Object Classification by Functional Parts

Guy Froimovich, Ehud Rivlin, Ilan Shimshoni

Abstract—
Object classification needs to address not only the changes resulting from various viewpoints but also the different shapes that can be classified into the same category. We present a new framework and its implementation for generic object classification from raw range images, combining structural and functional concepts. The framework addresses low and mid level problems for the decomposition of range images into primitive shape parts, presenting concepts for the classification of shape parts, and calculation of part properties and relations. New concepts are described, addressing the different aspects of generic class descriptions by functional parts. A mapping of functionality (and functional parts) to the primitive shape parts is presented, introducing functional part recognizers. Our approach mainly supports a top-level recognition process in which classes are verified using a verification tree in which functional parts and their realization hypotheses are explored. An algorithm for an effective traversing of the verification tree is presented, in which probabilities of hypotheses and classes are estimated. An experimental system applying our classification concepts to several classes was implemented and tested on a database of real raw range images of objects.

I. Introduction

Most of the problems in computer vision involve an understanding of a scene described by visual information. One of the most fundamental and challenging tasks in scene understanding is the recognition of objects appearing in that scene.

Much work has addressed the problem of 3-D object recognition. Most of the work focused on the *identification* of objects, where one identifies in an image a very specific object, one of a set of well specified objects. Structural recognition approaches found in the literature (e.g. [8], [17], [16], [3]) present an identification model based on a decomposition of objects into shape parts, motivated by psychological Recognition-by-Components concepts [6]. Yet, these approaches are still confined to recognition problems in which the shape of the objects to be recognized is well determined.

Little work has focused on the *classification* problem, where the imaged object is to be categorized and matched to one of a set of *classes* of objects and not a specific known object. It is apparent that difficulties described for the identification of objects are equally relevant to classification. In addition to that, the input to a classifier consists of low-level visual information, whereas the categorization of objects involves very high-level reasoning and understanding of object purpose. This high-level reasoning is not related directly to shape: instances of the same class may often look very different from one another. The imaged objects are not actually known to the classifier and therefore any straightforward matching techniques of the input to a known database, which are feasible in identification, are not applicable. Therefore, one should obtain a set of high-level criteria and properties which are general enough yet distinct to describe a class of objects, and a means of extracting such properties from the input images.

The need for "true" generic models for representing classes of objects and of recognizing them, has given rise to *functional model* approaches [24], [9], [20], [22], [21], [11], [2], [19], [18], [23], [12].

The work presented here describes a generic classification scheme from raw range images of objects, combining structural and functional approaches: the functional concepts are employed to describe classes of objects generically, whereas the structural concepts are used to acquire a robust part representation of the input image.

Guy Froimovich and Ehud Rivlin are with the Dept. of Computer Science The Technion – Israel Inst. of Technology. Email: ehudr@cs.technion.ac.il Ilan Shimshoni is with the Dept. of Industrial Engineering The Technion – Israel Inst. of Technology. Email: ilans@ie.technion.ac.il

II. Generic classification by functional parts

Our approach involves a classification of an imaged object through recognition by functional parts, thus combining part-oriented approaches with functional approaches. Such a classification involves a representation of classes by their functional parts, and a classification scheme in which these functional parts are recognized in the input raw image through a verification process of the different known classes. The following sections will discuss our concepts for functional parts, shape parts, mapping of functional parts to shape, functional representation of classes, and verification of classes.

The nature of functional parts will be discussed in the next section.

A. Representation by functional parts

A.1 Relating function to functional parts

Many objects in the real-world and especially man-made objects have a purpose or a certain task they were designed to participate in. This purpose is sometimes termed ([1], [7]) *the primary function*. Since this primary function is often the basis for categorization of objects into basic-categories, it provides a natural and generic representation of classes (categories). For example: a chair is an object on which a person sits; a cup is an object from which a person drinks hot drinks. We note that the primary function involves a very high-level understanding of the task and scene in which the object is used. We therefore further analyze the primary function and decompose it into several lower-level functions. We thus define *derived functions* as low-level functions that realize the high-level primary function. We then relate these derived functions to functional parts. A *functional part* is therefore, an area in the ob-

ject responsible for a well-defined set of derived functions. These functional parts and the relations between them are to satisfy the derived functions. In this level these functional parts are related to shape only in the sense that functionality of each part is realized by configurations of shapes, yet, as already pointed out before, it may be realized in many *different* shape configurations. We demonstrate the above functional analysis on the chair example: The primary "sittability" function is decomposed into several derived functions:
The object should provide a surface for a person to sit on.
The sitting surface should be placed stably at a sitting height.
The object may optionally provide a support for the person's back to lean on.
The object should provide room (proper clear space) for the person's body.
Towards satisfying these derived functions we present the previously mentioned decomposition into functional parts: The *chair-seat* provides the sitting surface. The *back-support* provides the support for the back. The *support-to-ground* provides a stable support for the seat at the specified height. Required relations between the parts refer to connections between them, clearance they provide, stability that the support-to-ground provides for the seat, etc.

