Available online at www.sciencedirect.com
ScienceDirect
Computer Vision and Image Understanding 105 (2007) 200–217
Computer Vision and Image Understanding
www.elsevier.com/locate/cviu

Efficient search and verification for function based classification from real range images

Guy Froimovich Ehud Rivlin [a], Ilan Shimshoni [b], Octavian Soldea [a,*]

[a] Department of Computer Science, The Technion, Israel Institute of Technology, Haifa, Israel
[b] Department of Management Information Systems, University of Haifa, Haifa, Israel

Received 10 January 2006; accepted 23 October 2006
Available online 12 December 2006

Abstract

In this work we propose a probabilistic model for generic object classification from raw range images. Our approach supports a validation process in which classes are verified using a functional class graph in which functional parts and their realization hypotheses are explored. The validation tree is efficiently searched. Some functional requirements are validated in a final procedure for more efficient separation of objects from non-objects. The search employs a knowledge repository mechanism that monotonically adds knowledge during the search and speeds up the classification process. Finally, we describe our implementation and present results of experiments on a database that comprises about 150 real raw range images of object instances from 10 classes.

Keywords: Function based reasoning; Recognition; Classification; Computer vision; Raw range images; 3D segmentation

1. Introduction

The object recognition field in computer vision made its debut in the area of object identification and has moved, in recent years, towards classification. Object identification seeks to recognize a certain object with a particular identity. The classification's goal is to target shapes that can, after analysis, be grouped into the same category. This is in contrast to object identification.

To date, most of the work on recognition has focused on the identification of objects, that is, recognizing a specific object in an image, one of a set of well-specified objects. Structural recognition approaches (e.g., [5,12,32,38]) use an identification model that is based on a decomposition of objects into shape parts. These approaches are motivated by psychological recognition-by-components concepts [7]. Little work has focused on the classification problem, where the imaged object is to be categorized as one of a set of classes of objects rather than a specific known object. In this context, functional based classification analyzes the function an object can fulfill. This function is in fact a classification criterion.

The difficulties described for the identification of objects are equally relevant to classification. In fact, classification of objects requires an even higher level of reasoning – and a better understanding of the object's purpose – than does identification. This high-level reasoning is not directly related to shape: instances of the same class often bear little resemblance to each other. Because the imaged objects are not actually known to the classifier, straightforward techniques for matching the input to a known database – feasible in identification – are not applicable here. Thus, a set of high-level criteria is required, along with properties that are distinct and general enough to describe a class of objects. A means for extracting these properties from the input images is also necessary.

The possible advantages of functional approaches to generic classification were recognized early on in works such as [16,55]. Although several systems for object classification (see [4,16,48,51]) were devised on the basis of these

* Corresponding author.
E-mail addresses: ehudr@cs.technion.ac.il (G.F.E. Rivlin), ishimshoni@mis.haifa.ac.il (I. Shimshoni), octavian@cs.technion.ac.il (O. Soldea).

1077-3142/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.cviu.2006.10.003

approaches, little experimental work has been done to test them. Only preliminary attempts have been made towards functional classification using raw images [49] or stereo image pairs [52].

However, the existing models for functional classification are still not truly generic, and the relationship of high-level concepts to low-level images is still not robust. In this context, most of the experiments described in the literature involve range images of non-cluttered scenes as well as computer generated polygonal models. Moreover, there is plenty of unexplored and unexploited amount of knowledge in modeling human cognitive processing, which should address the connections among the high- and low-level images understanding, see [1].

A promising new trend in the functional approach to object recognition is the use of image sequences, in conjunction with the analysis of static 3D models, to understand human interactions with the environment. The authors in [40] proposed a scheme for analyzing sequences of images in which a person interacts with a room full of chairs. While images are segmented by tracking the human operator, the elements in the room are labelled by measuring the degree at which the human interacts with them. In this context, in [3], the authors provide an overview of research on the use of physical tools. Moreover, they claim that intelligence can be evaluated by analyzing the activity of agents as tool users.

We describe a general scheme for classifying objects from raw range images. Our scheme combines functional and structural approaches to obtain a generic description of object classes. In the low-level phase, we process the raw input image to obtain a segmentation into primitive shape parts: sticks, plates, and blobs (deformed and non-deformed), as well as the relationships between them. In the high-level phase, we employ functional part recognizers to compute mappings from primitive parts to their functionalities. These recognizers receive as input primitive parts and constraints on the functional part to be recognized, and output a collection of hypotheses that measure how this part may be realized. We employ graph-driven verification of classes for the actual classification, and verify if the input image conforms to the generic description of each of the known classes. Each class verification involves a validation of hypotheses for the functional parts of the class.

We address the exponential complexity of the classification process by introducing a probability-based efficient search algorithm. This search algorithm employs a generalization of the indexing measure mechanism proposed in GRUFF (see [47] and [46]) Chapter 6, where indexing is used (only once) to decide which classes are to be considered for verification and in which order. In addition, after hypotheses for all the functional parts have been proposed, a whole-object-test using functional based reasoning is performed. This type of test validates the existence of relatively complex relationships among the functional parts of the object.

We implemented our algorithm and tested it on a database of 150 range images of real 3D objects. The system was able to recognize all the objects from the classes it was designed to recognize. The importance of the probability search and the verification stage are demonstrated experimentally. The probabilistic search increases the speed at which the objects are recognized. The verification stage generates accurate classification of the objects and their functional parts.

This paper is organized as follows. In Section 2, we survey the existing literature. In Section 3, we review the low-level stages of segmentation and decomposition into parts, the high-level method of class representation by generic functional parts, and the classification process. In Section 4, we give the details of our implementation and in Section 5 we describe our experimental results including time measurements for classification as well as ratios in speed improvement of our scheme when the probabilistic mechanism is turned on and off. We present our conclusions in Section 6.

2. Related work

Most of the work addressing the 3D recognition problem uses a model-based approach, where the input is matched to several models of objects. Several researchers use a geometric model in which the input is directly matched to a model of low-level geometric features (see [17,19,18,25,27,54]). These works, however, are confined to specific object shapes. Moreover, the geometric approach is not easily applicable for more generic recognition, where the objects to be recognized are not known to the system and therefore no specific geometric models can exist.

The *structural model* approaches propose a higher-level, cognitive-based model in which recognition is achieved by matching *parts* of the input objects and connections between them to models. The cognitive basis was established by Marr and Nishihara [35] which was extended by Biederman [7]. He employs the RBC (recognition-by-components) concept which defines an object category in human recognition as a particular set of qualitatively described primitive 3D shapes called geons and the qualitative relationships between them. Recognition is based on an indexing process that relies on the structural composition of the object. Examples for such an approach can be found in [5,12–14,32,38,39]. Although these approaches are more generic and robust than the geometric model based schemes, they are still confined to recognition problems in which the shape of the objects to be recognized is known. Note that instances of even simple classes of objects differ from one another greatly in terms of shape and structure.

In [24], the authors built a recognition system based on a verifying constraints step followed by a validation one. In the first step, the constraints are verified employing a finite set of production generation rules grouped in an attribute

grammar. In the second step, the validation uses a probabilistic model. This recognition system uses rectangles as primitives and was tested on four 2D images of interior rooms that contained objects, that are decomposable into rectangles. This is an example of perceptual grouping being used for recognition. In 3D the equivalent perceptual grouping step groups clouds of 3D points into primitives and finds the relationships between them (as is done in the initial step of our algorithm).

The need for "truly" generic models for representing classes of objects in classification processes has given rise to functional model approaches. Winston et al. [55] propose a theory explaining how physical models are learned and identified using functional definitions, physical examples, and precedents. The authors in [16] present the FUR (FUnctional Reasoning) project, a functional reasoning and shape–function integration system, in which several functions (such as support, grasp, enter, and hang) are presented, and the use of functional expert concepts for identification of functional primitives is discussed.

An impressive number of good results in the function-based classification field were demonstrated with the GRUFF, OMLET, and OPUS systems [22,48,49,51,56]. GRUFF, which employs generic representations using form and function, was extensively used on more than two hundred synthetic models of five categories [51]. OMLET, which is a version of GRUFF able to learn classes, was also extensively tested on several hundreds of synthetic models of two categories: cups and chairs [56]. GRUFF was also tested on a few raw images produced from simulators and stereo capturing systems (see [52]). OPUS, which is a version of GRUFF that deals with the unseen space, was tested on range images that included mock chairs built from boxes and several cups (see [46,50]). However, quantitative experiments on real range imagery of real objects are still lacking in the literature. An interesting refinement of the function-based recognition approach can be found in [22], where the authors describe a system for dealing with articulated objects whose function depends on how their parts are joined.

In [34], a recognition system of articulated objects is proposed. This system employs a learning stage based on accumulative Hopfield matching.

The authors in [4] propose a modeling system for generic objects. This system, which uses a prototype made up of superquadric parts, is based on the psychological notions of categorization as suggested in [45]. The authors in [44] describe a framework for recognizing objects by means of functional parts. This framework combines functional primitives, volumetric shape primitives, and the relationships between them. First, the input images are segmented into parts, which are further fitted to deformable superquadrics. Each part is classified into one of four types: strips, sticks, blobs, and plates. The recognition process is based on finding functional features, relationships, and attachments between pairs of parts (as well as other shape features).

In [53], a model for recognition functionalities that combines representations of shapes and object categories with action requirements is proposed. In this context, higher-level functional concepts are presented in [26], where the authors describe the problem of improvisation. The authors study the relation- ship between physical properties of objects, their functional and behavioral representation, and their use in problem solving. More recently, the authors in [31], propose a generic model inference scheme that is based on examples. That the model is generic is ensured by its high-level representation of the input object, which is segmented and partitioned into adjaceny-related regions. The regions are groups of 2D information elements that represent the lowest common abstraction detectable in the sequences of input images. However, the usefulness of such a generic system in direct recognition tasks is not clearly demonstrated and is left for future research. Moreover, the system is based on the assumption that the viewing direction of different input examples is consistent, and the problem of merging several views is left for future research as well.

