition of Articulated Objects Through Reasoning About Potential Function*. PhD thesis, University of South Florida, 1996.
- [34] K. Green, D. Eggert, L. Stark, and K. Bowyer. Generic recognition of articulated objects through reasoning about potential function. *Computer Vision and Image Understanding*, 62(2):177–193, September 1995.
- [35] K. Green, D. Eggert, L. Stark, and K.W. Bowyer. Generic recognition of articulated objects by reasoning about functionality. In *AAAI Workshop on Representing and Reasoning about Device Function*, pages 56–64, August 1994.
- [36] A. Gupta, S.R. Gorti, and D. Sriram. Representing design knowledge: Function, form, and behavior. *AAAI*, pages 37–41, 1993.
- [37] G.D. Hager. Real-time feature tracking and projective invariance as a basis for hand-eye coordination. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 533–539, 1994.
- [38] G.D. Hager, W. Chang, and A.S. Morse. Robot feedback control based on stereo vision: Towards calibration-free hand-eye coordination. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 2850–2856, 1994.
- [39] S. Ho. *Representing and Using Functional Definitions for Visual Recognition*. PhD thesis, University of Wisconsin, 1987.

- [40] J. Hodges. Naive mechanics—a computational model of device use and function in design improvisation. *IEEE Expert*, pages 14–27, February 1992.
- [41] J. Hodges. Functional and physical object characteristics and object recognition in improvisation. *Computer Vision and Image Understanding*, 62(2):147–163, September 1995.
- [42] A. Hoover. *The Space Envelope Representation for 3D Scenes*. PhD thesis, University of South Florida, 1996.
- [43] A. Hoover, D. Goldgof, and K.W. Bowyer. Extracting a valid boundary representation from a segmented range image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(9):920–924, September 1995.
- [44] A. Hoover, D.B. Goldgof, and K.W. Bowyer. Extracting known and inferred shape information from a single view. In *Proceedings of SPIE Workshop on Sensor Fusion V*, pages 2–13, 1992.
- [45] A. Hoover, D.B. Goldgof, and K.W. Bowyer. Building a b-rep from a segmented range image. In *Proceedings of IEEE Second CAD-Based Vision Workshop*, pages 74–81, February 1994.
- [46] A. Hoover, G. Jean-Baptiste, P.J. Jiang, X. an Flynn, H. Bunke, D. Goldgof, K. Bowyer, D.W. Eggert, A. Fitzgibbon, and R.B. Fisher. An experimental comparison of range image segmentation algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7):1–17, July 1996.
- [47] T. Iberall. The nature of human prehension: Three dextrous hands in one. In *Proceedings of IEEE International Conference on Robotic and Automation*, pages 396–401, 1987.
- [48] T. Iberall, J. Jackson, L. Labbe, and R. Zampano. Knowledge-based prehension: Capturing human dexterity. In *IEEE International Conference on Robotic Research*, pages 82–87, 1988.
- [49] K2T, Incorporated. *GRF-2 User’s Manual, Revision 2.0*. K2T, Incorporated, Duquesne, PA, March 1995.
- [50] K2T, Incorporated. *V300 User’s Manual, Revision 1.1*. K2T, Incorporated, Duquesne, PA, March 1995.
- [51] K. Kawamura, D.M. Wilkes, T. Pack, M. Bishay, and J. Barile. Humanoids: Future robots for home and factory. In *Proceedings of First International Symposium on Humanoid Robots*, pages 53–62, 1996.
- [52] D. Kim and R. Nevatia. A method for recognition and localization of generic objects for indoor navigation. In *IEEE Workshop on Applications of Computer Vision*, pages 280–288, 1994.

