Recognition by Functional Parts

Ehud Rivlin
Department of Computer Science
Technion–Israel Institute of Technology
Haifa, Israel

Sven J. Dickinson
Department of Computer Science
University of Toronto, 6 King's College Rd.
Toronto, Ontario, Canada M5S 1A4

Azriel Rosenfeld
Center for Automation Research
University of Maryland
College Park, MD, USA 20742

Abstract

We present an approach to function-based object recognition that reasons about the functionality of an object's intuitive parts. We extend the popular "recognition by parts" shape recognition framework to support "recognition by functional parts", by combining a set of functional primitives and their relations with a set of abstract volumetric shape primitives and their relations. Previous approaches have relied on more global object features, often ignoring the problem of object segmentation and thereby restricting themselves to range images of unoccluded scenes. We show how these shape primitives and relations can be easily recovered from superquadric ellipsoids which, in turn, can be recovered from either range or intensity images of occluded scenes. Furthermore, the proposed framework supports both unexpected (bottom-up) object recognition and expected (top-down) object recognition. We demonstrate the approach on a simple domain by recognizing a restricted class of hand-tools from 2-D images.

1 Introduction

When we consider recognition from a functional point of view, we leave the concept of shape-alone based recognition for a more general and flexible concept. For example, if we wish to model four chairs, each having a different configuration of differently shaped parts but all functioning as chairs, we would require four different object shape models. Alternatively, recognition based on functionality would enable a robot to possess knowledge of the needed function of a chair without explicitly specifying the possible shapes of a chair. The seminal work of Stark and Bowyer et al. [11, 10, 12, 13, 14, 15, 16, 16] has addressed function-based object recognition, focusing on domains including chairs and dishes. In their work, they define a set of functional primitives specific to each object class. For example, in their system that recognizes chairs, they have functional primitives for support, sitting height, stability, etc. From a CAD representation of an object, they can compute these primitives and categorize the object. Although their system has been tested mainly with CAD data, they have applied it to complete range images of an object acquired through an Odetics range scanner.

Despite the success of this approach, there are some limitations. To begin with, the approach assumes a 3-D representation of the image from which they can compute their functional primitives. Furthermore, their approach assumes an image of an isolated object; object occlusion in the image cannot be supported since no object segmentation is performed on the image data. It is important to note that the work of Stark and Bowyer takes a *global* approach to functional recognition, making it sensitive to occlusion and partial views. Due to the nature of their functional reasoning, it does not extend to function-based recognition from 2-D imagery containing multiple occluded objects.

In this paper, we present a theory of function-based recognition which is a natural extension of part-based shape recognition. Instead of focusing on global properties such as stability, height, existence of large horizontal surfaces, etc., we will reason about the functionality of an object's parts. Moreover, those parts are the *same* parts that we recover from the image for shape recognition. Thus, instead of reasoning about the functionality of a collection of 3-D points or planar

1063-6919/94 $3.00 © 1994 IEEE

surfaces, we propose to reason about a more intuitive notion of an object's parts (Pentland [9]). Although we will not index using part shape, we can use knowledge of part shape to help segment the image into parts. Given a set of recovered volumetric parts, we can then reason about the functionality of both the individual parts *and* interactions between the parts. Such interactions can include relative orientation, size, shape, or even motion!

We outline our theory of functionality for objects in Section 2, and introduce a representation for volumetric parts from which we reason about functionality. Section 3 discusses the recovery of the volumetric parts from both 3-D range and 2-D intensity images. Next, in Section 4, we describe our recognition algorithm as it applies to both expected (top-down) and unexpected (bottom-up) object recognition. Finally, in Section 5, we demonstrate the technique applied to the domain of hand tools.

