Proceedings of Workshop on Vision for Robots in IROS'95 Conference, Pittsburgh, PA, 1995.

Experimental State of the Art in 3D Object Recognition and Localization Using Range Data

Avinash C. Kak and Jeff L. Edwards

Robot Vision Laboratory
1285 EE Building, Purdue University
West Lafayette, IN 47907-1285

Abstract

This paper discusses current state of the art research at the Purdue Robot Vision Lab in the area of 3D object recognition and localization using range data. We review three approaches to model representation: feature spheres, Local Feature Sets, and multiple-attribute hash tables. The incorporation of these three representational schemes into the MULTI-HASH bin-picking system has resulted in significant reductions in time complexity for scene-to-model hypothesis generation and verification. We also discuss the significant failure modes of the MULTI-HASH system, as well as this system's limitations regarding the recognition of complex industrial objects. Finally, we briefly discuss the potential of our current 3D recognition algorithms to real-world industrial applications.

1 Introduction

This paper discusses the current state of the art in experimental research at the Purdue Robot Vision Lab in the area of 3D object recognition and localization using range information as sensory input. Specifically, we address the recent progress in solving the bin-picking problem, in which a number of objects lie jumbled together in a pile heavily occluding each other, and the task of the recognition system is to determine both the identity and pose of the objects in the scene. This bin-picking task is normally performed in conjunction with some form of robotic assembly or manipulation process.

The state of the art in our laboratory is probably best described by the nature of the objects that our systems can handle. Our lastest systems can handle objects with arbitrary second-order surfaces; a typical bin scene containing such objects is shown in Fig. 1(a). These systems include the 3D-POLY system of Chen and Kak [3], a system based on bipartite matching [9], and our most recent system, MULTI-HASH, that also has the ability to learn object recognition strategies via interaction with a human [6]. We have also recently developed a system for Nippondenso Corporation (some aspects of which are described in [14]) for the recognition and localization of tubular objects of the kind shown in Fig. 1(b). In addition, our laboratory has developed a system for the highly specialized case of recognizing and singulating postal objects [13]. This last system allows objects to be defined generically, in the sense that the model definition of a rectangular parcel admits such parcels of arbitrary size, proportions, etc.

(a) (b)

Figure 1: *Objects of the types that are used for recognition experiments at the Purdue Robot Vision Lab. (a) A typical MULTI-HASH scene. (b) A typical scene from the Nippondenso experiments.*

Our aim in this paper is to review the salient points of the most sophisticated experiments we can do today with these systems. We will discuss the failure modes of these experiments and whether these limitations are inherent to the underlying methodology of the system, or are merely the result of implementational expediency. We will also delve into the future potential of these experimental systems with regard

to making robots more useful in the real world. We will focus in particular on MULTI-HASH, on account of this system being an outgrowth of both the 3D-POLY system [3] and the bipartite-matching system [9]. MULTI-HASH is computationally more efficient, and more general compared to our previous systems, in the sense that it can distinguish objects based on both geometric and non-geometric features (such as color.) For tubular objects, we will present the experiments that we can now perform with the system we have developed for Nippondenso.

Although low- and mid-level issues are important to any object recognition system, success at these levels is not sufficient for recognizing objects in complex scenes. Over the past several decades there has come into existence a body of tools that can usually be counted on to provide reliable extraction of object features, given reasonably high-quality sensory data. For example, Figs. 2(a) and (b) show the segmentations of range maps for the scenes in Figs. 1(a) and (b).

(a) (b)

Figure 2: *Feature segmentation results. (a) Segmented features from MULTI-HASH scene. (b) Segmented features from Nippondenso scene.*

So if we accept the premise that low- and mid-level processing steps can now be implemented robustly to yield relatively clean surface segmentations (at least for the class of objects shown here), the next challenge then is to efficiently and robustly match models to these extracted features. This problem is beset with combinatorial difficulties, since any attempt that seeks to compare every possible grouping of scene features with sets of model features involves exponential complexity. One of the major lessons learned during the last few years has been that this otherwise exponential computational burden can be ameliorated by using clever representational schemes for object models. As a case in point, the primary reasons for the reduced computational efficiency of the 3D-POLY approach [3] are the use of Local Feature Sets to organize the model and scene features, and the representation of each model object using a spherical data structure; the spherical data structure and the local feature set concept are discussed in Sections 2.1 and 2.2, respectively. As a result, the complexity of the 3D-POLY system for single object recognition is $O(n^2)$, where n is the number of features (analytically continuous surface patches) on the model object.