- [53] K. Kise, H. Hattori, T. Kitahashi, and K. Fukunaga. Representing and recognizing simple hand-tools based on their functions. In *Asian Conference on Computer Vision*, pages 656–659, November 1993.
- [54] T. Kitahashi, N. Abe, S. Dan, K. Kanda, and H. Ogawa. A function-based model of an object for image understanding. In H. Jaakkola, H. Kanassalo, and S. Ohsuga, editors, *Advances in Information Modelling and Knowledge Bases*, pages 91–97. IOS Press, 1991.
- [55] S.M. Kosslyn, R.A. Flynn, J.B. Amsterdam, and Gretchen Wang. Components of high-level vision: A cognitive neuroscience analysis and accounts of neurological syndromes. *Cognition*, 34:203–277, 1990.
- [56] R. Krishnan. Planning strategies for robotic grasping and wielding. In *AAAI-93 Workshop on Reasoning about Function*, pages 219–220, Washington, D.C., July 1993.
- [57] E. Krotkov. Perception of material properties by robotic probing: Preliminary investigations. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 88–94, August 1995.
- [58] E. Krotkov, R. Klatzky, and N. Zumel. Analysis and synthesis of the sounds of impact based on shape-invariant properties of materials. In *Proceedings of the International Conference on Pattern Recognition*, pages 115–119, August 1996.
- [59] S. Kumar and D. Goldgof. Recovery of global nonrigid motion - a model based approach without point correspondences. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 594–599, June 1996.
- [60] I. Lapierre and C. Laugeau. A road scene understanding system based on a blackboard architecture. In H. Bunke, editor, *Advances in Structural and Syntactic Pattern Recognition*, pages 571–585. World Scientific, 1992.
- [61] T. Lindeberg. *Scale-Space Theory in Computer Vision*. Kluwer Academic Publishers, The Netherlands, 1994.
- [62] M. Mantyla. *An Introduction to Solid Modeling*. Computer Science Press, Rockville, Maryland, 1988.
- [63] D. Marr. *Vision*. W.H. Freeman and Company, New York, 1982.
- [64] J. Maver and R. Bajcsy. Occlusions as a guide to planning the next view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(5):417–433, May 1993.
- [65] P. Michelman and P. Allen. Forming complex dextrous manipulations from task primitives. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 3383–3388, 1994.

- [66] Microbot, Incorporated. *Microbot Alpha Volume I Reference Guide*. Microbot, Incorporated, Mountain View, CA, December 1982.
- [67] Microbot, Incorporated. *Microbot Alpha Volume II Programming Guide*. Microbot, Incorporated, Mountain View, CA, June 1984.
- [68] M. Minsky. *The Society of Mind*. Simon and Schuster, New York, 1985.
- [69] S. Palmer, E. Rosch, and P. Chase. Canonical perspective and the perception of objects. In J. Long and A. Baddeley, editors, *Attention and Performance IX*, pages 135–151. Lawrence Erlbaum Associates, Publishers, 1981.
- [70] S. Pinker. Visual cognition: An introduction. *Cognition*, 18:1–63, 1984.
- [71] A.R.G. Requicha. Representations for rigid solids: theory, methods and systems. *ACM Computing Surveys*, 12:437–464, December 1980.
- [72] E. Rivlin, S.J. Dickinson, and A. Rosenfeld. Recognition by functional parts. *Computer Vision and Image Understanding*, 62(2):164–176, September 1995.
- [73] E. Rivlin and A. Rosenfeld. Navigational functionalities. *Computer Vision and Image Understanding*, 62(2):232–244, September 1995.
- [74] E. Rivlin, A. Rosenfeld, and D. Perlis. Recognition of object functionality in goal-directed robotics. In *AAAI-93 Workshop on Reasoning about Function*, pages 126–130, Washington, D.C., July 1993.
- [75] L.G. Roberts. Machine perception of three-dimensional solids. In *et al.* Tippett, J.T., editor, *Optical and Electro-Optical Information Processing*, pages 159–197. MIT Press, Cambridge, Massachusetts, 1965.
- [76] E. Rosch. Human categorization. In Neil Warren, editor, *Studies in Cross-cultural Psychology*, pages 1–49. Academic Press, 1977.
- [77] E. Rosch, C.B. Mervis, W. Gray, D. Johnson, and P. Boyes-Braem. Basic objects in natural categories. *Cognitive Psychology*, 8:382–439, 1976.
- [78] F. Solina and R. Bajcsy. Shape and function. In *Proceedings of SPIE Volume 726 Intelligent Robots and Computer Vision XI: Fifth in a Series*, pages 284–291, 1986.
- [79] S. A. Stansfield. Haptic perception with an articulated, sensate robot hand. *Robotica*, 10, 1992.
- [80] S.A. Stansfield. Representing generic objects for exploration and recognition. *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1090–1096, 1988.
- [81] L. Stark. Recognizing object function through reasoning about 3-D shape and dynamic physical properties. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 546–553, 1994.