We demonstrate this partitioning into functional parts, as identified on a chair, in figure 1.

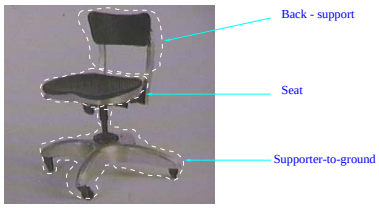

Fig. 1. Identifying functional parts in a chair instance

Each functional part has several functional properties. These properties may refer to simple properties such as orientation or dimensions of the functional part (analyzed in a functional context) or more high-level properties that are more specific to the task involved, such as: graspability for handles, stability for supporters etc. A functional description of a class is thus, represented hierarchicly by the primary function, its derived functions, and a list of its functional parts (mapped to derived functions), constraints on their properties and relations between the parts. This description is demonstrated in figure 2.

A.2 Generic functional parts

The previous section has introduced a framework in which each class is analyzed functionally and its functional parts identified. This view might lead one to think that the identified functional parts are specific to each class, and functional knowledge is not shared by different classes. However, a more generic approach can often be taken, making use of the same functional knowledge for several classes: having noted that several different classes may still share some low-level functions (derived functions), we claim that generic functional parts can be defined, accounting for functional parts of several classes. A *generic functional part* would thus be a functional part for which several properties and constraints are parametric. The functional description of a specific class would consist of parts that are mostly instantiations of these already defined generic parts where the class-specific constraints are set upon. As an example, we define the following two generic functional parts: The first is a "placeable" - being a flat part on which objects are to be placed (also requiring proper clearance above the placeable). We keep for the placeable several constraints as parameters: allowed dimension range, placed object size and allowed deformation. The second generic part is a "supporter-to-ground" - being a part designed to provide a proper support for other parts at a specific height above the ground. We now state that these generic parts can be instantiated as the functional parts of many classes. In particular, these generic parts are enough to account for most of the furniture classes: chairs (where the placeable accounts for both seat and back support), tables, beds, benches, sofas, etc.

B. Relating class description to shape

The previous section has described how classes of objects are described generically by functional parts. However, we recall that the input at hand that we wish to classify is a raw range image of an object and therefore involves a low-level shape information. The following section will describe how the functional descriptions are related to the low-level shape description.

B.1 Shape representation

We start by a definition of a more high-level shape representation to be related to the functional description. Our choice of high-level shape representation is a decomposition of the imaged object into a collection of primitive shape parts, motivated by the RBC concepts ([6]). These primitive parts would provide "building blocks" for our functional parts (in other words: each functional part would be realized by several possible configurations of primitive parts).

Our proposed primitive parts abstraction is a classification of primitive shape parts into 3 basic classes: sticks, plates and blobs (as can be found in [18]). A stick is a part in which one dimension is considerably larger than the others. A plate is a part in which two dimensions are considerably larger than the third. A blob is a part in which no dimension is considerably different than the other). In order to account for the variety of shapes in even simple classes such as mugs, we also allow bending deformations of these basic shape classes.

We address criteria for the structure of the part decomposition, geometrical properties and functional constraints for the primitive shape parts, by using several representations to describe each primitive part: a general assembly of surface regions, a vertex representation of a part (consisting