2 Representing Object Functionality

Our theory of function-based object recognition is a natural extension of part-based shape recognition. That is, we reason about the functionality of an object's parts and their interrelations. Figure 1 illustrates the concept. At the shape level, objects are constructions of coarse volumetric primitives with spatial relations between the primitives. At the function level, the shape primitives map to a set of functional primitives and the spatial relations map to a set of functional relations. At the functional level, objects are not represented in terms of shape, but in terms of a set of functional primitives and relations. In the following sections, we describe this hierarchical representation in more detail. We begin by describing the coarse shape representation and follow with the functional representation. Finally, we illustrate the representation by means of an example.

2.1 Representing Shape

2.1.1 Shape Primitives

Our shape representation models objects as constructions of coarse volumetric shape primitives belonging to four classes: sticks, strips, plates, and blobs. The representation is an extension to the generalized blob models (sticks, plates, and blobs) proposed by Mulgaonkar, Shapiro, and Haralick [8]. Our four classes are distinguished by their relative dimensions. Letting

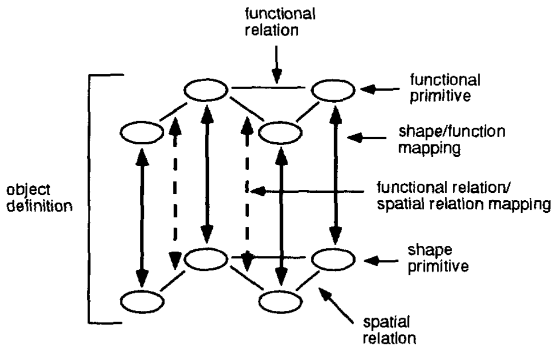

Figure 1: Representing Object Functionality

a_1, a_2, and a_3 represent length, width, and height, respectively, of a volumetric part, we can define the four classes as follows:

$$\begin{aligned}
&\textit{Stick}: && a_1 \simeq a_2 \ll a_3 \lor a_1 \simeq a_3 \ll a_2 \lor a_2 \simeq a_3 \ll a_2\\
&\textit{Strip}: && a_1 \neq a_2 \land a_2 \neq a_3 \land a_1 \neq a_3\\
&\textit{Plate}: && a_1 \simeq a_2 \gg a_3 \lor a_1 \simeq a_3 \gg a_2 \lor a_2 \simeq a_3 \gg a_2\\
&\textit{Blob}: && a_1 \simeq a_2 \simeq a_3
\end{aligned}$$

Intuitively, if all three dimensions are about the same, we have a blob. If two are about the same and the third is very different, we have two cases: if the two are bigger than the third, we have a plate, while if the two are smaller than the third, we have a stick. Finally, when no two dimensions are about the same, we have a strip. For example, a knife blade is a strip, because no two of its dimensions are similar.

2.1.2 Spatial Relations

We can qualitatively describe the ways in which two shape primitives can be combined. For example, we can attach two shapes end-to-end, end-to-side, or side-to-side, as proposed by Biederman when building objects out of geons [2]. To further specify these attachments, we adopt the convention of labeling each primitive's attachment surfaces [6]. For example, a square plate has six attachment surfaces, while a cylindrical stick has three attachment surfaces. For simplicity, we shall require any junction of two primitives to involve exactly one attachment surface from each primitive. In addition to specifying the two attachment surfaces participating in the junction of two primitives, we can also consider the angles at which they join, and we can classify the joints as perpendicular, oblique, tangential, etc. Another refinement would be to qualitatively describe the position of the joint on each surface.

2.2 Representing Function

2.2.1 Functional Primitives

Functional primitives represent the building blocks of a functional representation of an object. For example, the functional primitives defining a coffee cup would include a handle and a container; a chair would include a seat, a base, and a back [11, 10]. For the remainder of this paper, we will illustrate our approach to functional object recognition by focusing on a class of manipulation tasks. Bearing in mind that a manipulation task involves an agent grasping an object and using it to perform some action, we will define a class of objects that have an ***end-effector*** (the part which delivers the action) and a ***handle*** (the part that the agent grasps). Examples of such objects might include simple hand tools like a screw driver or a hammer, or everyday objects like cups, glasses, or plates.