- [82] L. Stark and K.W. Bowyer. Achieving generalized object recognition through reasoning about association of function to structure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):1097–1104, 1991.
- [83] L. Stark and K.W. Bowyer. Generic recognition through qualitative reasoning about 3-D shape and object function. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 251–256, 1991.
- [84] L. Stark and K.W. Bowyer. Indexing function-based categories for generic object recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 795–797, 1992.
- [85] L. Stark and K.W. Bowyer. Function-based recognition for multiple object categories. *CVGIP: Image Understanding*, 59(1):1–21, January 1994.
- [86] L. Stark and K.W. Bowyer. *Generic Object Recognition Using Form and Function*. World Scientific Publishing, New Jersey, 1996.
- [87] L. Stark, L.O. Hall, and K.W. Bowyer. An investigation of methods of combining functional evidence for 3-D object recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(3):573–594, 1993.
- [88] L. Stark, A. Hoover, D.B. Goldgof, and K.W. Bowyer. Function-based recognition from incomplete knowledge of shape. In *Proceedings of IEEE Workshop on Qualitative Vision*, pages 11–22, 1993.
- [89] L.H. Sullivan. The tall office building artistically considered. *Lippincott's Magazine*, March 1896.
- [90] M.A. Sutton, H. Liu, L. Stark, and K.W. Bowyer. Towards a domain-independent function-based recognition system. In *Proceedings of Florida Artificial Intelligence Research Symposium*, pages 294–298, 1995.
- [91] M.A. Sutton, L. Stark, and K.W. Bowyer. Capturing function in a generic representation scheme. In *Proceedings of 8th Israeli Symposium on Artificial Intelligence and Computer Vision*, pages 97–111, 1991.
- [92] M.A. Sutton, L. Stark, and K.W. Bowyer. ‘We do dishes, but we don’t do windows,’ Function-based modeling and recognition of rigid objects. In *Proceedings of SPIE Workshop on Intelligent Robots and Computer Vision XI: Algorithms, Techniques and Active Vision*, pages 132–142, 1992.
- [93] M.A. Sutton, L. Stark, and K.W. Bowyer. What is a “generic” object model for computer vision? In *Proceedings of Florida Artificial Intelligence Research Symposium*, pages 252–256, 1992.
- [94] M.A. Sutton, L. Stark, and K.W. Bowyer. Function-based generic recognition for multiple object categories. In A.K. Jain and P.J. Flynn, editors, *Three-Dimensional Object Recognition Systems*, pages 447–470. Elsevier Science Publishers, 1993.