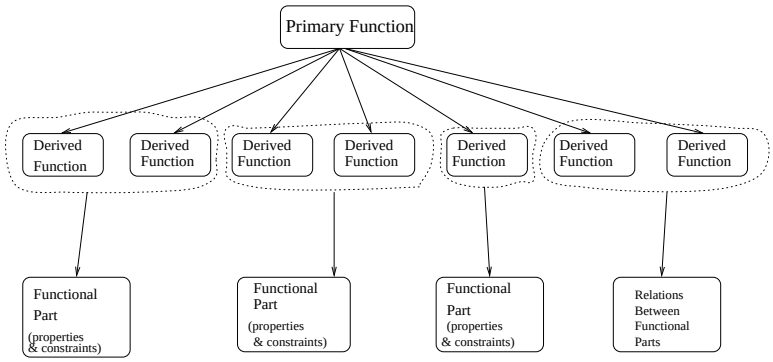

Fig. 2. Hierarchy of the functional representation

of several vertices on the deformed surface) and a fitting of the part to a known volumetric surface (quadric surfaces: cylinders, ellipsoids, cones or superquadrics) - if such a fitting exists. We define several properties of primitive parts that are of interest: classification of part, deformation level, convexity of part, vertices of parts, orientation, pose and dimensions. Relations of interest between such parts are: nature of connections between parts and relative orientation.

B.2 Relating functional parts to shape

By specifying for each functional part (either generic or class specific) how its functional criteria are realized by configurations of primitive shape parts, we can identify the existence of such functional parts in a decomposition of the image into the primitive shape parts.

We note that mapping between functional parts to shape parts is not a one-to-one mapping: a single functional part may be realized in several different shape configurations. Nevertheless, all realizations conform to the same functional properties defined for the functional part. For example: the "placeable" generic functional part defined above may be realized by a single plate or by a collection of plates or sticks placed side-by-side in a specific orientation; however, they both conform to the "placeability" functional properties and constraints.

We perform the mapping to shape by defining *functional recognizers* (see figure 3) for each of the functional parts (generic when possible). Each recognizer receives as input a set of primitive shape parts (being the part decomposition of the raw image) and an optional set of cues containing additional knowledge we have on the part. The output of such a recognizer is a set of hypotheses for realizing the part by configurations of the input shape parts. Each hypothesis is given a grade that specifies how well the hypothesis conforms to its functional requirements. The given cues are, in fact, a set of additional constraints set on the part that emerge from relations with other parts and from class-specific constraints. These constraints allow a more effective recognition and reduce the number of hypotheses found. By providing the supporter recognizer with additional cues such as what is the ground direction or what surface is to be supported, the recognition process becomes more robust, and many irrelevant hypotheses can be avoided. The actual recognition process will be described in detail in the sections to follow.

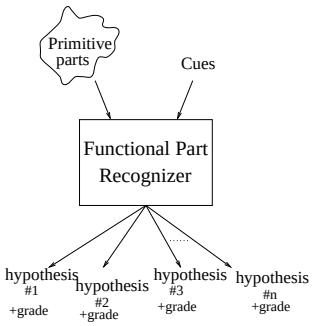

Fig. 3. Functional Part Recognizer

C. *Functional verification*

Having introduced the shape representation of the image on the one hand, and class functional descriptions on the other, as well as concepts for mapping between them, we now describe concepts for the high level process of verifying the conformance of the shape representation to the functional description of classes. This verification process is the basis for our recognition scheme.

We view the top level description of each class as a simple graph (figure 4) in which each node represents a functional part of the class, and the edges represent relations between functional parts. Each functional part is mapped to a "functional part recognizer" The additional knowledge provided as cues to the recognizers of functional parts is gathered throughout the verification process where hypotheses for other parts have already been found. This additional knowledge makes relations between parts explicit. Assuming we have class descriptions for several classes, our classification is achieved by performing a verification of each class. This verification process is performed according to the following algorithm:

A functional part is chosen, and its realization hypotheses are found.

Each hypothesis is explored by picking another functional part and exploring its own possible realizations. We note that the process is tree-based, where the nodes of the

tree are either functional parts to be explored or realization hypotheses for functional parts.

After a path was found in the verification tree, in which valid hypotheses were found for *all* functional parts of the class, a "whole object test" is performed. This test accounts for the fact that there are relations that cannot be effectively expressed in terms of cues provided by hypotheses and used by other recognizers, and are more naturally expressed as tests performed when all hypotheses are found. An example of a verification tree is shown in figure 5.