In [11], a physics based application for automatic understanding of gear mechanisms is described. Functional inferences are employed for reverse engineering, making it possible to cope with occlusion and partial information.

A good overview on function based classification methods can be found in [6]. The authors feel that commonsense based reasoning is a particular, specialized, very high-level type of functional reasoning [15,36,37].

We conclude that the functional approaches, though more generic than the structural ones, still fail to present truly generic models on the one hand and a robust relationship between high-level concepts and low-level representation on the other. The few experimental systems in the literature were usually designed to perform classification of high-level models of objects and not of raw images of them.

3. Generic classification by functional parts

Classification approaches have to cope with at least the following three fundamental problems: determining how the different classes of objects can be properly described, relating class descriptions to input images of objects, and developing a recognition process in which the matching of the imaged object to the known classes is verified efficiently and effectively. Because categorization of objects (and especially man-made objects) into basic categories is usually based on a well-defined purpose, a high-level functional representation of classes is likely to be useful for representing all instances of a class generically. In the following sections, we present our concepts of generic functional parts and shape parts. We also describe our mapping of functional parts to shapes, and discuss how functional recognizers are employed in our approach, as well as how the conformance of a represented image to a functional class is verified.

3.1. Representation by functional parts

If we know the function of the different parts of a shape, we can use this knowledge to arrive at a generic, abstract interpretation of the environment. Many objects in the real world, especially the man-made ones, have a function they are designed to fulfill. This function is sometimes termed *the primary function* [8,2]. The primary function, which is often the basis for categorization of objects into basic categories, provides a natural and generic representation of classes.

Recognizing an object's primary function requires a very high-level understanding of the task and the scene in which the object is used. Therefore, we decompose the primary function into several lower-level functions. The high-level primary function is realized by *derived functions*. Derived functions are lower-level functions that relate to functional parts. A *functional part* is, therefore, a region in the object responsible for a well-defined set of derived functions. These functional parts and the relationships between them are designed to realize the derived functions. For example, in a chair, the primary function is "sittability." We decompose the primary "sittability" function into several derived functions: the object should provide a surface for a person to sit on, the sitting surface should be stably placed at sitting height, the object may provide a support for the person's back, and the object should provide room (proper clearance space) for the person's body. The chair is decomposed into the previously mentioned functional parts according to how the parts realize these derived functions: 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. The functional parts are connected by means of certain required relationships: for instance, the distance between the parts of a chair provide clearance, while the support-to-ground provides stability. We demonstrate this decomposition into functional parts, as identified for a chair, in Fig. 1.

Each functional part has several functional properties. These properties may be simple properties such as orientation or dimension, or higher-level, task-specific properties such as graspability for handles or stability for supports.

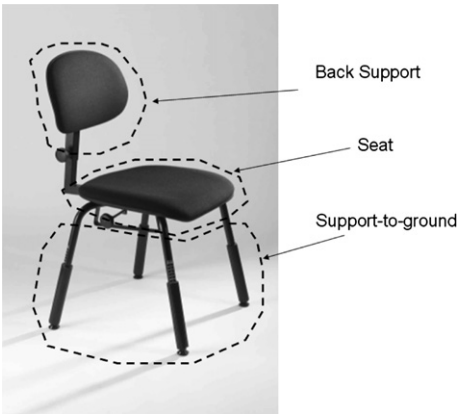

Fig. 1. Identifying functional parts in a chair.

A functional description of a class is, thus, represented hierarchically by the primary function, its derived functions, and a list containing the functional parts of that class, the constraints on their properties, and the relationships between the parts.

Different classes may share low-level (derived) functions. In this context, generic functional parts can be defined for several classes. A generic functional part can be seen as a functional part for which several properties and constraints are parametric. The functional description of a specific class consists of parts that are instantiations of generic parts and constraints. As an example, we define the following two generic functional parts: a "placeable" and a "support-to-ground." The "placeable" is a flat part on which objects are to be placed and, for its functionality, requires proper clearance from above. A "support-to-ground" is a part designed to provide proper support for other parts at a specific height above the ground. For the placeable, we impose several constraints as parameters: allowed dimension range, placed object size, and allowed deformation. Note that these generic parts can be instantiated as functional parts for a set of classes and are sufficient to account for most of the furniture classes: chairs (where the placeable accounts for both seat and back support), tables (where the placeable is allowed to be a plate, however, not a deformed one), beds, benches, sofas, etc.

3.2. Relating class description to shape

Recall that the input at hand consists of raw range images of objects which are first processed to obtain low-level shape information. The following section describes how the low-level shape information relates to functional descriptions.

Following the RBC (recognition-by-components) concepts defined in [7], we decompose the imaged object into a collection of primitive shape parts that will be used for high-level shape representation. Following [44], we classify primitive shape parts into three basic classes: sticks, plates, and blobs. 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 others. In order to account for the variety of shapes in even simple classes such as mugs, we also allow bending deformations of sticks and plates.

We address criteria for decomposition of the parts, geometric properties, and the functional constraints on primitive shape parts. We use several representations to describe each primitive part: a general assembly of surface regions, a vertex representation of a part (consisting of several vertices on the deformed surface), and the conformity of the part to a known volumetric surface (that is, quadric surfaces such as cylinders, ellipsoids, cones or superquadrics), if such a match exists. We define several interesting properties of primitive parts: their class, deformation level, convexity, vertices, orientation, pose, and dimensions. We

consider the following relationships between such parts to be of particular interest: the nature of the connections and the relative orientation.

If we bear in mind that there is no one-to-one mapping between functional parts and shape parts, it is easy to see that a single functional part may be realized in several different shape configurations. However, 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. All of these, however, conform to the "placeable" functional properties and constraints.

We employ *functional recognizers* (see Fig. 2) to map functional parts to shape parts. Each such recognizer receives as input a set of primitive shape parts (decomposed from the raw image) and an optional set of cues containing additional knowledge about the part. The output is a set of hypotheses that use configurations of the input shape parts to realize the functional parts. Each hypothesis is given a grade that specifies how well it conforms to its functional requirements. The given cues are, in fact, a set of additional constraints on the part that emerge from relationships with other parts and from class-specific constraints. These constraints allow a more effective classification and reduce the number of hypotheses. For example, a search for the functional "support-to-ground" of a chair, if carried out with no additional knowledge about the support, would output all possible collections of sticks (or blobs) that might provide support for any arbitrary part in an arbitrary direction. However, by providing the support recognizer with additional cues, such as ground direction or type of surface to be supported, the classification process can be made more robust and many irrelevant hypotheses avoided.

3.3. Design patterns

The design of many classes should account for the primary function of the class and its derived functionality. For example: a chair design is considered "valid" if it accounts for the functionality imposed by the "seatability" criterion. However, in many cases, among the possible realizations that conform to the derived functionality, several realizations (or patterns of realization) are preferred over others. The existence of such preferred patterns may account for very high-level functionality (such as convenience, reliability, etc.), in which a derivation to well-defined and computable criteria is very difficult, or even non-functional criteria (such as aesthetics, historical reasons, etc.).

We conclude that many functional parts, especially the ones that belong to man-made objects, are confined to several "popular" patterns for the realization of the derived functionality. We will term these patterns *design patterns*. For example: the common patterns for realizing "supporter-to-ground" of chairs are collections of three or four sticks or one stick with a base support. Collections of ten sticks, for example, may also be a valid "supporter-to-ground", however, they are not likely to be found in chairs. Making design patterns explicit, several advantages are gained: Very high-level functional and non-functional criteria can be taken into consideration in recognizing realization hypotheses for functional parts without having to derive these criteria explicitly.

We can define a more robust and straightforward function-to-shape mapping to these specific design patterns (also accounting more robustly to occlusion). Although these mappings would be less generic, they would still account for most common realizations of functional parts.

When several hypotheses are considered for the realization of a functional part, design patterns help set the correct priority order (which hypothesis is more likely to actually realize the functional part). In the support-to-ground example, a hypothesis that realizes four legs would be much preferred to one that realizes ten legs. Priority can be defined even for common design patterns: for instance, a four-leg pattern is more common than a three-leg pattern.

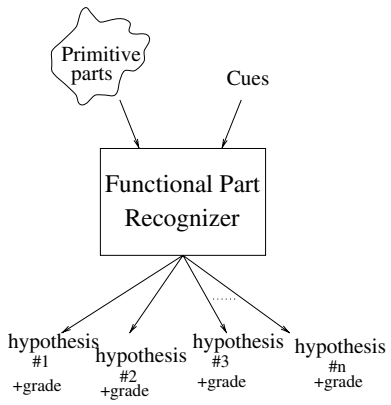

Fig. 2. A functional part recognizer.

3.4. Functional verification

Having introduced the shape representation of the image, the functional class descriptions, and the mapping of functional parts to shapes, we now describe the high-level process of verifying that the represented image conforms to the functional description of the class. This verification process is the basis for our classification scheme.

To facilitate the description of functional verification, we represent each class as a graph (see Fig. 3), in which each node represents a functional part of the class, and the edges represent relationships between functional parts. We call this graph the functional class graph. A "functional part recognizer" is assigned to each functional part, and additional knowledge, in the form of cues, is provided to them. This additional knowledge is gathered throughout the verification process and makes relationships between parts explicit. For example, the cue "what surface is to

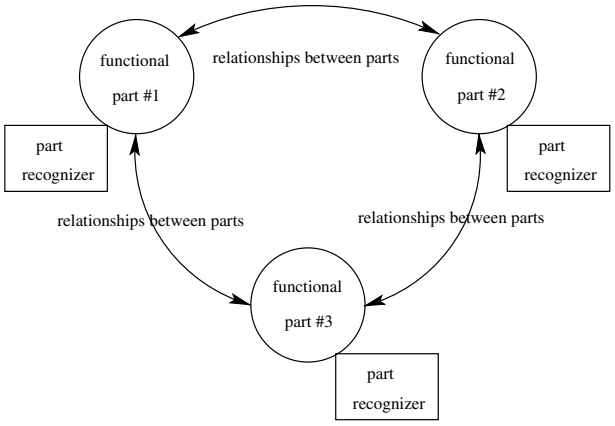

Fig. 3. A functional class graph.

be supported'' suggested in the above support-to-ground example is, in fact, provided by a hypothesis for the chair seat. This cue explicitly addresses relationships between the seat and the support-to-ground (stable support and height above ground, for example).