- [95] M.A. Sutton, L. Stark, and K.W. Bowyer. GRUFF-3: Generalizing the domain of function-based recognition system. *Pattern Recognition*, 27(12):1743–1766, December 1994.
- [96] M.A. Sutton, L. Stark, and K.W. Bowyer. Interactive confirmation of object functionality. In *AAAI Workshop on Modeling and Reasoning with Function*, August 1996.
- [97] M.A. Sutton, L. Stark, and K.W. Bowyer. Representing and reasoning about object functionality: Towards an integrated approach. In R.C. Bolles, H. Bunke, and H. Noltmeier, editors, *Intelligent Robots - Sensing, Modelling, and Planning*. World Scientific Publishing, 1997 (to appear).
- [98] M. Trobina and A. Leonardis. Grasping arbitrarily shaped 3-D objects from a pile. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 241–246, 1995.
- [99] B. Tversky and K. Hemenway. Categories of environmental scenes. *Cognitive Psychology*, 15:121–149, 1983.
- [100] B. Tversky and K. Hemenway. Objects, parts, and categories. *Journal of Experimental Psychology: General*, 113(2):169–193, June 1984.
- [101] L. Vaina and M. Jaulent. Object structure and action requirements: A compatibility model for functional recognition. *International Journal of Intelligent Systems*, 6:313–336, 1991.
- [102] P. Winston, T. Binford, B. Katz, and M. Lowry. Learning physical descriptions from functional definitions, examples, and precedents. *AAAI*, pages 433–439, 1983.
- [103] T. Wohlers. 3D digitizing systems. *Computer Graphics World Magazine*, pages 59–61, April 1994.

APPENDICES

APPENDIX A

SYSTEM TERMINOLOGY

This research encompasses a number of terms from artificial intelligence, computer vision and robotics. The following definitions are provided as a reference:

- Active and real-time vision** - Vision in which scene interpretation controls data acquisition in a way to facilitate the current task.
- Association measure** - A numerical value in the range 0.0 to 1.0 which indicates how well the shape satisfies the functional requirements of some category of objects.
- CAD-based vision** - When there is a strong correspondence between objects in the world and individual object models in the system.
- Category** - Refers to a basic level category. Examples include “cup” and “chair.”
- Category definition tree** - The graphical representation of the function-based definition of the categories known to the **GRUFF-I** system.
- Density (of an OPUS model)** - This measure is determined by dividing the total number of modeled range points from a range image by the total area of real planes in the recovered model. This value expresses the degree of ‘exaggeration’ in the reconstructed model.
- Dilation** - An image processing **morphological operation** which fills in areas in the image upon which a structuring element is centered.
- Endpoint-closed-loop system** - A robot positioning system which uses direct image feedback of the manipulator so 3-D positioning can be accurate regardless of calibration.
- Endpoint-open-loop system** - A robot positioning system which does not observe the manipulator, and therefore its target accuracy depends directly on the hand-eye transformation. This approach is used in the **GRUFF-I** system.
- Erosion** - An image processing **morphological operation** which returns all locations where a structuring element is contained in the image.
- Evaluation measure** - A measure in the range 0.0 to 1.0 which reflects the strength of the association of a **function label** to a **functional element** in the 3-D shape of an object.

- Function-based vision** - The idea that classes of objects can be defined and recognized by functional characteristics such as *provides containment* or *provides sittability*.
- Functional element** - A portion of the input object that fulfills the functional requirements associated with a specific **function label**. There are three types of functional elements that can be identified: (1) a single surface of the object, such as the face of the chair which provides a sittable surface; (2) a virtual surface formed by a group of surfaces which act together to fulfill the required function, such as slats on the back of a chair which act together to provide back support; and (3) a three-dimensional portion (module) of the structure or the entire object structure (such as for stability).
- Function from X** - A number of methods for accomplishing object recognition by extracting “functional” information from input images, models, or interaction.
- Function from manipulation/navigation** - A method for object recognition which addresses recovering the function of an object by manipulating or navigating around it.
- Function from motion** - An approach which attempts to recognize the function some object is serving by observing a task being performed with the object.
- Function from shape** - A technique for object recognition which uses the idea that the 2-D or 3-D shape of an object provides some indication of its function.
- Function label** - A name for the functional property being evaluated. An example would be *provides graspability*.
- Function stable orientation** - Refers to a shape and orientation which is stable when forces are applied to it, corresponding to how the shape is to be used.
- Function verification plan** - The interaction-based functional tests which must be passed in order for the object to be recognized as a member of a category. For example, to be used as a cup, the object must pass tests such as *confirm graspability* and *confirm containment*.
- Generic object recognition** - Recognition of objects as members of classes of objects, as opposed to identification of a specific object.
- GRUFF** - **G**eneric **R**ecognition **U**sing **F**orm and **F**unction. This is the original shape-based recognition system developed by Stark, *et al.* [83].