After reducing the complexity of object recognition for such objects to $O(n^2)$, our quest became one of investigating additional means of representing and organizing model and sensory data for further improvements in the combinatorial aspects of object recognition. Toward that end, it was interesting to note that the 3D-POLY system uses a hypothesize-and-verify approach in which $O(n)$ hypotheses are generated; verifying each hypothesis requires $O(n)$ time, resulting in $O(n^2)$ overall effort for the recognition of a single scene object. The computational burden incurred by the verification stage could not be reduced below $O(n)$ since, in the worst case, all the object features not used for hypothesis formation must be used for verification and, for large n, the number of such features approaches the total number of features. Consequently, only the hypothesis formation stage was open to complexity reduction; that is exactly what we have accomplished with MULTI-HASH. In the MULTI-HASH system, the use of a multiple-attribute hash table, discussed in Section 2.3, reduces the number of generated hypotheses even further. Using an interactive learning approach, MULTI-HASH generates a hashing function based on those feature attributes that are most powerful for disciminating between model features. The goal is to construct a hash table that will return only $O(1)$ hypotheses for a single scene object, thus resulting in an overall time complexity of $O(n)$.

We must hasten to add that our arguments above mentioned nothing about the dependence of complexity on the quantity of objects in a model library. If M denotes the number of distinct models in a library, then the recognition complexity of 3D-POLY becomes $O(Mn^2)$. By way of comparison, even when one takes the entire model library into account, the complexity of MULTI-HASH remains at $O(n)$, assuming that the hash-table is constructed in such a manner that each bin contains no more than a constant number of hypothses (something that is increasingly difficult to achieve as M becomes large.) Based on these results, it is apparent that proper model representation is the key to the design of computationally efficient object

recognition systems.

In the following section, we will briefly review the three key tools relevant to model representation and feature organization that have accounted for efficiency improvements achieved by MULTI-HASH: the spherical data structure and Local Feature Sets that were first introduced in 3D-POLY, and the multiple-attribute hash table. The first tool is most useful for verifying hypotheses, while the latter two provide efficient methods for hypothesis generattion.

In Section 3, we discuss the failure modes associated with MULTI-HASH, and whether they are consequences of implementational expediencies or inherent to the underlying model representational scheme. We argue that the system's primary limitation is that it is incapable of dealing with relatively complex model objects. Finally, in Section 4, we discuss issues related to the potential industrial applications of the bin-picking technologies that have recently been developed at the Robot Vision Lab.

2 Object Representation is the Key to Success

Although this paper restricts its discussion to experimental systems developed at the Purdue Robot Vision Lab, our work would not have been possible without building upon the advances introduced by other researchers in the field. Among the many different approaches that have been proposed for representing model objects, the three that have had the most direct influence on the design of our systems are the following:

- The local-feature-focus concept, first introduced in [1] and later extended to 3D object recognition in the 3DPO system [2], attempts to improve the efficiency of hypothesis generation by grouping local features into sets. This concept was the precursor of the Local Feature Sets, discussed in Section 2.2, that are used in both 3D-POLY and MULTI-HASH; The system described by Flynn and Jain [5] uses a similar notion for hypothesis generation.
- The extended Gaussian image, discussed by Horn in [7], maps the shape of an arbitrary 3D object onto the surface of a unit sphere. The resulting object representation is compact and facilitates the scene-to-model matching process. The extended Gaussian image representation is similar in many ways to the feature sphere data structure used in 3D-POLY and MULTI-HASH.
- The geometric hashing concept, first introduced by Lamdan and Wolfson [10], is another interesting approach to organizing feature data for efficient hypothesis generation, in which model identities and poses are inserted off-line into a hash table using affine-invariant features as keys. Although their exist important differences between the multiple-attribute hash tables used in MULTI-HASH and the original geometric hashing scheme proposed in [10], the two methods share a common goal: to efficiently retrieve a small number of the most promising scene-to-model match hypotheses for subsequent verification. The systems described in both [12] and [5] also use a form of hashing for this purpose.

The remainder of this section will discuss three examples of how model representation and feature organization schemes can be used to improve the efficiency of 3D object recognition from 3D data. In each of the three schemes, the efficiency gains are brought about by organizing the features in ways that take advantage of certain constraints to prune the search space.