Our classification scheme employs a bottom-up verification of each class and can also be seen as searching for a best mapping of primitive parts to functional ones. This verification process is performed as described in Algorithm 1.

Let FP be the set of functional parts that appear in a functional class graph FCG and PP a set of primitive parts. Our classification algorithm (see Algorithm 1) constructs a set of pairs denoted by $map \in PP \times FP$. An object is classified if

$$\forall fp \in FP, \quad \exists p \in PP \times FP,\ p = (p, fp).$$

Each hypothesis h is described in our algorithm as a subset of $PP \times FP$.

Note that the recursive steps, in which *map* is constructed (in the input and in lines 8 and 10 of Algorithm 1), are equivalent to proposing and verifying hypotheses. We describe the algorithm in terms of extending the mapping and hypotheses. In this algorithm, we call a *utility* and *probability* evaluation procedure, denoted **UP_Algorithm**, which is responsible for determining the search path to be followed. This procedure employs our probabilistic model and is described in Section 3.5 in Algorithm 2. The **UP_Algorithm** returns two sets as lists L_{fp} and SL. L_{fp} (line 5 of Algorithm 1) is a list of functional parts ordered by their decreasing utility U values, and SL is a set of lists of hypotheses, where $SL(fp)$ represents an ordered list of hypotheses, realizing the different design patterns, for the functional part fp, sorted in decreasing probability.

Each hypothesis (line 7 of Algorithm 1) is explored by picking a functional part and exploring its possible realizations. Relationships between functional parts are expressed throughout the traversal of the search graph by cues provided by already found hypotheses, which are given as input to recognizers of unexplored parts.

Algorithm 1. Functional Verification Algorithm

Input:
A set of primitive parts - PP
FCG - A functional class graph, (let FP be its set of functional parts)
map - A list of pairs $(p_1, p_2) \in PP \times FP$
A set of cues - *repository*
A threshold for grade - *threshold*
Output:
A functional classification grade

Functional Verification (PP, *map*, FCG, *repository*, *threshold*)

1: **if** all the nodes in FCG have been explored **then**
2: **return** (*whole – object – test*(PP, *map*, FCG, *repository*))
3: **end if**
4: *grade* ← 0.0
5: (L_{fp}, SL) ← **UP_Algorithm**(PP, FCG, *map*, *repository*, *threshold*) (see Algorithm 2)
6: **for all** $fp \in L_{fp}$ **do**
7: **for all** hypothesis $h \in SL(fp)$ **do**
8: *map_fp* ← *map* ∪ h
9: *repository_fp* ← *repository* ∪ {*new_cues*}
10: *grade* ← max(*grade*, **FunctionalVerification**(PP, *map_fp*, FCG, *repository_fp*, *threshold*))
11: **if** *grade* > *threshold* **then**
12: **return** (*grade*)
13: **end if**
14: **end for**
15: **end for**
16: **return**(*grade*)

We keep a *common knowledge repository* (line 9 of Algorithm 1) – a set of cues that are relevant to several functional parts. This repository, which is updated when different hypotheses are explored, provides cues for classification processes of other parts. By using a common repository, we can often avoid the exponential nature of the classification problem. The cues are updated whenever a new hypothesis is proposed. For example, in the case of the chair, the sitting surface may be kept in the common repository of the chair class. When a hypothesis of a sitting placeable is explored, the repository is updated accordingly. The sitting surface realization, as found in the repository, can then provide an effective cue for the classification of a support-to-ground.

When the algorithm finds a possible realization for each functional part in FCG, a ''whole object test'' is performed (line 2 of Algorithm 1). This test is necessary because some relationships cannot be expressed in terms of cues. These relationships would best be tested after all hypotheses have been found. When verifying a chair, for example, we should test whether the combination of a seat, a back-support, and a support-to-ground provides enough room for

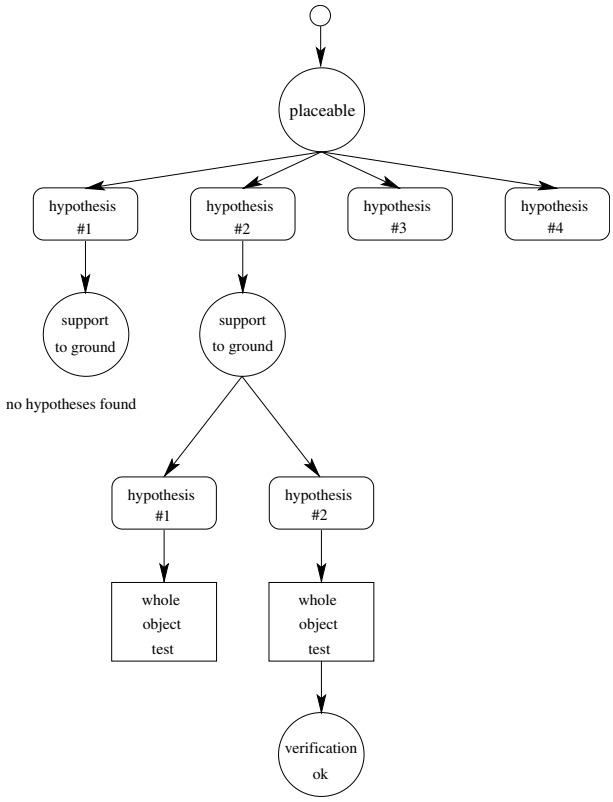

Fig. 4. A verification tree.

the sitting person's body and legs. Such a test is effectively performed at the "whole object test" phase, after hypotheses for all three parts have been found. The imaged object is verified as belonging to the class if this test succeeds. If the test fails, other paths will be attempted. An example of a verification tree is shown in Fig. 4.

3.5. Cost estimation of the search

In the verification process proposed above, the functional class graph is traversed, and each traversal corresponds to a path in the search tree from the root to one of its leaves. If the input imaged object were indeed an instance of the verified class, a valid path to a leaf would be found in the tree verification. Moving the verification procedure control to a child of a node of the verification tree means checking an unexplored functional part node in the functional class graph, i.e., verifying one of its hypotheses. Usually, since each functional part has several realization hypotheses, the verification tree is exponential in the number of parts to be verified. Therefore, an efficient traversal of the tree is a major factor in performance speed-ups. In this section, we propose a scheme that defines the order in which the functional parts to be recognized are chosen, as well as the order in which hypotheses for each part are explored.

We wish to estimate, at each phase of the traversal, just "how good" our current hypothesis path is. We also wish to define criteria for choosing the next step in our traversal, in order to minimize the time spent trying to verify a certain path. We describe a traversal scheme based on probabilities and utility weights, and estimate how much futile work is expected to be carried out in the analysis of each functional part.

Using utility estimations in order to decide what action to be taken has already been used in computer vision in order to decide which action robots should take for navigation purposes [43] and controlling multi-agent systems [29]. The closest example to our utility model is [9], which recommends utility based reasoning for reconstruction of specific 3D objects (parts) and not classes. We use utility estimations to guide the recognition process.

In the case where the functional part *does* exist in the image, we expect to find it in one of the realization hypotheses. We assume that, on average, one-half of the realizations have to be explored until the "correct" hypothesis is found. The total amount of futile work in this case is: $\frac{1}{2} * W_r * n_h$, where W_r is the expected work for exploring each realization of the functional part r, and n_h is the expected number of possible realizations in the image for this functional part. If, however, the functional part was *not* found, all realizations were explored but none led to a valid verification. The total amount of futile work in that case is: $W_r * n_h$. We conclude that the total mean futile work (considering both cases and the probability of each case) is therefore:

$$W_{\text{futile}} = P_+ * W_r * \frac{1}{2} n_h + P_- * W_r * n_h = \left(1 - \frac{1}{2} P_+\right) W_r n_h, \quad (1)$$

where P_+ is the probability that the class would be positively verified assuming the current hypothesis for a sub-object, and $P_- = 1 - P_+$. Having defined W_{futile}, we define the *utility* as:

$$U(fp) = 1/W_{\text{futile}}. \quad (2)$$

The validation of functional parts and their hypotheses, described in line 5 of Algorithm 1, is detailed in the *utility* and *probability* evaluation algorithm shown in Algorithm 2.

Algorithm 2. Utility and Probability Computation Algorithm

Input:
PP - A set of primitive parts
FCG - A functional class graph, (let *FP* be its set of functional parts)
map - A list of pairs $(p, fp) \in PP \times FP$
repository - A set of cues
threshold - A threshold for grade
Output:
L_{fp} - A list of ordered functional parts (in their decreasing futile values)
SL - A set of lists of ordered hypotheses, where *SL*(*fp*) represents an ordered list of hypotheses for the functional part *fp* (in their decreasing P_+values)

UP_Algorithm (*PP*, *FCG*, *map*, *repository*, *threshold*)

1: $L_{fp} \leftarrow$ all the unexplored functional parts components $fp \in FP$, i.e., $\forall fp \in L_{fp}$, $\nexists(p, fp) \in map$
2: **for all** $fp \in L_{fp}$ **do**
3: $SL(fp) \leftarrow \emptyset$
4: **for all** hypotheses $h \in hs(fp)$ of realization of $fp \in L_{fp}$ **do**
5: Compute the probability P_+ of h (see Eq. (3))
6: **if** $P_+ > threshold$ **then**
7: $SL(fp) \leftarrow SL(fp) \cup \{h\}$
8: **end if**
9: **end for**
10: $sort(SL(fp))$(in the P_+ decreasing values)
11: **end for**
12: return (L_{fp} and SL)

During the execution of Algorithm 2, we consider all the functional parts f_p that were not treated until the current stage (line 1 of Algorithm 2). We generate all the hypotheses $hs(f_p)$ that can be computed following the design patterns that were implemented (line 4 of Algorithm 2). Further, we evaluate their utility (line 5 of Algorithm 2) and sort the hypotheses. Bad hypotheses are discarded (line 7 of Algorithm 2). The sorted list of remaining hypotheses is returned to the caller at the end.