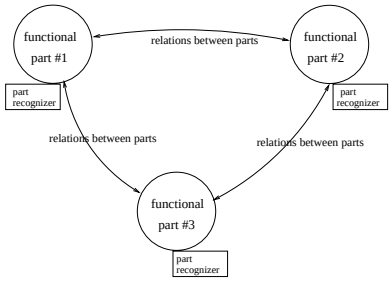

Fig. 4. Top level description of a class

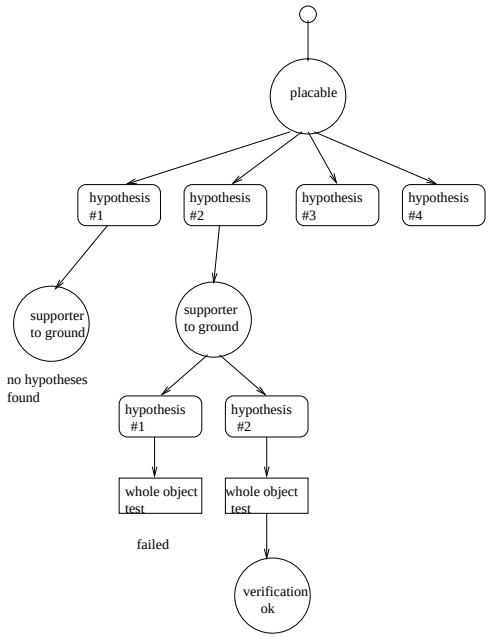

Fig. 5. Verification tree

III. Description of the system

The previous section has introduced several generic classification concepts. These concepts were related to several levels of the classification problem: shape descriptions of raw images by shape parts, description of classes by functionality: class-specific and generic functional parts, relation of shape to function, etc. We now present a system that applies these concepts. The system takes as input a raw image of an object and outputs a classification result. The proposed classification system is based on a high-level top-down process of class verification (as described before) in which it is tested if a decomposition of the input image into primitive shape parts conforms to any of the known class descriptions. The part decomposition of the image is acquired by a low level phase in which the image is segmented into regions and a mid level in which regions are collected and classified into primitive shape parts. The following sections will describe in detail the low, mid, and high-level phases of the system.

A. Low-level processing

The low-level phase of the system involves the initial processing of the input images. The input to this phase is a raw range image represented as a *point cloud* (a collection of 3-D points). The output of this phase is a segmentation of the point cloud into regions, where a *region* refers to a collection of image points having similar geometrical properties focusing on surface parameters.

Based on the concepts found in the literature [14], [4], [5], [10], [13], [15], several segmentation techniques were investigated throughout our research. The majority of these techniques were confined to the segmentation of polyhedral objects in which *regions* could be represented as planar surfaces. We chose to employ in our system a rough segmentation technique, as described below, for both curved and polyhedral objects. In the rough segmentation, we decompose the image into patches of size $n \times m$ image points (in the current implementation this size is adjusted according to typical object dimensions). We fit these patches to a surface, computing the surface parameters and a measure of fit. We then collect these patches into regions by a region growing process. We currently fit each patch to a planar surface (represented by: $P\hat{n} + d = 0$). The surface parameters are therefore $\hat{n}$ and d and are found by a simple least-squares fitting. In the region growing process we add patches to the current region by comparing patch parameters with the current seed, setting a relatively high tolerance for similarity of parameters, and therefore curved regions are also made possible. Boundaries between regions conform to local jump in d or $\hat{n}$ or to patches with bad grade of fit. We also tried to fit patches to quadrics, but we found that this fitting could not provide robust and stable results for small patches. A quadric fitting was therefore left to a higher-level phase as described in the following section.

B. Mid-level processing

At this phase, the initial segmentation of the image into regions is further processed to produce a decomposition of the image into primitive parts. Towards a part-based decomposition, several steps are taken, as described next.

We start with region refinement, then we build a region connectivity graph in which nodes are mapped to the regions (that were found in the low level) and edges are mapped to relations between regions. The graph is built by classifying the region-to-region connection to one of the following classes: 1) disconnected, when the regions that have no common boundary; 2) occlusion, when the two neighboring regions (having a common boundary) that are not connected and therefore, one occludes the other (i.e. there

is a jump in depth between the two regions); 3) connection - a relation that was not classified as *disconnection* or *occlusion* is considered *connection*. We classify the *connection* relation into 3 sub-classes: smooth, concave and convex. The classification into these sub-classes is performed using the mean jump in the normals along the boundary between regions.

Next we partition into primitive parts. We represent each part as a collection of regions. We traverse the connectivity graph and form collections of regions that are connected smoothly or convexly to each other. Each such collection (being a sub-graph of the connectivity graph) is mapped to a single primitive part.