2.1 The Feature Sphere Data Structure

Feature spheres are powerful data structures used for reducing the computational complexity incurred in the verification stage of object recognition. In this approach, a small number of extracted scene features are initially used to form a set of pose transformation hypotheses corresponding to possible model identities and poses for the scene object. These hypotheses are then verified or rejected by matching the attributes of the remaining scene features with those of the predicted model features. This sub-section focuses on the verification stage; sub-sections 2.2 and 2.3 address the efficiency of the hypothesis generation stage.

The verification stage will take $O(n^2)$ time if we exhaustively test each of the n scene features against each of the $O(n)$ model features. Instead, we desire a means of imposing some form of constraint on the model features in order to summarily reject most of them without consideration, so that only a small number (ideally one in the case of a correct match, or zero in the case of an incorrect hypothesis) must be tested against the scene feature. The feature sphere approach uses the concept of a principal direction to achieve this goal.

Essentially, the principal direction Φ represents the characteristic position or orientation (in an object-centered coordinate frame) of a feature (surface, edge, or vertex) with respect to the other features on the object, and is represented by a directional unit vector in 3-space. The seven types of features used to represent an object in both the 3D-POLY and MULTI-

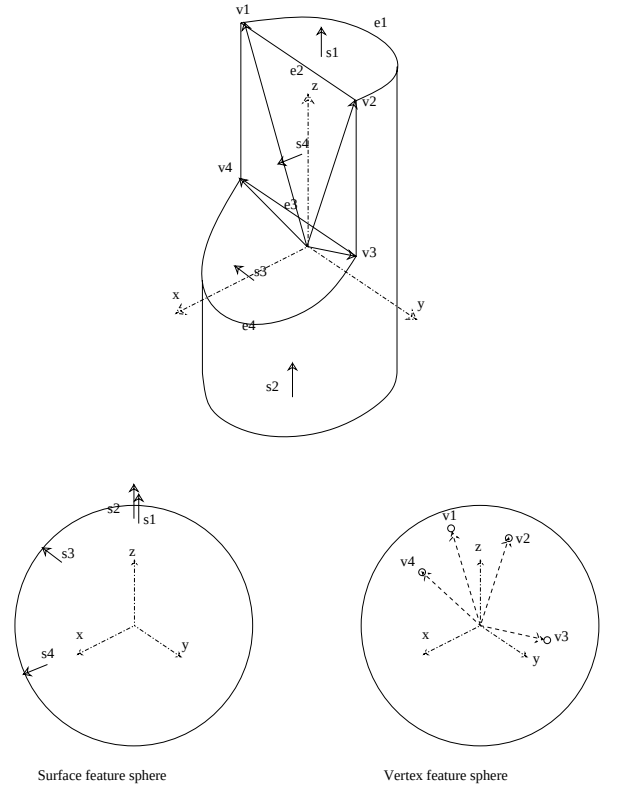


Figure 3: *A typical model object and the principal directions of its surfaces.*

HASH systems are: points; straight lines; elliptical curves; planar, cylindrical/conical, and spherical surfaces; and otherwise unclassified 2nd-order surfaces. The actual definition of Φ for a feature depends upon that feature's type. For example, the Φ of a planar surface is that surface's outward normal, the Φ of a cylindrical surface is the axis direction, and the Φ of a point is simply the normalized position vector of that point. Fig. 3 shows a typical object composed of distinct 2nd-order surfaces, and the principal directions for that object's surface and vertex features. The reader is referred to [3] for a more thorough treatment of principal direction.

The most useful property of principal direction is that it can be reliably extracted from structured-light range data for each of the seven types of surface feature. For example, even if only 40% of a cylinder's surface is visible in a range map, algorithms exist that will extract a decent estimate of that cylinder's axis direction in most cases. It should also be noted that since principal direction is defined only with respect to an object-centered coordnate frame, Φ can only be computed for a model feature, or for a scene feature that has been embedded in an object-centered coordinate frame through a pose hypothesis generated from previous scene-to-model feature matches.

Once a pose transform hypothesis has been formed, we can transform the pose of each of the extracted scene features into the object-centered coordinate frame and compute their principal directions Φ. We then seek to match these scene features with the model features in order to verify the hypothesis; a model and scene feature that do not share the same principal direction (within a certain tolerance) cannot possibly match; thus a feature's principal direction provides exactly the powerful constraint that we need to reduce the $O(n^2)$ complexity of the verification process.