We now describe in detail how W_r, n_h and $P_{+/-}$ are estimated. For n_h, an estimate is gradually refined throughout the verification process: We begin with a primary estimate based on low-level knowledge acquired by a linear pass on the found primitive parts. Such low-level knowledge may pertain to, for example, the number of sticks found in the image, or the number of plates. The n_h estimate is then refined by considering data found by hypotheses of other functional parts (provided in the common knowledge repository as *cues*). We have experimentally evaluated the speedup provided by the probability mechanism and show the results in Section 5.4.

We estimate the expected work W_r for a part by integrating two factors: the difficulty of exploring the functional part at hand (considering. the known cues) and the average depth of a class verification path after this functional part has been explored (per realization). Usually, W_r is a function of the cues $\{c_i\}$ for the classification of the part, and, as noted previously, these cues express constraints and additional knowledge about the part. Therefore, they affect the number of valid hypotheses proposed for that part, and may even affect the actual steps taken in recognizing it. In other words, the work that will be expended in exploring the realizations of a part may depend on what cues are provided to the classification process. As can be seen in Eq. (1), functional parts with a lower number of hypotheses are processed first. The lower n_h is, the higher the utility U is, and the estimated validation time is shorter.

We estimate P_+ (line 5 of Algorithm 2) recursively by evaluating the probabilities of hypotheses for a sub-object throughout the verification process. We assume that a class R can be verified by determining whether the imaged object indeed belongs to the class R (we denote this event by R^+ and the complementary one by R^-). We also assume that an object of class R is constructed from several functional parts, denoted as r_i. Let R_i be a sub-object consisting of the parts $r_0, \ldots, r_i$. At each phase of the verification, we have a hypothesis H^{R_i} for a sub-object (constructed out of the hypotheses for several functional parts of the object). After a new functional part r_{i+1} is explored, we add hypothesis h_{i+1} to the current hypothesis for the sub-object. We now wish to obtain a probability estimate for our hypothesis. We define the event that the hypothesis H for the sub-object R_i is valid recursively because: $V_{H^{R_{i+1}}} \equiv \{V_{H^{R_i}}, V_{h_{i+1}}, V_{h_{i+1} \leftrightarrow H}\}$, where $H^{R_{i+1}} \equiv H^{R_i} \bigcup h_{i+1}$. Here, $V_{H^{R_i}}$ is the event that hypothesis H realizes R_i, $V_{h_{i+1}}$ is the event that the part r_{i+1} realizes the hypothesis h_{i+1}, and $V_{h_{i+1} \leftrightarrow H}$ is the event that the relationships between R_i and r_{i+1} that are required for validation are positively verified. In other words, a hypothesis H' is constructed from a hypothesis H for a sub-object and a hypothesis h for another functional part of the object. In that case, the probability that the hypothesis will be realized is constructed from the probability that H will be realized, and the probability that h will be realized. Moreover, H' relies on the relationships between H and h. These relationships are expressed in probabilistic terms as well. The probability that the image will indeed be positively verified when assuming a hypothesis for a sub-object is

$$\begin{aligned} P_+ &= P(R^+|V_{H'}) \\ &= \frac{P(V_{H'}|R^+)P(R^+)}{P(V_{H'})} \\ &= \frac{P(V_{H'}|R^+)P(R^+)}{P(V_{H'}|R^+)P(R^+) + P(V_{H'}|R^-)(1-P(R^+))}, \end{aligned} \tag{3}$$

where $P(R^+)$ is the a priori probability for an imaged object to be an instance of the class R, and $P(H'|R^{\pm})$ are calculated recursively using the following equations:

$$P(V_{H'^{R_{i+1}}}|R^+) = P(V_{h_{i+1} \leftrightarrow H}|V_{H^{R_i}}, V_{h^{r_i}}, R^+)P(V_{h^{r_{i+1}}}|V_{H^{R_i}}, R^+) \times \underbrace{P(V_{H^{R_i}}|R^+)}_{\text{recursive term}} \tag{4}$$

and

$$P(V_{H'^{R_{i+1}}}|R^-) = P(V_{h_{i+1} \leftrightarrow H}|V_{H^{R_i}}, V_{h^{r_i}}, R^-)P(V_{h^{r_{i+1}}}|V_{H^{R_i}}, R^-) \times \underbrace{P(V_{H^{R_i}}|R^-)}_{\text{recursive term}} \tag{5}$$

with the following starting conditions: $P(\emptyset|R^+) = 1$, $P(\emptyset|R^-) = 1$.$P(V_{h_{i+1} \leftrightarrow H}|V_{H^{R_i}}, V_{h^{r_i}}, R^{\pm})$ and $P(V_{h^{r_{i+1}}}|V_{H^{R_i}}, R^{\pm})$ can be estimated statistically.

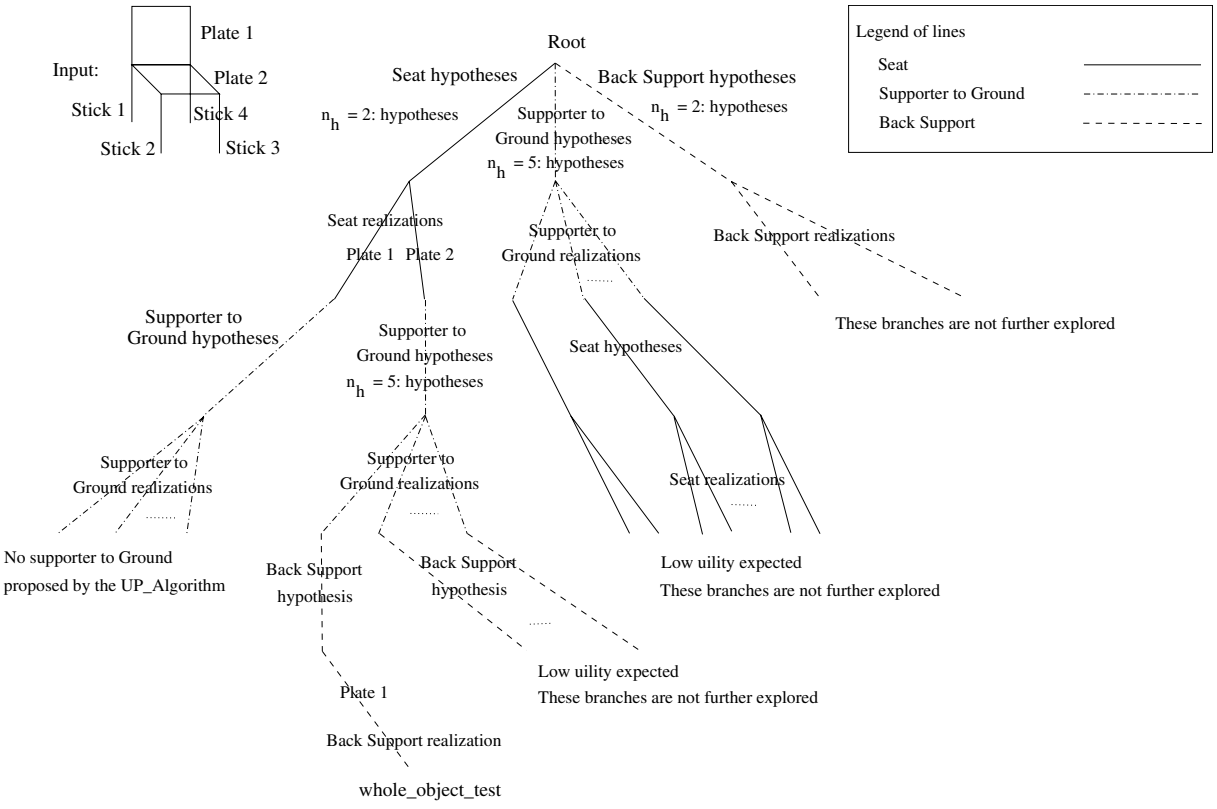

Fig. 5. Expected work and probability in the verification tree.

A possible verification tree with hypotheses and probability considerations is outlined in Fig. 5. In this example, our scheme tries to find first a seat functional part among the three components seat, supporter to ground, and back support. This is the result of the fact that there is a lower number of plates in the input than the number of sticks. In this case, the number of possible realizations of seat and back support is two while the number of realizations for a ground support is five. The algorithm has to therefore choose between trying to verify the existence of seat or a back support. A seat is chosen because after a seat hypothesis is generated and incorporated into the knowledge repository, verifying a supporter to ground is much cheaper. After the seat hypothesis realization, the algorithm chooses to check the existence of a supporter to ground, and not the back-support, since the W_r is lower when the seat is already known. In the end, one hypothesis for the back support is checked and the whole-object-test validates the result.

4. Matching shape parts to functionalities

Our approach is a bottom-up function-based classification scheme which receives as input raw range images and 3D models. The high-level processing of these images is based on the lower-level stages, the details of which will be given in the following sections, along with a discussion of implementation issues. Because the lowest level of processing is that of segmentation, we also present a short review of the literature on that subject.

4.1. Low-level processing—3D segmentation

The literature of range image segmentation is vast and comprises works that deal with comparisons among different algorithms (see [28,30,42]). Besides segmentation into parts using geometric characterizations, we are also interested in recovering primitives [10,23,33,41,57] as well as in analyzing the interconnections among the parts [19].

We begin the segmentation with two steps that follow the first part of the UE algorithm [20,28] and yield an over-segmented image into regions. Next, 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 relationships between regions. The graph is built by classifying the region-to-region connection as one of disconnected, occluded, or connected.