Having found an initial partitioning into parts, we turn to their classification. In the current implemented system, we classify a part into *a stick* (with allowable deformation) or a *plate* (allowing deformations on the plate). We fit the deformed plates to a quadric to achieve a more compact and high-level representation. We then represent the deformed plate, in addition to the vertices representation, by a quadric and a classification of the quadric to a known quadric class (cylinder/ellipsoid) (using the eigenvalues of the matrix representing the quadric), if such exists.

At this level of the problem we choose to regard a blob not as a primitive shape but as a collection of deformed plates (to be investigated in the higher-levels).

As a final phase in the part decomposition, we perform merging of parts. This involves a high-level test to check if two classified parts can be merged together to form one part. This may account for over-segmentation in the low-level phase and occlusion of parts. In the currently implemented system, the merging involves only the merging of small close sticks to one stick.

After the final decomposition has been set, we find the relations between parts. We classify the connection relation (disconnection, occlusion, smooth-connection, concave-connection, convex-connection) in a similar way to the procedure performed on the regions.

C. High-level processing

As discussed in section II, in order to verify if a shape-part decomposition of an image conforms to a known class, one should acquire functional representations for known classes, using generic and class-specific functional parts. This section describes our application of these concepts to several classes, and our implementation of the high-level verification phase.

C.1 Generic functional parts

Here we present our definition and application of several generic parts. These parts are then used to represent the functional parts of many classes (described in the next section). We describe each functional part by a textual description of its functionality, properties of the part, the set of constraints on it, a description of how the part is realized in terms of shape, and how it is recognized in an image.

Placeable: a functional part providing a surface on which other objects are placed.

- Parametric constraints for the placeable part:
  1. Allowed range for the placeable dimensions
  2. Maximum level of deformation
  3. Direction of placing surface
  4. Clearance volume (specification of volume, relative to the placeable, that should be clear of any obstacles).
- Main properties of a placeable:

Placeable surface (which in itself includes dimensions of surface, center, orientation etc.)

- Shape realization of placeables:

In our current implementation, a placeable may be realized in two ways:

1. By a single plate (deformed or non-deformed)
2. By a collection of close sticks or plates, placed one next to the other in the same orientation, forming a single non-continuous surface.

A schematic view of placeable realizations is shown in figure 6.

Recognition of placeables is performed by identifying such plates or stick/plates collection in the mid-level representation of the image, and testing if they conform to the parametric constraints (clearance, surface direction, dimensions, deformation level).

Fig. 6. Possible placeable realizations

Supporter to ground: a functional part that provides a stable support to the ground for other parts.

- Parametric constraints:
  1. Shape part to be supported
  2. Direction of ground
  3. Allowed range for supporter height
- Main properties:
  1. Height(as a function of part to support)
  2. Supports_Part(as a function of part to support)
- Shape Realization

In our current implementation, a supporter-to-ground may be realized in several ways:

– Collection of legs (3 or 4 legs)

– 1 leg + base support, where "base-support" in itself may be realized by a plate connected to the leg in its center, or n sticks connected to the leg.

– a blob.

A schematic view of supporter-to-ground realizations is shown in figure 7.

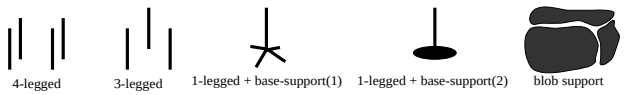

Fig. 7. Several possible supporter-to-ground realizations

We also point out that a "leg" as used in the definitions above, is a stick or a collection of several sticks joined together.

Recognition process of supporters-to-ground involves identification of sticks in the primitive parts collection.

Then, possible configurations of sticks forming 3/4/1-legged hypotheses are tested with the parametric constraints of height and stability. The stability constraint is checked by performing two tests. The first test finds the polygon defined by the connection points of the supporter to the part, and then checks if the center of the supported part is within this polygon (in the case of 1-legged supporter - this test actually checks if the leg is connected to the center of the part). The second test finds the projection of the part on the ground, and then checks that a large enough portion of that projection lies within the polygon defined by the supporter ground tips.

By defining a leg as a collection of sticks, we overcome over-segmentation problems in which a single stick in the image was over-segmented and decomposed into several sticks. In such a case, the system will also consider a hypothesis in which these sticks form a single leg, thus handling correctly the recognition of the over-segmented supporter. Another common problem in recognition of supporters is occlusion. In many views of objects having supporters-to-ground, the legs are partly occluded by the supported part. We address this occlusion problem by testing hypotheses in which the actual connection of each leg is not explicitly seen in the image but is inferred by the seen part of the leg.