GRUFF-I - **Generic Recognition Using Form, Function and Interaction.**
The system described in this dissertation, which is a modified version of the original **GRUFF** system, in that it incorporates actual physical devices for vision and robotic subsystems.

Hand-eye transformation - Determination of the parameters necessary to convert from a camera centered coordinate system to a world coordinate system.

Hypothesized functional plan - The shape-based functional definition of a specific category or subcategory of objects. For example, to function as a cup, the 3-D shape of an object must be able to *provide stability, provide accessibility* and *provide containment*.

Indexing measure - A measure used to prune the search space of categories to consider.

KPs (Knowledge Primitives) - The building blocks for different functional properties. They return a measure in the range 0.0 to 1.0 which indicates how well a particular primitive invocation is satisfied by the shape. The **GRUFF-I** system uses six shape-based primitives: dimensions, clearance, stability, proximity, enclosure and relative orientation, and two interaction-based primitives: apply force and observe deformation.

Model - The recovered 3-D OPUS model of the input **object**.

Model-based vision - See **CAD-based vision**.

Morphological operations - Image processing operations which are used for shape analysis in images. Basic operations include **erosion** and **dilation**. A dilation-erosion sequence is used to fill in undesirable holes in the image, without removing noise. A erosion-dilation sequence is used to remove isolated pixels smaller than a structuring element.

Object - The physical artifact placed in the scene area.

OPUS model - Object Plus Unseen Space model. The 3-D boundary representation model, created from a segmented range image. The OPUS model captures the visible real faces and introduces faces along the line of sight referenced as occlusion faces. The resulting OPUS model of the object is a valid 3-D solid model enclosing the object and its surrounding occluded space.

PKP (Procedural Knowledge Primitive) - Simply the implementation of a KP as a procedure.

Purposive vision - Vision in which only the information necessary to accomplish the current system goal is extracted from the scene.

Range image - An array of distance measurements from a reference point to points in a scene.

Residual (of an OPUS model) - The average of the distances between the range points in a range image and the fitted surfaces in the recovered model derived from this image.

Scale space processing - Applying different sets of parameters to an image to obtain more than one set of results [61]. Scale space processing could be incorporated into the Model Building Sub-system by applying different sets of parameters to the range image until a valid 3-D **OPUS model** is formed.

Segmentation - The labeling of each pixel in an image, such that pixels that belong to the same surface possess the same label [42].

Self stable orientation - Refers to a shape and orientation which is stable when no forces are applied to it.

Shadow areas - These areas occur when using a structured light scanner, whenever the camera views a point which the projector doesn't illuminate. Such areas tend to occur when the surface of the object obstructs the projector patterns.

Structured light scanner - A vision system composed of a projector and a CCD camera which allows the recovery of a range image of the scene area.

Subcategory - Categories which are specializations of basic level categories (also called subordinate categories). An example is "mug."

Superordinate category - A generalization of a basic category. An example is "furniture."

Thresholding - An image processing operation which produces a binary image from a gray scale image.

Triangulation - The process of determining the (x, y, z) coordinates of a three dimensional point.