The segmentation analyzes the relationships between adjacent parts in the raw range image. Assume that in a raw range image of a chair the seat is realized by a plate and the ground support is realized by several sticks. The relationship of connectivity between the seat and the leg that is the farthest from the capturing device is usually the hardest and error prone case detection (see Fig. 15(a)). We consider such a relationship as an occlusion and the segmentation detects it by analyzing the jumps in

the distances of the points at edge regions from the capturing devices. In the higher levels of the algorithm hypotheses that assume that the occluding and occluded parts are actually connected are tested.

Because *regions* conform to surfaces of *parts* or fractions of surfaces, we represent each part as a collection of regions. We traverse the connectivity graph and collect connected regions into individual primitive parts. We compute the region-to-region connections of each new merged region to all other regions.

We further classify a part into *a stick* (with allowable deformation) or a *plate* (allowing deformations on the plate). 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 mid-level, see Section 4.2). We note that a stick is a part with one dimension that is considerably larger than the others. In a plate, two dimensions are similar and the third is smaller.

We consider two primitive parts: sticks and plates. For sticks and plates, we distinguish between proper and deformed parts.

(1) Sticks – we represent a stick by its two extremities and one midpoint, thus accounting also for deformed sticks.
(2) Plates – we represent planar plates using three vertices defining a bounded rectangle that has the same orientation as the approximated region. For deformed plates, we employ a nine-point representation as shown in Fig. 6, where the corners represent an oriented bounding rectangle, the midpoints are averages of edge corners, and the center point is the mass center of the region.To evaluate the degree of deformation of a plate and obtain a more compact and high-level representation for deformed plates, we match the point cloud of this part to *quadrics* (see [21]). Quadrics enable classification of the primitive parts as a cylindric or elliptic surfaces.

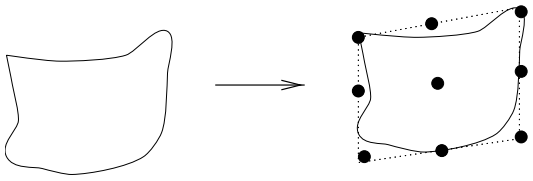

Fig. 6. Deformed plate representation.

The input to our classification scheme, and thus to the lowest-level processing stage, is a raw range image represented as a *point cloud*. This point cloud is segmented in the first processing stage. Fig. 7 illustrates a segmentation result of an airplane. The airplane's fuselage is modeled by a deformed stick while its wings are represented by plates.

4.2. Mid-level processing

In range images, a plate (deformed or non-deformed) may be a face of a blob, which is a part in which all the dimensions are similar. We represent blobs by a collection of plates (deformed and non-deformed) and the relationships between them. This allows us to verify functional constraints more easily in the higher level and handle a variety of blobs: polyhedral, curved, convex, and concave. In order to describe concave blobs such as containers, we allocate a curve for the aperture of the concavity. We search for pairs of deformed plates having occlusion relationships and locations at which one plate is convex and the other is concave. After the boundary of these plates is found, a curve for the possible aperture within these plates is extracted. We experimented more types of blobs realizations. However, the most reliable realizations were inferred from sets of deformed plates. While recovering blobs seems to be a segmentation task, this problem can be considered as low-level processing. On the other hand, we employ elements of perceptual grouping of primitives, towards this goal, which is a higher level processing task. The most natural way to characterize blobs recovering tasks is, therefore, mid-level processing. Fig. 8 shows a blob reconstruction in a real range image of a couch-like chair.

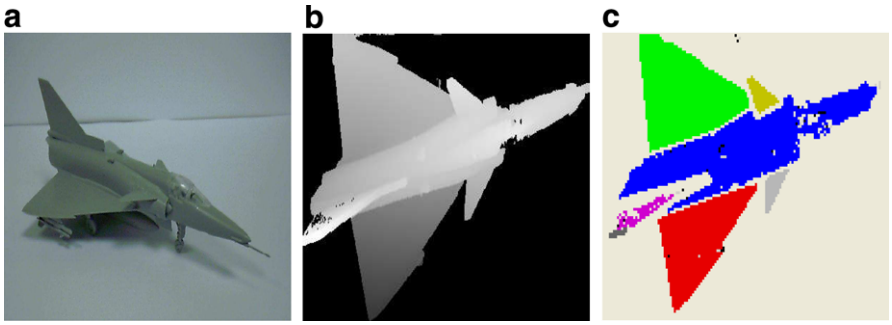

Fig. 7. Low-level processing: the segmentation of a raw range image of an airplane. (a) Represents a digital image of an air-plane, (b) is a range image of the airplane in (a), (c) represents the segmentation of the range image (b), where the blue region shows the fuselage identified as a deformed stick while the wings and the stabilizers are represented by plates.

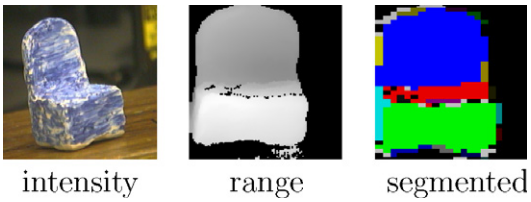

Fig. 8. The classification of a couch-like chair having a blob support-to-ground.

4.3. High-level processing

As discussed in Section 3, in order to verify if the decomposed shape parts of an image conform to a known class, functional representations for known classes should be obtained. This can be done using generic and class-specific functional parts.

4.3.1. Generic functional parts

We implemented and tested several generic functional parts: placeable, leg, support-to-ground, handle, container, central-axis, wing, off-ground support, and stabilizer (in air).

Let us first consider the placeable. There are certain parametric constraints on its dimensions, allowable deformation, surface direction, and clearance, i.e., the amount of space, relative to the placeable, that should be clear of obstacles. We check several properties for the placeable, including surface and dimensions, as well as its center and orientation. Possible realizations of the placeable are shown in Fig. 9.

The constraints on the "support-to-ground" include the shape part to be supported, the direction of the ground, and the allowed range for height. The properties we check for "support-to-ground" include its height and the supported parts. The possible realizations for a "support-to-ground" are shown in Fig. 10. Here, we consider a "leg" as a stick or a collection of several sticks joined together.

Supports-to-ground are classified by identifying the sticks in the primitive parts collection. Then, possible configurations of sticks forming three/four/one-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 a one-legged support, 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 support-to-ground.

The "handle" is defined as a functional part that enables a person to grasp the object it is attached to. The parametric constraints that we consider for a handle are the number of sides (one or two), the direction of handle, the connected part, and the graspability dimensions.

The parametric constraints that we consider for a "container" functional part are the general dimensions, the capacity, the orientation, and the aperture curve parameters. In our current implementation, a container may be realized as a cylindrical part or as a combination of two deformed plates, one of which is convex and satisfies an occlusion relation with the other, which is concave.

The "central-axis" is a functional part that is mainly realized by a stick, which is eventually deformed. The justification for introducing such a functional part is that it can be found in many symmetric objects, for example in airplanes, umbrellas, screws, and rail wheels. These objects are characterized by a central part that unifies the functionalities of all other parts of the analyzed object.

A "in-the-air-support" is a functional part that is realized by two wings having the same surface and size. (Wings are realized by plates.) In addition, the two wings share the same normal. A "stabilizer" is a surface that can provide vertical or horizontal stability. For vertical stability, one small wing suffices. For horizontal stability, however, two small wings of similar size are required. Similarity in size and form are also requirements for horizontal stability.

4.3.2. Functional description of classes

We have implemented classifiers for ten classes: chairs, mugs, glasses, tables, plates, airplane models, rolling-pins, umbrellas, hangers, and bags. The exact numbers of objects used in classification are detailed in Table 1. These common or widely available objects can be conveniently captured in 3D. In the following, we describe the functional decompositions for chairs, tables, mugs, and airplanes, other classes having similar structures.

We recognize the chair functionality class to be comprised of the relationships between a seat, a supporter-to-ground, and a back-support, which is optional (see

Fig. 9. Possible placeable realizations.

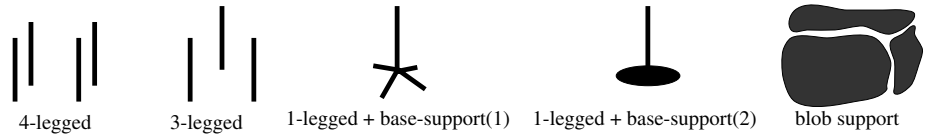

Fig. 10. Several possible support-to-ground realizations.

Table 1
We tested 150 objects

Tested objects	Valid classes	Non-valid classes
Chairs	15	8
Mugs	12	17
Glasses	7	3
Tables	7	5
Plates	3	0
Airplanes	17	1
Rolling-pins	2	2
Umbrellas	2	2
Hangers	4	1
Bags	3	1
Total	72	40

Besides the objects shown in the table, there are another 38 objects that can be divided in two groups. The first group consists of 23 objects for which we did not construct classifiers. The second group consists of 15 objects that were segmented but not classified because the functional part recognizers were limited in their ability to recognize rare or unusual realizations of objects. Each object was captured in one orientation, i.e., different range images represent different objects.

Fig. 11). The seat and the back support are each realized by a placeable, and the seat-to-ground support is realized by a support-to-ground.

We verify the relationships between the parts to determine that the seat is perpendicular to the back support; the ground support provides a stable support for the seat; the seat is connected to the back support; the ground support is connected to the seat; and a combination of a seat, a back support, and a ground support provides seating and room for the legs.

We consider that a table consists of a desktop and a ground support. Moreover, we model mugs by a container and a handle.

Airplanes are realized by a central axis part, an off-ground support, a vertical stabilizer, and, optionally, a horizontal stabilizer. These functional parts are realized by a deformed stick, a set of two plates, a single plate, and a set of two plates respectively. The vertical stabilizer is verified by measuring the angle between the normals to the surface of the off-ground support and to the vertical stabilizer. This angle is considered valid when it is close to $\frac{\pi}{2}$. In addition, we require that the fuselage, which is represented by the central axis, be connected to all the other functional parts. For airplanes, we used the decomposition shown in Fig. 12.