In a similar way we define other functional parts like handle, container, etc...

C.2 Functional description of classes

In the following section we will apply our representation concepts to the functional representation of several classes that were tested by our system. We describe the decomposition of these classes into functional parts (each part being an instance of one of the generic functional parts described before), and then present functional criteria for these parts and for relations between them.

Chairs

• Functional parts (as shown in figure 8): seat, seat-ground-support, back-support (optional)

1. seat ::= placeable(sitting size, seat max deformation,[ground_z])
2. back-support := placeable(back size, back max deformation, $\perp$ seat-direction)
3. seat-ground-support ::= supporter-to-ground([seat], [ground_z], sitting_height)

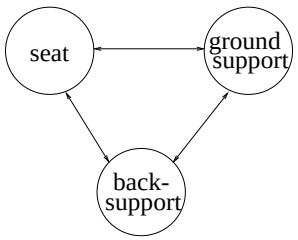

Fig. 8. Chair functional parts

• Relational criteria:

1. seat is perpendicular to back-support
2. ground-support provides a stable support for seat
3. seat is connected to back-support
4. ground-support is connected to seat
5. combination of: seat + back-support + ground-support provides seating and room-for-legs clearance.

A table is similar to a chair but with different dimensions, without a back-support and the placeable must be non-deformed.

Similarly we define a mug as having a mug-container connected to a mug-handle which can be grasped by a person.

In a similar way we defined more classes using our defined generic functional parts. Those include: glass, bowl, pot, bed, shelf, etc...

Each class was denoted a "class-verifier" - a module responsible for verifying if the input imaged object is a valid instance of the class. The class-verifier thus includes the functional parts and their constraints, a recognizer for each, probability and work weights, a "whole-object-test" and a common knowledge repository. The high-level implementation of the classification of an imaged object applied the class-verifiers to the data in probabilistic order.

IV. Experimental Results

We have tested the implemented system, described in section III, on an image database consisting of various range images of *real* objects, scanned by a Cyberware laser-based range scanner. Several kinds of objects have been scanned and added to our input database: objects that are instances of classes *known* to the system, objects that are functionally *non-valid* instances of the known classes (i.e: a chair that is unstable, a mug that is not graspable, etc.), and finally: objects that are instances of classes that are unknown to the system (boxes, hand-tools etc.). Throughout the tests, we have verified that the low and mid-level phases, as well as the high-level classification, were performed correctly. Examples for intensity images of some tested objects are shown in Figure 9. Summary of tests performed on real objects containing our supported functional parts, is presented in Table 1. The table shows that almost all the tested objects were segmented and then classified correctly (valid and non-valid instances of known classes and instances of unknown classes). Only a few (15 objects) were segmented but could not be classified because they consisted of realizations of functional parts that were not yet implemented.

We now present several examples for these tests. Figure 10 demonstrates the classification of a standard valid chair having all three functional parts (a seat, a back-support and a supporter-to-ground). The mid-level phase has segmented the image correctly into 2 plates and a collection of 4 sticks + a few dummy sticks (due to over-segmentation). In the high-level phase the first part chosen to be elaborated was the chair seat. Two hypotheses for seat placeables were found. The first seat hypothesis was further explored: the next part chosen to be elaborated was chair back support. One hypothesis was found. The next chosen part was chair support to ground. Several hypotheses were found (due to several possible combinations