The basic functionality of a feature sphere is straigtforward: you supply it a principal direction Φ (corresponding to some scene feature) and it gives you, in constant time, a list of zero or more model features whose principal directions match Φ (within a tolerance that you specify.) The scene feature must then be tested for a match only against the model features in this list, rather than the entire set of model features.

The feature sphere implements this behavior by tesselating the unit sphere into cells (or tessels) of arbitrary resolution, as shown in Fig. 4(a). During off-line model-building, a pointer to an attribute-value frame (describing geometrical and appearance characteristics) of each model feature is deposited in the tessels corresponding to the Φ for that feature. By laying the tessels flat, an (i, j, k) indexing scheme can be established, as shown in Fig. 4(b).

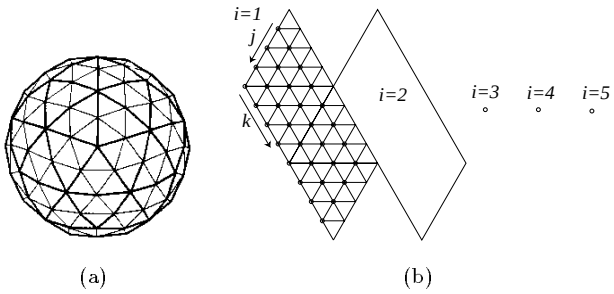


Figure 4: *(a) Tesselation of a unit sphere used in the feature sphere data structure. (b) The indexing scheme used for addressing the cells of the feature sphere.*

The value of the feature sphere lies in its ability to support two constant-time functions based on this indexing scheme. The first function is referred to as the tessel assignment function $L(\Phi)$, and returns the label (i, j, k) of the tessel corresponding to Φ. The second function is the find-neighbors function:

$$N(L_0) = L_1, L_2, \ldots, L_k$$

where $L_1, L_2, \ldots, L_k$ are the labels of the immediate neighbors of the tessel with label L_0. Together, these two functions allow constant-time access to a list containing only those model features whose Φ lie within a chosen tolerance of a scene feature's Φ. Again, the reader is referred to [3] for details concerning the nature of the indexing scheme and the implementation of the functions $L(\Phi)$ and $N(L_0)$.

If we assume that the principal directions for the features of a given model are distributed approximately randomly over the surface of the unit sphere such that there will be at most k features associated with any given Φ, then the worst-case time complexity for verifying a hypothesis will be $O(nk) = O(n)$. This use of feature spheres for rapid hypothesis verification is an example of how substantial reductions in computational complexity can be achieved by representing feature data in a way that takes advantage of constraints inherent to the problem at hand.

2.2 Local Feature Sets for Hypothesis Generation

Just as the feature sphere representation improves the efficiency of the verification stage, the use of Local Feature Sets (LFSs) can improve the complexity of the hypothesis generation stage. A similar approach was used in the 3DPO system [2] as well as the system described by Oshima and Shirai [11]. Local Feature Sets are employed in both the 3D-POLY and MULTI-HASH systems developed at the Robot Vision Lab.

Both of these systems are designed to operate with model objects composed of distinct 2nd-order surfaces. For such objects, the determination of a pose transformation requires that 3 scene features be matched to model features. Therefore, a brute-force approach to generating hypotheses might select 3 scene features and proceed to test every combination of these 3 features with each of the model features; for every combination in which the scene and model feature attributes were consistent, a hypothesis would be generated. Such an approach would take $O(n^3)$ time. We seek to reduce this complexity by again imposing constraints on possible match candidates, just as we did in the hypothesis verification stage.

When a pose transformation hypothesis is available, such as during the verification stage, then position and orientation attributes, such as the principal direction Φ, can provide powerful constraints for restricting the set of possible scene-to-model matches. Unfortunately, during the hypothesis generation stage, the principal direction of scene features cannot be used since a pose transformation is not yet available. In this situation, two other types of constraints, based on either shape or relational attributes, could perhaps be used instead Shape attributes include such properties as surface type (planar, cylindrical, etc.), radius, and area; a relational attribute could be a list of all adjacent feature labels. In our experiments, we have found that shape attributes are ineffective at constraining the hypothesis generation stage: either the attributes are viewpoint variant and hence unreliable, or they are viewpoint invariant but provide very little help in distinguishing between features. In constrast, we have found that relational attributes are quite useful for constraining scene-to-model matches in the absense of pose information.