2.2.2 Functional Relations

A given set of parts might independently satisfy the needs for an end-effector or a handle. However, they must be joined in a particular way so as to satisfy the needs of a particular task. The set of functional relations linking the primitives describes the function of the interaction between the primitives. In the hammer example, the functional relation linking the handle and end-effector specifies that the handle is used to swing the end-effector in a direction which maximizes the force tangential to the swing arc while maximizing striking stability.

2.3 Mapping Shape to Function

In general, the mapping between shape primitives (and their relations) and functional primitives is many-to-one. For example, three or more chair legs may satisfy the functional primitive of chair base. For simplicity, we will restrict ourselves to object models with a one-to-one mapping between shape primitives and functional primitives. Consider, for example, the functional model for a hammer specifying an end-effector and a handle. The end-effector should be blob-like, ensuring that the dimensions of the striking surface are roughly equal (rotationally symmetric to allow striking error in any direction). If the end-effector were stick-like, the distance between the handle junction and the striking surface would be large, making it more difficult to locate the nail. If the end-effector were plate-like, it would have insufficient momentum for driving a nail. The handle, on the other hand, should be stick-like, small enough that it can be grasped by a human hand, and long enough to provide a high moment at its junction with the end-effector.

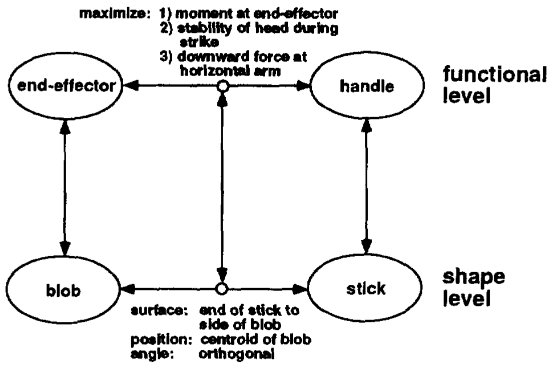
maximize: 1) moment at end-effector
2) stability of head during strike
3) downward force at horizontal arm
end-effector
handle
functional level
blob
stick
shape level
surface: end of stick to side of blob
position: centroid of blob
angle: orthogonal

Figure 2: Functional Model for a Hammer

2.4 Mapping Function Relations to Spatial Relations

The specification as to how the functional components defining an object are combined is captured by a set of functional relations. These functional relations are then mapped to a set of spatial relations linking the shape primitives. In the hammer example, the functional relation maps to an attachment between the stick (handle) and the blob (end-effector) such that the axis of the stick is orthogonal to the (principal) axis of the blob and is attached to the centroid of the blob. The complete model for the hammer, including functional and shape primitives, functional and shape relations, and the mapping from functional shapes and relations to spatial shapes and relations is outlined in Figure 2.

3 Recovering Shape

In the last section, we described a set of functional primitives defined on a set of shapes consisting of sticks, strips, plates, and blobs. Since these four shape classes are defined according to their relative dimensions, we need to not only segment an input image into parts, but recover 3-D (dimensional) information from those parts. In this section, we describe an approach to recovering sticks, strips, plates, and blobs from an image. The approach consists of recovering a superquadric from the image, providing explicit dimensions which we can then use to classify our shape. Superquadrics offer a compact, coarse, volumetric description of an object's parts [9]. If finer shape modeling is required, deformable superquadrics can be used to capture both global part shape (using a superquadric) and local shape (using a deformable mesh) [17]. Since superquadrics capture more shape attributes than just the x, y, and z dimensions of a part, they provide us with a foundation from which to recover a richer vocabulary of qualitative shapes with which to reason about function. For example, we may decide to distinguish among curved-axis vs. straight-axis shapes or tapering vs. constant cross-sectional sweep rules [2].

The approach we take, due to Dickinson and Metaxas [4, 7], is to use a qualitative segmentation of the image to provide strong constraints on the deformable model fitting procedure described in [17]. The result is a technique which allows us to recover certain classes of superquadrics from image data, under orthographic, perspective, and stereo projection [7]. Furthermore, the technique supports the recovery of occluded parts, allowing us, unlike the work of Stark and Bowyer, to reason about the functionality of objects that are only partially visible. We will not describe the above recovery methods in this paper; details can be found in [4, 7]. We will, however, proceed now to describe the geometry of a deformable superquadric and show how we classify a superquadric as a stick, strip, plate, or blob.