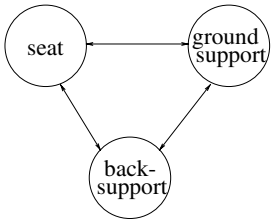

Fig. 11. Chair functional parts.

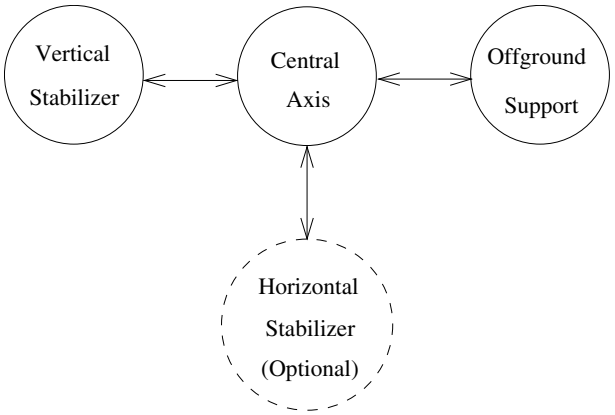

Fig. 12. Airplane functional parts.

4.3.3. Implementation of class verification

In addition to the generic functional part definitions and class-specific functional definitions, a generic class verification procedure was implemented, following the concepts defined in Section 3. The verification procedure evaluates the quality of the primitive shape parts by constructing hypotheses and computing the quality of their relationship to hypotheses for other parts (in other words, how well the hypotheses conform to the cues). Moreover, the verification procedure accounts for the distribution among design patterns (for example, a four-legged support-to-ground hypothesis gets a higher grade than a one-legged one). In addition, a matching grade that represents the degree of conformance to constraints is computed as well. This matching grade implementation employs conformance tests for Gaussian distributions of range constraints. (We have also tested uniform distributions, with no significant difference in the results.)

5. Experimental results

We have tested our implementation on a database that includes 150 range images of real objects, scanned employing a Cyberware laser-based range scanner. The database comprises ten classes of objects. Each one of the range images includes one object, where different range images represent different objects. In addition, we used several cluttered images, where the cluttering was built on objects captured in the single object range images.

The database includes 72 objects that are valid instances of classes for which we have built classifiers, 48 objects that are functionally non-valid instances of these known classes (e.g., an unstable chair or an ungraspable mug), and 23 objects for which we did not implement classifiers (boxes and hand-tools, for example). We also tested 15 objects for which there are classifiers; however, we did not implement the required specialized functional part recognizers because these particular objects are very rare realizations of their functional parts (see, for example, Fig. 13). Throughout the tests, we verified that the low and mid-level phases, as well as the high-level classification, perform

correctly. Examples of intensity images of some of the tested objects are shown in Fig. 14. A summary of the tests performed on real objects containing our supported functional parts is presented in Table 1.

In functional reasoning schemes, and computer vision systems in general, the number of parameters to be tuned is non-negligible. We implemented our scheme so that most of the parameters belong to the classifiers only and not to the common generic code. A few objects in each class are sufficient for determining the parameters required for robustness.

In the airplanes case, six instances were sufficient to fix the parameters for classifying all 17 valid airplanes we used in the experiments. We computed the standard deviations of all the parameters in an increasing sequence of three, four, five, and six airplanes. These measures provided boundaries for parameters. With each set of boundaries, we experimented the classification results over all the airplanes in the database. There was no need into enlarge the training set to more than six planes. In this context, one of the important parameters was the angle between the central axis and the normals of the off-ground supports. We have found that an error of 1% around $\frac{\pi}{2}$ is characteristic to our models. In the following sections we present several running examples of the algorithm.

Fig. 13. A rare realization of the support-to-ground functional part.

5.1. Classification of valid instances

Fig. 15(a) demonstrates the classification of a standard valid chair that has three functional parts (a seat, a back support, and a support-to-ground). The correct hypothesis combination passed the verification stage and the image was classified as a chair. Verification of the chair image as mug and glass failed because no container was found. Verification of the chair image as table failed because no

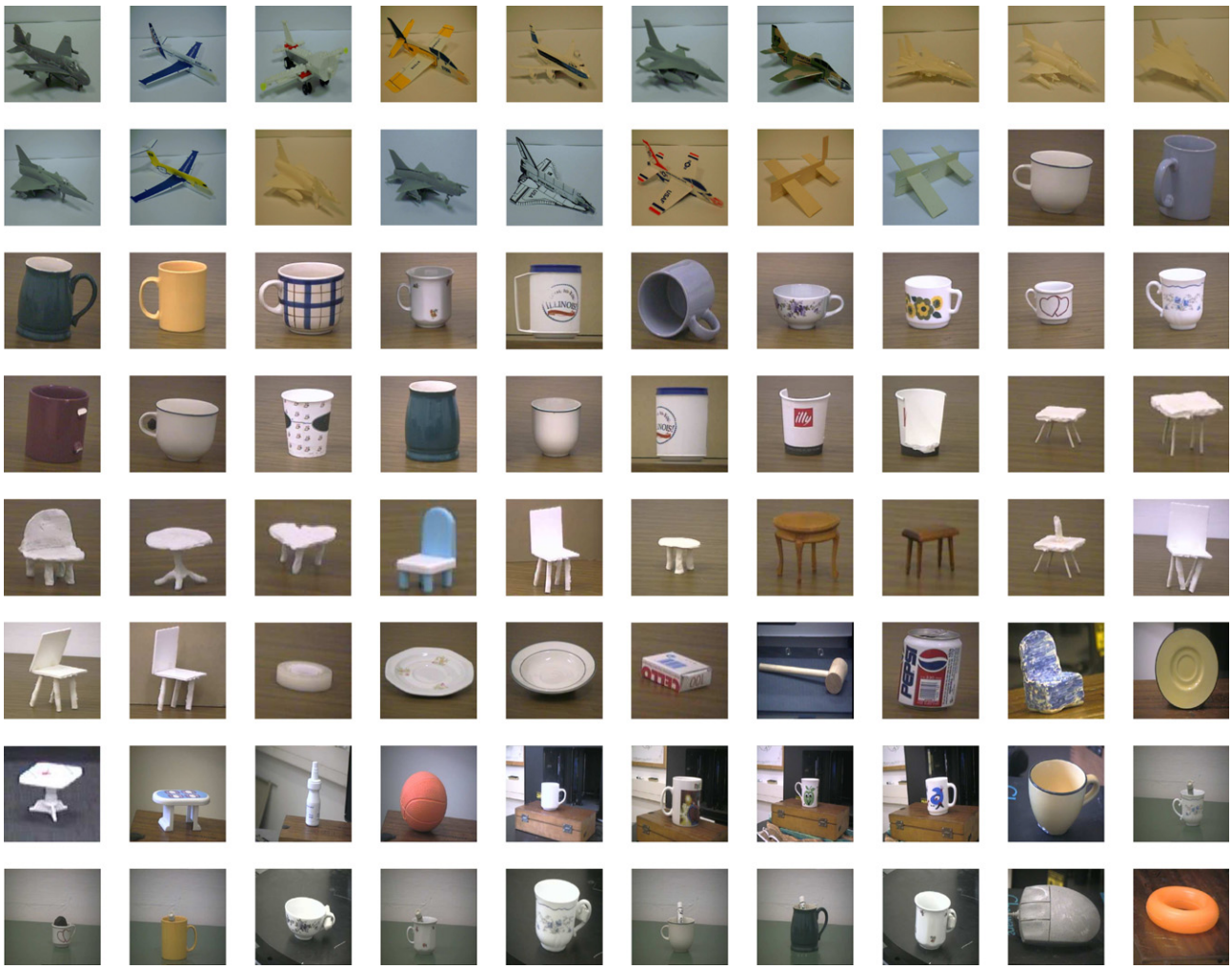

Fig. 14. Intensity images of some tested objects.

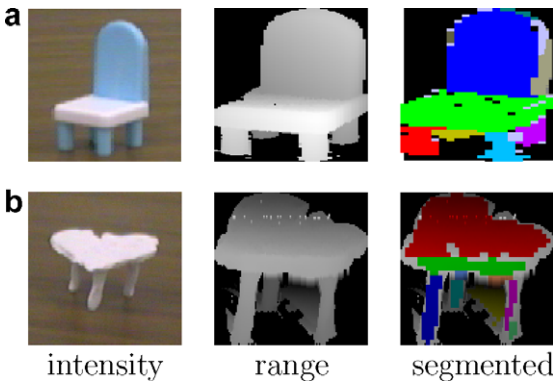

Fig. 15. The first triplet, (a) represents a standard valid chair with a back support. The second triplet, (b) shows the classification of a valid three-legged chair.

correct tabletop was found (improper clearance or dimensions).

Moreover, Fig. 15(a) is an example of reasoning in terms of occlusion. The farthest leg, the one that is segmented in yellow, is connected with the seat by an occlusion relationship. This relationship is inferred at segmentation and provided to the input to the reasoning stages as a cue for the knowledge repository mechanism. At the recognition stage, a hypothesis assuming that the leg is connected to the plate is tested and verified.

Fig. 15(b) demonstrates the classification of a stool, which is a three-legged chair without a back support. Unlike a standard chair, the seat is heart-shaped and is realized by a deformed plate. The low-level phase segmented the image correctly into one deformed plate and three sticks conforming to the legs. The first functional part to be elaborated was chosen to be the support-to-ground. This part was chosen for its higher utility estimate, which derives from the fact that the part consists of one simple hypothesis of a combination of the three legs. This is in contrast to the previous example, where the first explored part was the seat. In that example, the choice of which part to explore first was based on the fact that the segmentation provided many sticks, which led, in turn, to a large number of possible configurations as supports-to-ground. After a hypothesis for support-to-ground was found, the seat part was explored and a valid hypothesis found. The image was classified as a valid chair.