The use of relational attributes for such purposes is based on the following observation: if one selects a group of scene features that possess a set of viewpoint-invariant relational attributes, then their corresponding model features must possess identical attributes. In particular, if one selects a set of adjacent scene surfaces, then their corresponding model surfaces must also be adjacent, barring any viewpoint-related artifacts (in real scenes, such artifacts do occur occasionally and tend to result in a minor degradation of system robustness.) Conveniently, the detection of adjacent surfaces in a scene is straightforward.

With this in mind, we will describe the concept of a Local Feature Set (for the 2nd order objects we have been dealing with) as a grouping of local features consisting of a vertex feature and the distinct surfaces features meeting at that vertex. Since a LFS will always be composed of at least three surfaces (and usually no more), a correspondence between a scene LFS and a model LFS is sufficient for determining a pose transformation hypothesis. (actually, such a correspondence can result in several distinct pose transformation hypotheses depending upon surface ordering see [3] for more details.) An object with n scene features will tend to have $O(n)$ vertices, and hence $O(n)$ Local Feature Sets.

The 3D-POLY system performs hypothesis generation by first selecting a LFS from the scene. This LFS is then exhaustively tested against each of the $O(n)$ model LFSs; if the features of a given scene-to-model LFS correspondence match well, then a hypothesis is generated. As a result, the complexity of the hypothesis generation stage is reduced from $O(n^3)$ to $O(n)$. This is because there are $O(n^3)$ possible group-

ings of three non-adjacent model features, but there are only $O(n)$ groupings of three adjacent model features. Essentially, the system never bothers to check the scene LFS against any of the $O(n^3)$ groupings of non-adjacent model features, whereas in the brute-force approach, such hopeless correspondences would not only be checked, but could also result in the generation of hypotheses. The substantial savings comes from representing models in terms of LFSs and thereby organizing the feature data to take advantage of the constraints embedded in the problem.

2.3 Multiple Attribute Hash Tables

The third and final example of how an appropriate model representation scheme can greatly influence the efficiency of an object recognition system is the multiple-attribute hash table used in MULTI-HASH [6]. The key improvement of MULTI-HASH over 3D-POLY lies in the improved hypothesis generating stage; 3D-POLY exhaustively tests all correspondences between scene and model LFSs; MULTI-HASH uses a hash table that hashes over a number of different LFS attributes, some of which are non-geometric in nature, in order to generate fewer possible scene-to-model LFS correspondences.

In many ways, the use of this hash table for retrieving a list of good model LFS candidates is similar to the use of the feature sphere for returning a list of candidate model features: you supply a scene LFS along with its computed attributes, and the hash table will return, in constant time, a list of zero or more model LFSs from which to form scene-to-model hypotheses.

By using a hash table, the hypothesis generation stage incurs only $O(1)$ cost and generates only $O(1)$ hypotheses, assuming that the hash table is constructed such that each bin contains at most a constant number of model LFSs. Therefore, the overall cost (including verification) is only $O(n)$. Furthermore, when the model library includes M different models, the complexity remains unchanged. Ofcourse, as M increases, the task of constructing a good hash table becomes increasingly difficult. In addition, it should be noted that complexity analyses such as this are based on several simplifying assumptions, and that the actual performance of the system is strongly dependent upon the nature of the model objects, especially when hash tables are involved.

During the automatic construction of the hash table, MULTI-HASH attempts so use viewpoint invariant attributes with maximum discriminatory power; some of these attributes may be non-geometric. In particular, the current implementation of MULTI-HASH makes use of color information as an important attribute for distinguishing between surface features. Fig. 1(a) shows a typical scene in which such color information is useful.

In addition to selecting highly discriminatory attributes, MULTI-HASH seeks to adaptively partition the hash table bins such that each bin contains at most one LFS (in the ideal case.) In order to facilitate this task, MULTI-HASH explicitly models the uncertainties associated with the LFS attributes; the uncertainty distribution for each attribute is modelled as a single-modal truncated Gaussian. These Gaussian distributions are determined by repeated sampling of the attribute values using an interactive learning tool. This tool, implemented on a Silicon Graphics workstation, allows the user to acquire sample sensory data for a model object and associate this data with the appropriate model features. It is often the case that, given a particular library of model objects, it is not possible to achieve the ideal case of one LFS per bin. Consequently, an optimal hash table is defined as the table with minimum average bin entropy; the reader is referred to [6] for details.