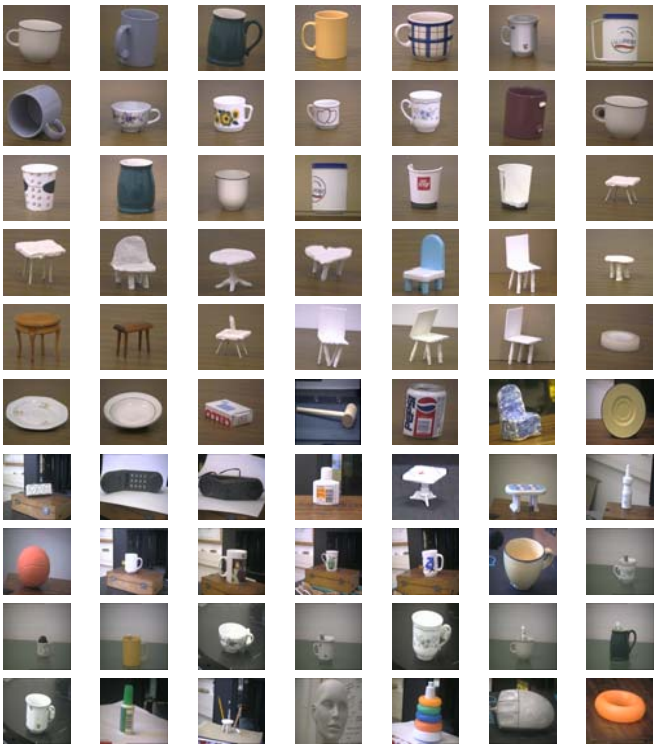

Fig. 9. Intensity images of some tested objects

Class	Tested objects
valid chairs	15
valid mugs	12
valid glasses	7
valid tables	7
valid plates	3
non-valid chairs	8
non-valid mugs	17
non-valid glasses	3
non-valid tables	5
non-class	23
segmented, not-classified	15
total tested	**115**

TABLE I
SUMMARY OF TESTS

of legs). The correct combination passed the "whole object test" and the image was verified successfully as a chair.

Figure 11 demonstrates the classification of a stool - a 3-legged chair without a back-support. We also note that the seat, unlike the standard chair is heart-shaped and deformed plate.

The Classification of a a blob supporter-to-ground in a couch is demonstrated in Figure 12. The mid-level segmented the image into several plates conforming to the chair itself and a collection of plates and sticks (deformed and non-deformed) due to over-segmentation and clutter. The high-level phase chose to start with the seat (due to utility considerations) and found several valid hypotheses, then found a valid blob supporter-to-ground only for one

intensity range segmented

Fig. 10. A standard valid chair with a back-support

intensity range segmented

Fig. 11. A classification of a valid 3-legged chair

of them. The image was verified correctly as a valid chair.

intensity range segmented

Fig. 12. A classification of a couch having a blob supporter-to-ground

Figure 13 demonstrates the correct classification of a valid 3-legged chair in a neighborhood consisting of other "non-class" objects (pencils and a piece of chalk). Clutter due to the other objects did not affect the functional validity of the "real" chair and therefore the image was verified correctly as a valid chair.

intensity range segmented

Fig. 13. A correct classification of a valid chair in a cluttered environment

The classification of a valid non-cylindrical mug with an over-segmented decomposition into parts is demonstrated in Figure 14: The mid-level phase has segmented the image correctly into 2 deformed plates (not cylindrical but still convex and concave with an aperture contour). An over-segmented collection of sticks conforming to the handle were also found. The high-level phase found a valid container (in terms of volume and clearance) and then found several hypotheses for a handle (conforming to different possible combinations of sticks from the over-segmented handle).

The classification of a non-valid chair is demonstrated on a chair missing one of its legs (Figure 15). The high-

intensity range segmented

Fig. 14. A classification of a valid non-cylindrical and over-segmented mug

level process found a valid seat and a back support. Yet, among supporter-to-ground hypotheses, none was found to provide a stable support for the seat.

intensity range segmented

Fig. 15. A classification of a non-valid chair, missing one of its legs

In Figure 16 we demonstrate the classification of a mug that is not valid due to a non-graspable handle. The high-level phase found a valid container and then found several hypotheses for handles, yet, all were ruled out. The "real" handle had correct grasping dimensions but did not have grasping clearance. Therefore, the verification of *mug* failed.

intensity range segmented

Fig. 16. A classification of a coffee mug having an obstacle in its handle

In the above examples, we have demonstrated how over-segmented parts are handled in the high-levels (the legs of a chair supporter-to-ground or the handles of mugs are found even if they are constructed of several connected deformed sticks). We also note that in all the chair examples, the legs were partly occluded. Nevertheless, the high-level phase has considered hypotheses where the legs "could" be connected to the seat and provide a stable support.

V. Conclusions

In this paper we have addressed the problem of object classification from raw range images. A new framework has been presented addressing the low and high-levels of the problem by combining structural and functional approaches. An experimental system applying our framework has been implemented and tested on over a hundred range images of real objects. We have presented in the paper several results of these tests, demonstrating the part decomposition of different shaped objects and the classification of instances of several classes, thus proving the feasibility of our approach and its robustness to over-segmentation and cluttered scenes.