The classification of a blob as a support-to-ground in a couch-like chair is demonstrated in Fig. 8. The mid-level processing stage segmented the image into several plates conforming to the chair itself and a collection of plates and sticks (deformed and non-deformed). This was due to over-segmentation and clutter. The high-level phase chose to start with the seat (for reasons of utility) and found several valid hypotheses, followed by a valid blob support-to-ground for one only. The image was correctly verified as a valid chair.

Fig. 16(a) demonstrates the classification of a standard valid mug with a cylindrical container. The classification of a valid non-cylindrical mug with an over-segmented decomposition into parts is demonstrated in Fig. 16(b).

The segmentation of an airplane is shown in Fig. 7. The vertical stabilizer is usually implemented by a plate. In the scene, several plate candidates are recovered by the segmentation stage. Therefore, several whole-object-tests verifying their orthogonality to the horizontal stabilizer are required until the vertical stabilizer can be correctly classified. This demonstrates the importance of the whole-object-test for the correct classification of the object and its parts. Note that using certain design patterns prevent us for recognizing other exotic designs. For example, an F117 (see Fig. 17), as it does not have vertical and horizontal stabilizers, will not be recognized by our current scheme.

5.2. Classification of non-valid class instances

The classification of a non-valid chair is demonstrated, in Fig. 18, on a chair missing one of its legs. Among support-to-ground hypotheses, none was found to provide a stable support for the seat.

Fig. 17. A F117 plane.

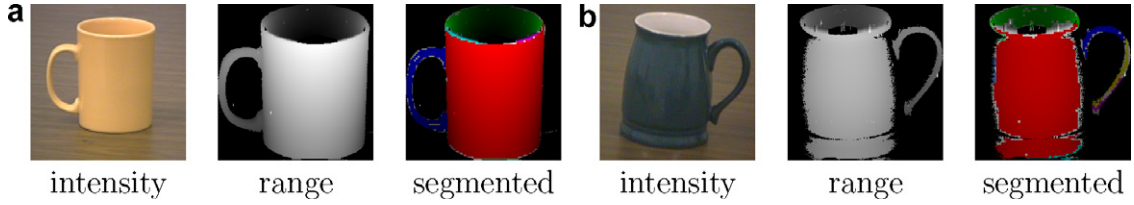

Fig. 16. The first triplet, (a) shows a classification of a valid cylindrical mug while the second one (b) represents a classification of a valid non-cylindrical and over-segmented mug.

Fig. 18. A classification of a non-valid chair, missing one of its legs.

Fig. 19. A classification of a coffee mug having an obstacle in its handle.

In Fig. 19, we demonstrate the classification of a mug that is not valid because it has an ungraspable handle. Although the high-level phase found a valid container and then found several hypotheses for handles, all were ruled out. The "real" handle had correct grasping dimensions but did not have grasping clearance. Therefore, the verification of *mug* failed.

5.3. Classification in a cluttered environment

Fig. 20 demonstrates the correct classification of a valid three-legged chair in a neighborhood consisting of other "non-class" objects (pencils and a piece of chalk). The low-level stage segmented the image into a plate conforming to the chair seat and a collection of sticks (deformed and non-deformed) conforming to the legs and to the other objects. The high-level phase chose to start with the seat (for reasons of utility, and in contrast to the previous example, where the existence of many sticks suggested a large number of support-to-ground realizations to explore). This phase found several valid hypotheses, then found a valid support-to-ground consisting of the three legs. The clutter of the other objects did not affect the functional validity of the "real" chair, which was correctly verified.

Besides accuracy in classification, schemes for classification in cluttered environments have to cope with time requirements feasibility. We provide the time measurements required by a classification in cluttered environments experiment in Section 5.4.

Fig. 20. A correct classification of a valid chair in a cluttered environment.

5.4. Utility and probability considerations

Using the examples shown above, one can see how probability and utility affected the classification process. We want to emphasize the importance of two things: the order in which the functional parts were chosen and the order in which the realization hypotheses were explored. For each functional part, the hypotheses that were most probable were explored first.

Consider, for example, the class of chairs. There, the algorithm has to decide whether to recognize first the support-to-ground or the seat. In the chair in Fig. 15(a), the first part to be elaborated was the seat (because. over-segmentation of the sticks in the scene resulted in many support-to-ground hypotheses). In the chair in Fig. 15(b), however, the first part to be realized was the support-to-ground because the small number of leg combinations being considered led to greater utility.

In order to prove the efficiency of our utility functions we considered the classification of the chair in Fig. 21. We segment the raw range image of the object and cluttered it with deformed sticks. Each time a deformed stick was added we measured the time required by the classification. We summarize the times in Table 2. The measurements indicate that the time required for classifications increases linearly with the number of added primitives.

All the images of airplanes in our experiments contain one deformed stick and several plates. The fuselage is analyzed first, followed by the vertical stabilizer and the off-ground support. This is because classification of the fuselage requires a deformed stick, while a vertical stabilizer or an off-ground support requires one or two wings realized as plates. Therefore, the first functional part to be analyzed is the fuselage, followed by the vertical stabilizer and the off-ground support.

In order to demonstrate the importance of the utility- and probability-driven approach, we compared, for the

Fig. 21. An image of a chair that was used in a cluttered environment.

Table 2
Times measured in a cluttered image of the chair in Fig. 21

	Original object	Cluttering sticks added				
		1	2	3	4	5
Time in seconds	0.061	0.065	0.07	0.076	0.08	0.084

airplane class, the time required by our classification scheme when the probabilistic mechanism described in Section 3.5 was turned on and off. When the mechanism is turned off, an arbitrary functional part is chosen by the **UP_Algorithm**. Turning the mechanism on led to a 25% speedup in classification time in average. We did not, however, consider the time required for the probabilistic mechanism itself. This time is considered to be negligible.

In average, the time required for classification of each object is half a second. The less time demanding classification was observed when the input consisted of cups and airplanes. In these cases, the time required by the cups and airplanes classifiers is a tenth of a second. The highest classification time was registered when the input consisted of chairs. The chair classifier required up to one second for a positive classification. All the classifiers that provided negative classifications required less than $\frac{1}{100}$ seconds. We measured these classification times on a Pentium 4 processor, 1.7 GHz, equipped with 256 Megabytes of memory.

We also tried an interleaved implementation. We asked if the analyzed object is one of five classes: chair, glass, mug, bed, or plate. Each of the five classifiers works as usual; however, only one of them receives processing time at a certain moment. The classifiers compete to gain processing time at the moment of proposing functional parts and hypotheses to be recognized. The classifier that proposes hypotheses for a functional part with the highest P_+ (see Section 3.5) receives processing time. The scheme runs until one of the classifiers reports a valid classification or all the classifiers stop their processing.

We also measured the time required by all the classifiers together and divided it by two. This was done because, not knowing the best order for the classifiers, we wanted to try all possible combinations. Therefore, on average, half the time that would be required for running all the possible combinations is required for finding the correct one.

In this experiment, we used chairs, that is objects which can be recognized, i.e., for which we have built a classifier. Note that the objects in our database are relatively simple. We expect the search times to be even faster when the probability mechanism applied to complex objects, rather than the relatively simpler ones we used in our experiments. Moreover, a higher number of classes to be verified will also improve the run time ratio.

5.5. The role of verification in functional reasoning

The final validation checks the interrelationships that can, naturally, only be checked in the final stages of computation, when all the functional parts have been recognized, or, at least, hypothesized. Consider the case of airplanes. The central axis, or the fuselage, is connected to all other functional parts. When the classification does not begin with hypothesizing the fuselage, the test of connectivity between the central axis and all other parts of the airplane should naturally be delayed until all the functional parts have been hypothesized.

We performed several experiments that prove the efficiency and reliability of the final verification stage. For example, we tested airplanes to see if the normal of the off-ground support is perpendicular to that of the vertical stabilizer. All seventeen valid airplanes were correctly classified and their vertical stabilizer recognized. When we omitted this stage, all the airplanes were still classified, but small wings from the horizontal stabilizer were incorrectly recognized as vertical stabilizers.

Moreover, the verification stage has the added advantage of improving classification certitude. As part of our tests we modified correct objects, that were formerly recognized by our scheme, into invalid objects (for example we add an object on the seatable surface). We submitted them again to classification. In these tests, the final detection of the modification took place in whole-object-test stage.

6. Conclusions

In this paper we address the problem of object classification from raw range images. A new implementation for a functional based classification scheme is presented. Our approach addresses low and high-levels aspects of the problem by combining structural and functional approaches, each aspect being treated in a separate phase. In the low-level phase, a shape part decomposition process segments the raw input image into a collection of primitive shape parts, and properties of parts and relationships between them are calculated, to be later analyzed in a functional context. In the high-level phase, a probability and utility-guided verification process verifies whether the decomposed parts of the image conforms to the description of one of the known classes. A functional model for class description by functional parts is also proposed, demonstrating how various classes are represented by only a few *generic* functional parts. A mapping of functional parts to configurations of primitive shape parts is presented as well, using functional part recognizers and common design patterns.

The algorithm has been implemented and tested on 150 range images of real objects. Differently shaped objects are decomposed into parts and classified, thus proving the feasibility of our approach and its robustness to over-segmentation and cluttered scenes.

The main contributions of this work are a new search criteria, implemented by means of a probability mechanism, and a verification stage that can be performed in the last stages of computation only, when all the hypotheses about functional parts are known. We report that the recognition process speeds up when the probability mechanism is turned on. We expect this speedup to increase even further when the probability mechanism used in the recognition of complex objects, rather than the relatively simple objects used in our experiments. Moreover, it was experimentally proven that the verification stage led to more accurate classification.

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 of the existing functional parts. Applying statistical analysis to classification results of a large number of input images will lead to more efficient probability design. That is, the estimations of the amount of work required by the verification process will be more accurate and the order of verifying hypotheses will enable shorter processing time. The introduction of new classes can be generalized by adding learning algorithms to the classification process. We expect to learn various realizations in a granular way. This means that functional parts that have the same geometry and are shared between different classes do not have to be learned separately with each new class.