3.1 Geometry of a Deformable Superquadric

Geometrically, the models that we can recover from either range or image data are closed surfaces in space whose intrinsic (material) coordinates are $\mathrm{u} = (u, v)$, defined on a domain Ω. The positions of points on the model relative to an inertial frame of reference Φ in space are given by a vector-valued, time varying function of u:

$$\mathbf{x}(\mathrm{u}, t) = (x_1(\mathrm{u}, t), x_2(\mathrm{u}, t), x_3(\mathrm{u}, t))^\top, \quad (1)$$

where $^\top$ is the transpose operator. We set up a noninertial, model-centered reference frame ϕ, and express these positions as

$$\mathbf{x} = \mathbf{c} + \mathbf{R}\mathbf{p}, \quad (2)$$

where $\mathbf{c}(t)$ is the origin of ϕ at the center of the model and the orientation of ϕ is given by the rotation matrix $\mathbf{R}(t)$. Thus, $\mathbf{p}(\mathrm{u}, t)$ denotes the canonical positions of points on the model relative to the model frame. We further express $\mathbf{p}$ as the sum of a reference shape $\mathrm{s}(\mathrm{u}, t)$ and a displacement function $\mathbf{d}(\mathrm{u}, t)$:

$$\mathbf{p} = \mathrm{s} + \mathbf{d}. \quad (3)$$

The ensuing formulation can be carried out for any reference shape given as a parameterized function of u. Based on the shapes we want to recover (sticks, strips, plates, and blobs with possible tapering and bending global deformations), we first consider the case of superquadric ellipsoids [1], which are given by the following formula:

$$\mathbf{e} = a \begin{pmatrix} a_1 C_u{}^{\epsilon_1} C_v{}^{\epsilon_2} \\ a_2 C_u{}^{\epsilon_1} S_v{}^{\epsilon_2} \\ a_3 S_u{}^{\epsilon_1} \end{pmatrix}, \quad (4)$$

where $-\pi/2 \leq u \leq \pi/2$ and $-\pi \leq v < \pi$, and where $S_w{}^{\epsilon} = \mathrm{sgn}(\sin w)|\sin w|^{\epsilon}$, and $C_w{}^{\epsilon} = \mathrm{sgn}(\cos w)|\cos w|^{\epsilon}$, respectively. Here, $a \geq 0$ is a scale parameter, $0 \leq a_1, a_2, a_3 \leq 1$ are aspect ratio parameters, and $\epsilon_1, \epsilon_2 \geq 0$ are "squareness" parameters.

We then combine linear tapering along principal axes 1 and 2, and bending along principal axis 3 of the superquadric $\mathbf{e}$[1] into a single parameterized deformation $\mathbf{T}$, and express the reference shape as

$$\mathrm{s} = \mathbf{T}(\mathbf{e}, t_1, t_2, b_1, b_2, b_3) = \quad (5)$$

$$\begin{pmatrix} (\frac{t_1 e_3}{a a_3 w} + 1)\, e_1 + b_1\ cos(\frac{e_3 + b_2}{a a_3 w} \pi b_3) \\ (\frac{t_2 e_3}{a a_3 w} + 1)\, e_2 \\ e_3 \end{pmatrix}, \quad (6)$$

where $-1 \leq t_1, t_2 \leq 1$ are the tapering parameters in principal axes 1 and 2, respectively, and where b_1 defines the magnitude of the bending and can be positive or negative, $-1 \leq b_2 \leq 1$ defines the location on axis 3 where bending is applied and $0 < b_3 \leq 1$ defines the region of influence of bending. Our method for incorporating global deformations is not restricted to only tapering and bending deformations. Any other deformation that can be expressed as a continuous parameterized function can be incorporated as our global deformation in a similar way.