Future research includes the definition of some additional generic functional parts, thus enabling the classification of many more classes, and the generalization of the shape-to-function mapping by implementing more realizations to the existing functional parts. The introduction of new classes is to be generalized by adding learning algorithms to the classification process. We expect to learn various realizations of existing generic functional parts, as well as high-level representation of classes by these parts.

References

[1] G. Adorni, M. DiManzo, F. Giunchiglia, and L. Massone. A conceptual approach to artificial vision. In *4th conference on Robot Vision and Sensory Controls*, pages 231–260, 1984.

[2] R. Bajcsy and F. Solina. Three dimensional object representation revisited. In *Proc. Int. Conf. Comp. Vision*, pages 231–240, 1987.

[3] R. Bergevin and M.D. Levine. Generic object recognition: Building and matching coarse descriptions from line drawing. *PAMI*, 15(1):19–36, January 1993.

[4] P.J. Besl and R.C. Jain. Segmentation through variable-order surface fitting. *PAMI*, 10(2):167–192, March 1988.

[5] S.M. Bhandarkar and A. Siebert. INTEGRA: An integrated system for range image understanding. *PRAI*, 6(5):913–953, 1992.

[6] I. Biederman. Recognition by components: A theory of human image understanding. *Psychological Review*, 94(2):115–147, 1987.

[7] L. Bogoni and R. Bajcsy. Interactive recognition and representation of functionality. *Comp. Vis. Im. Understanding*, 62(2):194–214, September 1995.

[8] R.A. Brooks. Symbolic reasoning among 3-D models and 2-D images. *Artificial Intelligence*, 17:285–348, 1981.

[9] M. DiManzo, E. Trucco, F. Giunchiglia, and F. Ricci. Fur: Understanding functional reasoning. *International Journal of Intelligent Systems*, 4:431–457, 1989.

[10] S. Ghosal and R. Mehrotra. Segmentation of range images: An orthogonal moment-based integrated approach. *RA*, 9(4):385–399, 1993.

[11] K. Green, D. Eggert, L. Stark, and K. Bowyer. Generic recognition of articulated objects through reasoning about potential function. *CVIU*, 62(2):177–193, September 1995.

[12] J. Hodges. Functional and physical object characteristics and object recognition in improvisation. *Comp. Vis. Im. Understanding*, 62(2):147–163, September 1995.

[13] R. Hoffman and A.K. Jain. Segmentation and classification of range images. *PAMI*, 9(5):608–620, September 1987.

[14] A. Hoover, G. Jean-Baptiste, X.Y. Jiang, P.J. Flynn, H. Bunke, D.B. Goldgof, K. Bowyer, D.W. Eggert, A. Fitzgibbon, and R.B. Fisher. An experimental comparison of range image segmentation algorithms. *PAMI*, 18(7):673–689, July 1996.

[15] X.Y. Jiang and H. Bunke. Fast segmentation of range images by scan line grouping. *Machine Vision and Applications*, 7(2):115–122, 1994.

[16] D.J. Kriegman, T.O. Binford, and T. Sumanaweera. Generic models for robot navigation. In *DARPA88*, pages 453–460, 1988.

[17] A.P. Pentland. Recognition by parts. In *ICCV87*, pages 612–620, 1987.

[18] E. Rivlin, S.J. Dickinson, and A. Rosenfeld. Object recognition by functional parts. In *Image Understanding Workshop*, pages II:1531–1539, 1994.

[19] E. Rosch. Cognition and categorization. In *ed. E. Rosch and B. Lloyd, Erlbaum, Hillsdale, NJ*, 1978.

[20] L. Stark and K.W. Bowyer. Function-based generic recognition for multiple object categories. *Comp. Vis. Graph. Im. Proc.*, 59(1):1–21, January 1994.

[21] L. Stark, A.W. Hoover, D.B. Goldgof, and K.W. Bowyer. Function-based recognition from incomplete knowledge of shape. In *WQV93*, pages 11–22, 1993.

[22] M.A. Sutton, L. Stark, and K.W. Bowyer. Gruff-3: Generalizing the domain of a function-based recognition system. *Pattern Recognition*, 27(12):1743–1766, December 1994.

[23] L.M. Vaina and M.C. Jaulent. Object structure and action requirements: A compatibility model for functional recognition. *International Journal of Intelligent Systems*, 6:313–336, 1991.

[24] P.H. Winston, T.O. Binford, B. Katz, and M. Lowry. Learning physical descriptions from functional descriptions. In *AAAI-83*, pages 433–439, 1983.