References

[1] SOAR, http://sitemaker.umich.edu/soar, 2006.

[2] G. Adorni, M. DiManzo, F. Giunchiglia, L. Massone, A conceptual approach to artificial vision, in: The Fourth Conference on Robot Vision and Sensory Controls, 1984, pp. 231–260.

[3] R. St. Amant, A.B. Wood, Tool use for autonomous agents, in: Proceedings of the National Conference on Artificial Intelligence (AAAI), 2005, pp. 184–189.

[4] R. Bajcsy, F. Solina, Three dimensional object representation revisited, in: Proceedings of the IEEE International Conference on Computer Vision, 1987, pp. 231–240.

[5] R. Bergevin, M.D. Levine, Generic object recognition: building and matching coarse descriptions from line drawing, IEEE Transactions on Pattern Analysis and Machine Intelligence 15 (1) (1993) 19–36.

[6] E. Bicici, R. St. Amant, Reasoning About the Functionality of Tools and Physical Artifacts. Technical Report 22, NC State University, 2003.

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

[8] L. Bogoni, R. Bajcsy, Interactive recognition and representation of functionality, Computer Vision and Image Understanding 62 (2) (1995) 194–214.

[9] R.C. Bolles, P. Horaud, 3dpo: A three-dimensional part oriented system, The International Journal of Robotics Research 5 (3) (1996) 3–26.

[10] S.K. Bose, K.K. Biswas, S.K. Gupta, An integrated approach for range image segmentation and representation, Artificial Intelligence in Engineering 1 (1996) 243–252.

[11] M. Brand, Physics-based visual understanding, Computer Vision and Image Understanding 65 (2) (1997) 192–205.

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

[13] R.A. Brooks, Model-based three-dimensional interpretations of two-dimensional images, IEEE Transactions on Pattern Analysis and Machine Intelligence 5 (2) (1983) 140–149.

[14] R.A. Brooks, R. Greiner, T.O. Binford, The acronym model-based vision system, in: Proceedings of the International Joint Conference on Artificial Intelligence, pp. 105–113, 1979.

[15] E. Davis, The naive physics perplex, AI Magazine 19 (4) (1998) 51–79.

[16] M. DiManzo, E. Trucco, F. Giunchiglia, F. Ricci, Understanding functional reasoning, International Journal of Intelligent Systems 4 (1989) 431–457.

[17] C. Dorai, A.K. Jain, Cosmos – a representation scheme for 3d free-form objects, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (10) (1997) 1115–1130.

[18] T.J. Fan, G. Medioni, R. Nevatia, Description of surfaces from range data using curvature properties, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 86–91, 1986.

[19] T.-J. Fan, G. Medioni, R. Nevatia, Recognizing 3-D objects using surface descriptions, IEEE Transactions on Pattern Analysis and Machine Intelligence 11 (11) (1989) 1140–1157.

[20] A. Fitzgibbon, D. Eggert, R.B. Fisher, High-level cad model acquisition from range images, Computer-Aided Design 29 (4) (1997) 321–330.

[21] A. Fitzgibbon, M. Pilu, R.B. Fisher, Direct least square fitting of ellipses, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (5) (1999) 476–480.

[22] K. Green, D. Eggert, L. Stark, K. Bowyer, Generic recognition of articulated objects through reasoning about potential function, Computer Vision and Image Understanding 62 (2) (1995) 177–193.

[23] E. Hameiri, I. Shimshoni, Estimating the principal curvatures and the darboux frame from real 3-d range data, IEEE Transactions on Systems, Man, and Cybernetics, Part B 33 (4) (2004) 626–637.

[24] F. Han, S.-C. Zhu, Bottom-up/top-down image parsing by attribute graph grammar, in: IEEE International Conference on Computer Vision, pp. 1778–1785, 2005.

[25] M. Hebert, T. Kanade, I. Kweon, 3-D vision techniques for autonomous vehicles, Analysis and Interpretation of Range Images (1990) 273–337.

[26] J. Hodges, Functional and physical object characteristics and object recognition in improvisation, Computer Vision and Image Understanding 62 (2) (1995) 147–163.

[27] R. Hoffman, A.K. Jain, Evidence-based recognition of 3-D objects, IEEE Transactions on Pattern Analysis and Machine Intelligence 10 (6) (1988) 783–802.

[28] A. Hoover, G. Jean-Baptiste, X. Jiang, P.J. Flynn, H. Bunke, D.B. Goldgof, K. Bowyer, D.W. Eggert, A. Fitzgibbon, R.B. Fischer, An experimental comparison of range image segmentation algorithms, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (7) (1996) 673–689.

[29] F.V. Jensen, H.I. Christensen, J. Nielsen, Bayesian methods for interpretation and control in multiagent vision systems, in: Kevin W. Bowyer (Ed.) SPIE – The International Society for Optical Engineering, Applications on Artificial Intelligence X: Machine Vision and Robotics, vol. 1708, March 1992, pp. 536–548.

[30] X. Jiang, K. Bowyer, Y. Morioka, S. Hiura, K. Sato, S. Inokuchi, M. Bock, C. Guerra, R.E. Loke, J.M.H. du Buf, Some further results of experimental comparison of range image segmentation algorithms, in: Proceedings of the International Conference on Pattern Recognition, vol. 4, pp. 877–881, 2000.

[31] Y. Keselman, S. Dickinson, Generic model abstraction from examples, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (7) (2005) 1141–1156.

[32] D.J. Kriegman, T.O. Binford, T. Sumanaweera, Generic models for robot navigation, In Image Understanding Workshop, 1988, pp. 453–460.

[33] A. Leonardis, A. Jaklič, F. Solina, Superquadrics for segmenting and modeling range data, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (11) (1997) 1289–1295.

[34] W.-J. Li, T. Lee, Object recognition and articulated object learning by accumulative hopfield matching, Pattern Recognition 35 (2002) 1933–1948.

[35] D. Marr, H.K. Nishihara, Representation and recognition of the spatial organization of three dimensional structure, in: Proceedings of the Royal Society of London B, 1978, pp. 269–294.

[36] M. Minsky, Commonsense-based interfaces, Communication of the ACM 43 (8) (2000) 67–73.

[37] L. Morgenstern, Mid-sized axiomatizations of commonsense problems: A case study in egg cracking, Studia Logica 67 (2001) 333–384.

[38] A.P. Pentland, Recognition by parts, in: Proceedings of the IEEE International Conference on Computer Vision, 1987, pp. 612–620.

[39] A.P. Pentland, Automatic extraction of deformable part models, International Journal of Computer Vision 4 (2) (1990) 107–126.

[40] P. Peursum, S. Venkatesh, G.A.W. West, H.H. Bui, Using interaction signatures to find and label chairs and floors, IEEE Pervasive Computing 3 (4) (2004) 58–65.

[41] M. Pilu, R.B. Fisher, Recognition of geons by parametric deformable contour models. In European Conference on Computer Vision, 1996, pp. 71–82.
[42] M.W. Powell, K.W. Bowyer, X. Jiang, H. Bunke, Comparing curved-surface range image segmenters, in: Proceedings of the IEEE International Conference on Computer Vision, 1998, pp. 286–291.
[43] R. Rimey, C. Brown, Task-Oriented Vision with Multiple Bayes Nets. Active Vision, The MIT Press, Cambridge, Massachusetts, 1992.
[44] E. Rivlin, S.J. Dickinson, A. Rosenfeld, Recognition by functional parts, Computer Vision and Image Understanding 62 (2) (1995) 164–176.
[45] E. Rosch, Cognition and Categorization, in: E. Rosch, B. Lloyd (Eds.), Erlbaum, Hillsdale, NJ, 1978.
[46] L. Stark, K. Bowyer, Generic object recognition using form and functionMachine Perception and Artificial Intelligence, vol. 10, World Scientific, 1996.
[47] L. Stark, K.W. Bowyer, Indexing function-based categories for generic recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1992, pp. 795–797.
[48] L. Stark, K.W. Bowyer, Function-based generic recognition for multiple object categories, Computer Vision, Graphics, and Image Processing 59 (1) (1994) 1–21.
[49] L. Stark, A.W. Hoover, D.B. Goldgof, K.W. Bowyer, Function-based recognition from incomplete knowledge of shape, in: Proceedings of the IEEE Workshop on Qualitative Vision, 1993, pp. 11–22.
[50] M. Sutton, L. Stark, K. Bowyer, Function from visual analysis and physical interaction: a methodology for recognition of generic classes of objects, Image and Vision Computing 16 (1998) 745–763.
[51] M.A. Sutton, L. Stark, K.W. Bowyer, Gruff-3: Generalizing the domain of a function-based recognition system, Pattern Recognition 27 (12) (1994) 1743–1766.
[52] M.A. Sutton, L. Stark, K. Hughes, Exploiting context in function-based reasoning. Sensor Based Intelligent Robots: International-Workshop, October 2000.
[53] L.M. Vaina, M.C. Jaulent, Object structure and action requirements: A compatibility model for functional recognition, International Journal of Intelligent Systems 6 (1991) 313–336.
[54] B.C. Vemuri, J.K. Aggarwal, Representation and recognition of objects from dense range maps, IEEE Transactions On Circuits and Systems 34 (11) (1987) 1351–1363.
[55] P.H. Winston, T.O. Binford, B. Katz, M. Lowry, Learning physical descriptions from functional descriptions, in: Proceedings of the National Conference on Artificial Intelligence, 1983, pp. 433–439.
[56] K. Woods, D. Cook, L. Hall, K. Bowyer, L. Stark, Learning membership functions in a function-based object recognition system, Journal of Artificial Intelligence Research 3 (1995) 177–222.
[57] K. Wu, M.D. Levine, Recovering parametric geons from multiview range data, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 21–23 June 1994, pp. 159–166.