We collect the parameters in s into the parameter vector:

$$\mathbf{q}_s = (a, a_1, a_2, a_3, \epsilon_1, \epsilon_2, t_1, t_2, b_1, b_2, b_3)^\top. \quad (7)$$

Once we have recovered a superquadric from an image (range or intensity), it is a very simple matter to

[1]These coincide with the model frame axes x, y and z respectively.

extract the dimensions of the superquadric. The width (x dimension) of the superquadric is given by

$$width = aa_1, \tag{8}$$

its height (y dimension) by

$$height = aa_2, \tag{9}$$

and its length (z dimension) by

$$length = aa_3. \tag{10}$$

Given the dimensions of the part, we can classify the part as either a stick, strip, plate, or blob according to the rules described in Section 2.

4 Recovering Object Function

Our function-based object recognition strategy supports bottom-up (or unexpected) object recognition, whereby an object is presented to the system and the system identifies the object based on the functionality of its parts. In addition, our strategy supports top-down (or expected) object recognition, whereby the system looks for a particular object in the image by mapping its functional parts to image feature predictions. In this section, we will describe both these strategies.

Unexpected Object Recognition

In an unexpected object recognition task, we first segment an input image into a set of homogeneous regions from which we recover a set of qualitative 3-D parts using local part-based aspect matching techniques [6, 5, 3]. Next, using the techniques of Dickinson and Metaxas [4, 7], we use the recovered qualitative shape to constrain the fitting of a set of deformable superquadrics to the qualitative parts. From the resulting quantitative parts, we compare the dimensions of the parts to abstract a set of sticks, strips, plates, and blobs. Furthermore, we can recover the spatial relations spanning the recovered parts.

If there is no a priori knowledge of what object is in the image, then groups of spatial primitives and their spatial relations can be used to infer a set of functional primitives and relations. The recovered functional primitives and relations are then compared to a set of functional object models. In our simple domain of hand tools, we can map shape primitives to possible functional primitives and map shape relations to possible functional relations, providing a number of functional object hypotheses that are then compared to the object database. As an example, suppose we place a hammer in front of the camera and ask the system to identify the object. The recovery process would recover a stick and a blob in some spatial configuration. The blob then maps to end-effector as well as to all other functions a blob could serve. Similarly, the stick maps to a handle as well as all other functions that it could serve. Finally, the spatial relation between the stick and blob would map to all functional relations joining a stick and a blob in that configuration. Combining the various interpretations for the stick, the handle, and their relationship would yield a number of object hypotheses which satisfy the recovered functionality.

Expected Object Recognition

In an expected object recognition task, we use knowledge of the target object's functional model to constrain our search in the image both in terms of what we look for and where we look for it. Given a functional object model, we first choose some functional primitive whose presence in the image would provide the least ambiguous mapping to the target object. For example, in looking for a cup on a table containing glasses and cups, we should look for a cup handle and not a container since the handle is unique to the cup. Next, the functional primitive is mapped to one of the four abstract shape primitives, i.e., sticks, strips, plates, and blobs. Finally, the shape primitive is mapped into an image region shape prediction in terms of extent or elongation. Like the unexpected object recognition algorithm, the image is first processed to extract a region topology graph. By examining the extent (or elongation) of an image region, along with that of its immediate neighbors, we can derive a simple heuristic for drawing attention to a particular image region. We can thus focus the recovery of the shape primitive and constrain the search for other primitives belonging to the object.

For example, if we are searching for blobs or plates, we can rank-order the image regions by increasing extent. Furthermore, regions whose immediate neighbors include a region with similar extent can be favored as being part of a blob, while regions whose neighbors do not include a region with similar extent can be favored as being part of a plate. Similarly, if searching for sticks or strips, we can rank-order the image regions by decreasing extent. Regions whose immediate neighbors include a region with similar extent can be favored as being part of a stick, while regions whose neighbors do not include a region with similar extent can be favored as being part of a strip. These rules can provide a useful ordering on the positions from which shape recovery is attempted.