APPENDIX B

HISTORY OF THE GRUFF SYSTEM

The GRUFF (Generic Recognition Using Form and Function) research project has sought to explore the role that knowledge about object functionality plays in object recognition. The initial phase of this project concentrated on developing and demonstrating prototype systems whose domain of competence would be selected categories of static, rigid objects which were described purely in terms of their geometric shape. The initial demonstration system was capable of reasoning about the object category *chair* [82, 83]. The first system was enhanced to have a domain of competence of multiple distinct basic level categories (chair, bench, bed, table and bookshelf) [84, 85]. This GRUFF-2 system was further enhanced to include multiple basic level categories within the superordinate category *dishes* [91, 92, 94, 95], and later with object categories in the superordinate category *handtools* [90], as shown in Figure 47. An analysis of the complexity of the shape-based processing for rigid objects is presented in [94, 95].

Versions of the system up to this point had assumed that the input description of an object would be a complete rigid 3-D shape description. In a parallel line of research, members in the USF computer vision group developed the “Object Plus Unseen Space” (OPUS) model as a means of representing the qualitatively different types of shape information that could be extracted from one or several views of an object from images obtained from a laser range finder mounted on a mobile robot [43, 44, 45, 88]. Work in a simulated environment explored the use of such information to provide a plan for interaction to indicate how a robot might efficiently interact with the object [81]. Finally, an additional line of work focused on non-rigid objects such as scissors [33]. These developments are summarized pictorially in Figure 48 to clarify the scope of this dissertation in the context of the entire **GRUFF** project.

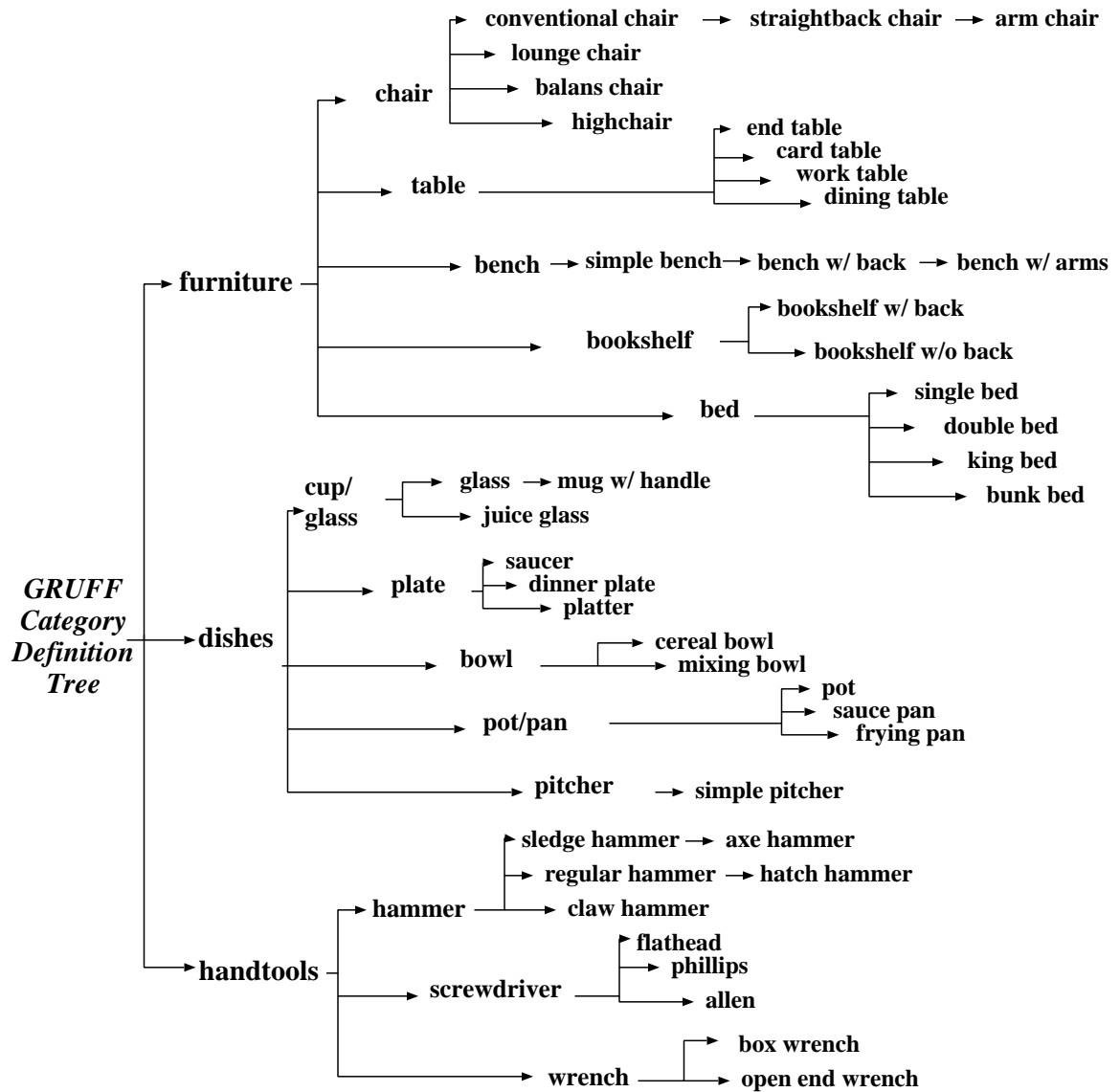


Figure 47. Complete category definition tree. This tree contains information about all the categories of shapes known to the **GRUFF** system.

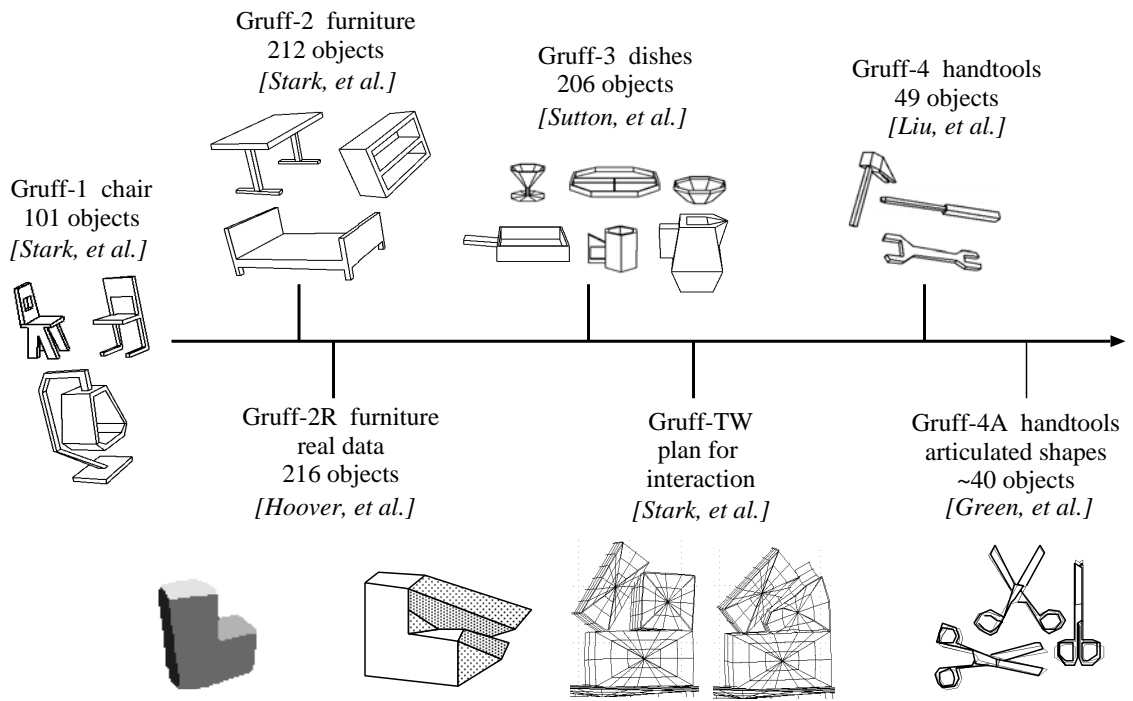


Figure 48. History of the **GRUFF** system approach.