From a candidate search position, the next step is to recover a superquadric from which the 3-D part dimensions and orientation can be recovered. This consists of first recovering the qualitative shape of the part [6, 5], which is then used to constrain the fitting of a superquadric to the image data. Once the part is verified as a stick, strip, plate, or blob, the search for other parts of the object can be constrained to those image regions adjacent to or in the vicinity of any previously recovered volumes.

5 Results

In this section, we apply the function-based expected object recognition algorithm to the image of the mallet shown in Figure 3(a). In Figure 3(b), we show the segmented region image. Without any a priori knowledge of scene content, each of the functional primitives, namely the end-effector and handle, are deemed equally likely to appear in the image. The algorithm arbitrarily chooses the end-effector (mallet head) and maps that choice to a search in the image for a blob. The algorithm rank-orders regions in the image according to their ratio of area to extent (computed from bounding box). The large region is chosen first and the bottom-up algorithm is used to recover the most likely interpretation of the region and its neighbors. The two most likely recovered volumes are found, corresponding to the head and handle of the mallet, respectively.

In Figures 3(c) and (d), we show the results of using the recovered qualitative shape to constrain the fitting of a superquad to each part; the parameters of the two superquads are given in Table 1. Since only a monocular image was used, the same arbitrary depth was chosen for both objects during the fitting stage. Without recovering true depth of the two parts, we cannot ensure that they intersect.[2] However, in this case, since the two parts intersect in the image, we will assume that they intersect in 3-D.

From the recovered superquad parameters in Table 1, we can proceed to classify each part as either a stick, a strip, a plate, or a blob according to Equations 8, 9, and 10 in Section 2.1.1; the results are shown in Table 2. Using Equations 1, 1, 1, and 1, and defining two dimensions as similar

[2]See [7] for an approach to deformable model recovery from stereo pairs.

Superquad Parameter	**Part**	
	Head	**Handle**
a	37.19	37.19
a_1	0.45	0.22
a_2	0.45	0.22
a_3	0.69	1.14
t_x	-4.40	4.97
t_y	0.51	-3.88
t_z	-50.0	-50.0
r_{11}	0.49	0.54
r_{12}	-0.22	0.07
r_{13}	-0.84	0.84
r_{21}	-0.14	0.78
r_{22}	0.93	0.27
r_{23}	-0.33	-0.53
r_{31}	0.86	-0.26
r_{32}	0.28	0.96
r_{33}	0.42	0.09
ϵ_1	0.0	0.0
ϵ_2	1.0	1.0
$bend_z$	0.0	0.0
$taper_z$	0.0	0.0

Table 1: Recovered Superquad Parameters for Mallet

if the ratio of the biggest to the smallest is within 4:1 (width:height:length ratios for the two parts are 1:1:1.53 for the head and 1:1:5.18 for the handle), then the mallet head is classified as a blob, while the mallet handle is classified as a stick.

Since our search procedure is looking for the mallet head (end-effector), it chooses the blob, and proceeds to search for the handle in the vicinity of the recovered blob. Due to region undersegmentation, the regions corresponding to the body surfaces of the head and handle of the mallet were joined. However, those contours not used to recover the head but still belonging to the large region are free to be part of other recovered volumes. Since we have already recovered a stick *and* its defining contours were not used to infer the blob, we can instantiate the handle in the image. The last step in recognizing the object is to satisfy the functional relation between the two parts which is mapped into a spatial constraint on the part junction. Since the computed relative orientation of the two parts is such that their z axes are orthogonal ($>$ 60 deg in our qualitative partitioning of angle), and since the junction occurs at the end of the handle and at the middle of the head, the algorithm successfully verifies

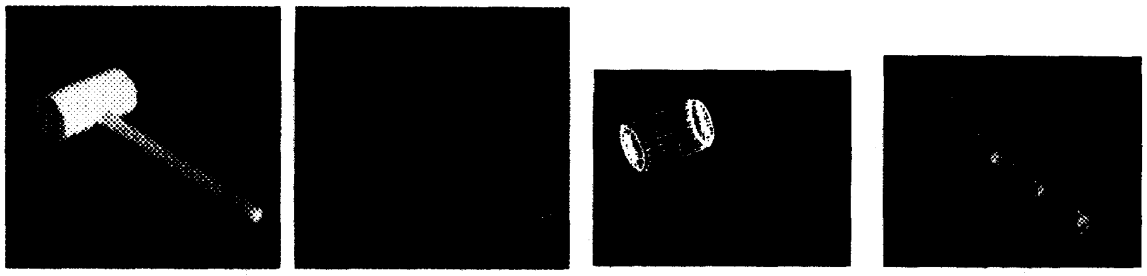

Figure 3: Hammer Recovery: original image, segmented region image, recovered head, and recovered handle.

Dimension	Part	
	Head	Handle
width	16.74	8.18
height	16.74	8.18
length	25.66	42.40

Table 2: Recovered Dimensions for Mallet

Dimension	Part	
	Block	Cylinder
width	20.08	16.74
height	20.08	16.74
length	50.58	14.88

Table 3: Recovered Dimensions for Unknown Object

the hammer in the image.

In the second example, we apply our function-based unexpected object recognition approach to a scene containing a short cylinder attached to the side of a block; the image is shown in Figure 4(a), while the segmented region image is shown in Figure 4(b). Figures 4(c) and (d), show the recovered superquadrics for the block and cylinder, respectively.

From the recovered superquad parameters, we can proceed to classify each part as either a stick, a strip, a plate, or a blob according to Equations 8, 9, and 10 in Section 2.1.1; the results are shown in Table 3. Using Equations 1, 1, 1, and 1, and again defining two dimensions as similar if the ratio of the biggest to the smallest is within 4:1 (width:height:length ratios for the two parts are 1:1:2.51 for the block and 1:1:0.89 for the cylinder), then both the block and the cylinder are classified as blobs. Although their connection position and orientation is consistent with the hammer model, the hammer model requires that its handle be a stick. The unknown object cannot, therefore, be classified as a hammer.

6 Limitations

The domain of hand tools defines a simple, one-to-one mapping between an object's functional primitives and relations and their corresponding shape primitives and relations. In the more general case, the mapping from shape primitives to functional primitives is many-to-one, and a much more elaborate reasoning strategy is required to support the inference of a functional primitive from a collection of interacting shape primitives. Nevertheless, we strongly believe that such a reasoning mechanism must operate at the level of an object's coarse volumetric parts.

The object representation described in this paper is appropriate for objects composed of simple volumetric parts. Furthermore, we support only functionality that is defined in terms of an object's shape. Functions that are based on color, texture, or more importantly, motion, are not currently supported, although in current work we are enhancing our representation to include motions of an object's parts.

7 Conclusions

We have presented an approach to function-based object recognition that reasons about the functionality of an object's parts. Previous approaches have relied on more global object features, often ignoring the problem of object segmentation and thereby restricting themselves to range maps of unoccluded

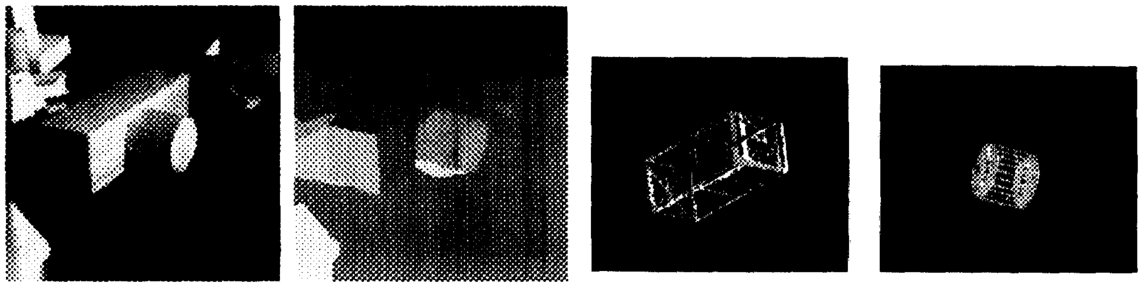

Figure 4: Object Recovery: original image, segmented region image, recovered block, and recovered cylinder.