APPENDIX C

SYSTEM PARAMETERS

The Model Building Subsystem is based on two algorithms developed by Hoover [42]. The meaning of each of the parameters used in the first stage for range image segmentation is as follows:

ImageTypeFlag - Input format for the K2T GRF-2: 480x640; 8bit; range is bits 0...7.

MaxPerpDist - During segmentation, the maximum perpendicular distance (in range values) between the equation of the surface grown so far and a neighboring pixel to be added.

MaxPointDist - During segmentation, the maximum point-to-point distance (in range values) between a point to be added to a region and a point already in the region.

MinRegionPixels - During segmentation, the minimum size (in pixels) of a region in order to be retained.

ReportFlag - Output format flag indicating notification upon entry to major procedures.

SegTypeFlag - Output format flag for segmentation: labels are 0..n-1; 255 for undefined.

The meaning of each of the parameters used in the second stage for OPUS model building is as follows:

HighEdgeDist - During jump edge detection, this represents the high value for hysteresis thresholding. In order to create a jump edge, at least one pixel in the edge must be this distance from the intersection of the two regions bordering the edge.

LongGlueEdge - During gluing, this represents the length of the outer-patch edge used to expand the pt-link to split the big patch.

LowEdgeDist - Threshold value for edge pixels; decides crease or jump edge pixel; value refers to distance (in range values) between the pixel and the line of intersection of the two regions bordering the edge.

MaskFlag - Indicates that shadow areas should be labeled as 'the floor.'

Table 5. Thresholds for Dimensions Primitive

PRIMITIVE	dimension type	least	lo_ideal	hi_ideal	greatest
dimensions	height (meters)	0.020	0.025	0.060	0.080
	width (meters)	0.040	0.050	0.060	0.080
	depth (meters)	0.040	0.050	0.060	0.080

MaxPerpDist - During smoothing, the maximum perpendicular distance (in range values) between the equation of the surface grown so far and a neighboring pixel to be added (also used in floor heuristic).

MaxPointDist - During smoothing, the maximum point-to-point distance (in range values) between a point to be added to a region and a point already in the region.

MinEdgeAngle - During edge segmentation, the minimum angle allowed to exist in a single edge. If any pixel in the edge, combined with the two end pixels of the edge, makes an angle (2-D) smaller than this, then the edge is broken into two pieces.

MinEdgeLength - During edge segmentation, the minimum length (2-D, in pixel edges) allowed in a final edge.

MinGlueDist - During gluing, threshold used to apply distance criteria.

MinRegionPixels - This represents the minimum size (in pixels) of a region to be retained in building the model.

ShortGlueEdge - During gluing, the length of outer-patch edge used to expand the pt-link for a new tri-vertex.

There is also a set of thresholds and parameters for the Shape-based Reasoning Subsystem. For example, this subsystem uses a set of predefined dimensional constraints for each primitive invocation, as shown in Table 5 for the **dimensions** primitive.

VITA

Melanie A. Sutton received the B.S. degree in Computer Engineering and the M.S. degree in Computer Science from the University of South Florida in 1992 and 1993, respectively. Melanie joined the Computer Science Department at the University of West Florida in 1996. Her current research interests include image processing (including medical applications), computer vision and robotics. For more information on the **GRUFF** project, including publications and 3-D models which are available electronically, visit the USF Computer Vision / Image Analysis Research Lab Web page, located at <http://marathon.csee.usf.edu/>. Melanie's personal Web page can be reached at <http://www.cs.uwf.edu/~msutton/>.