scenes. We extend the popular "recognition by parts" shape recognition framework to support "recognition by functional parts", by combining a set of functional primitives and their relations with a set of abstract volumetric shape primitives and their relations. We show how these shape primitives and relations can easily be recovered from superquadric ellipsoids which, in turn, can be recovered from either range or intensity images of occluded scenes. Furthermore, the proposed framework supports both unexpected (bottom-up) and expected (top-down) object recognition.

References

[1] A. Barr. Superquadrics and angle-preserving transformations. *IEEE Computer Graphics and Applications*, 1:11–23, 1981.

[2] I. Biederman. Human image understanding: Recent research and a theory. *Computer Vision, Graphics, and Image Processing*, 32:29–73, 1985.

[3] S. Dickinson. Part-based modeling and qualitative recognition. In A. Jain and P. Flynn, editors, *Three-Dimensional Object Recognition Systems*, Advances in Image Communication and Machine Vision Series. Elsevier Science Publishers, Amsterdam, 1993.

[4] S. Dickinson and D. Metaxas. Using qualitative shape to constrain deformable model fitting. In *Proceedings, Sensor Fusion V, SPIE OE/Technology '92*, Boston, MA, November 1992.

[5] S. Dickinson, A. Pentland, and A. Rosenfeld. From volumes to views: An approach to 3-D object recognition. *CVGIP: Image Understanding*, 55(2):130–154, 1992.

[6] S. Dickinson, A. Pentland, and A. Rosenfeld. 3-D shape recovery using distributed aspect matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):174–198, 1992.

[7] D. Metaxas and S. Dickinson. Integration of quantitative and qualitative techniques for deformable model fitting from orthographic, perspective, and stereo projections. In *Proceedings, Fourth International Conference on Computer Vision (ICCV)*, Berlin, May 1993.

[8] P. Mulgaonkar, L. Shapiro, and R. Haralick. Matching "sticks, plates and blobs" objects using geometric and relational constraints. *Image and Vision Computing*, 2(2):85–98, 1984.

[9] A. Pentland. Perceptual organization and the representation of natural form. *Artificial Intelligence*, 28:293–331, 1986.

[10] L. Stark and K. Bowyer. Achieving generalized object recognition through reasoning about association of function to structure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):1097–1104, 1991.

[11] L. Stark and K. Bowyer. Generic recognition through qualitative reasoning about 3-D shape and object function. In *Proceedings, IEEE Conference on Computer Vision and Pattern Recognition*, pages 251–256, Maui, HI, 1991.

[12] L. Stark and K. Bowyer. Indexing function-based categories for generic object recognition. In *Computer Vision and Pattern Recognition (CVPR '92)*, pages 795–797, Champaign, IL, June 1992.

[13] L. Stark and K. Bowyer. Indexing function-based categories for generic object recognition. *CVGIP: Image Understanding*, to appear.

[14] L. Stark, L. Hall, and K. Bowyer. An investigation of methods of combining functional evidence for 3-D object recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, to appear.

[15] L. Stark, A. Hoover, D. Goldgof, and K. Bowyer. Function based recognition from incomplete knowledge of shape. In P. Kahn, Y. Aloimonos, and D. Weinshall, editors, *IEEE Workshop on Qualitative Vision*, pages 11–22, June 1993.

[16] M. Sutton, L. Stark, and K. Bowyer. Function-based generic recognition for multiple object categories. In A. Jain and P. Flynn, editors, *Three-Dimensional Object Recognition Systems*, Advances in Image Communication and Machine Vision Series. Elsevier Science Publishers, Amsterdam, 1993.

[17] D. Terzopoulos and D. Metaxas. Dynamic 3D models with local and global deformations: Deformable superquadrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):703–714, 1991.