If we wished to use exhaustive search to generate an optimal hash table in which each attribute axis was discretized into k levels, then we would have to evaluate $O(2^{n \times \#attributes})$ different hash tables. In light of this fact, It is important to note that the structure of a hash table is equivalent to that of a binary decision tree, and that each decision node in the tree corresponds to a bin partition in the hash table. Furthermore, the problem of generating an optimal decision tree for minimization of classification error is NP-hard [8]. Consequently, there is no known method for generating an optimal hash table in less than exponential time. Therefore, MULTI-HASH uses a set of heuristics to generate a near-optimal decision tree; this decision tree is then converted into an equivalent hash table (conversion to a hash table allows $O(1)$ access time, versus the $O(\log n)$ access time of a decision tree.) Essentially, the heuristics used for decision tree generation make locally optimal decisions, and do not guarantee a globally optimal decision tree. However, our experiments have shown that good decision trees, and hence hash tables, are generated using our heuristic algorithm. Again, refer to [6] for details.

Once again, the proper organization of model features to take advantage of inherent constraints results in substantial computational savings. In the case of the multiple-attribute hash table, the representation imposes the constraint that scene LFS attributes will closely match their corresponding model attributes.

In other words, since a scene LFS containing a yellow surface will result in the hash table retrieval only of model LFSs also containing a yellow surface, the system never has to waste time by exhaustively testing at runtime all the model LFSs, only to find out that most of them are eliminated because they violate some attribute constraint (such as color.) Instead, MULTI-HASH performs these comparisons off-line. Finally, it should be noted that the feature sphere representation described in Section 2.1 is essentially a hash table in which only a single feature attribute, the principal direction, is used as the key. One major difference between the feature sphere and the multiple-attribute hash table is that the latter uses adaptive partitions, whereas the former uses a fixed tesselation of the attribute space that is much more susceptible to hot spots. Also, the hash table stores pointers to LFSs in its bins, whereas the feature sphere stores pointers to individual features.

3 Failure Modes and Limitations of our Systems

The previous section discussed efficiency issues relevant to the 3D-POLY and MULTI-HASH systems; this section addresses an issue of equal importance: robustness. In this section, when we refer to "2nd-order objects", we will mean objects, such as those shown in Fig. 1(a), composed only of distinct, well-defined 2nd-order surfaces (usually geometric primitives such as planes, cylinders, etc.) that can be described by a single attribute frame. This section will cover two important topics: the reasons why MULTI-HASH sometimes fails to recognize such 2nd-order objects, and the limits to which the MULTI-HASH system can be extended to achieve recognition of more complex objects.

To get a rough idea of the robustness of MULTI-HASH, ten random scenes similar to that in Fig. 1(a) were exposed to the system; in seven of those scenes, MULTI-HASH successfully recognized at least one object. Experiments have shown that there are two root causes for this 30% failure rate: improper surface segmentations, and the occasional inability to extract a vertex-centered LFS from the scene due to the particular orientation of objects (without a scene LFS, hypothesis generation cannot occur.) Since the square and round model objects used for our experiments, shown in Fig. 1(a), contained only 12 and 4 LFSs, respectively, it is quite common for a random scene to present no convex vertices to the sensor (especially after segmentation errors are included.)

We suspect that the improper segmentation problem could be greatly alleviated by two minor improvements to the MULTI-HASH implementation. First, the structured light scanner used by MULTI-HASH is an inexpensive apparatus that is quite adequate for proof-of-concept work, but it has not been optimized for performance. Fig. 5 shows an example of the range data acquired by the sensor from the scene in Fig. 1(a), and illustrates that the range maps generated by this scanner leave room for improvement in terms of resolution, signal-to-noise ratio, and occlusion. Secondly, although the surface segmentation algorithms currently used as a pre-processing stage by MULTI-HASH achieve impressive results given the quality of the range data they have to work with (see Fig. 2(a),) they are still prone to occasional mistakes, such as over-segmenting surfaces or merging two distinct surfaces together. With more expensive sensors, and an even better segmentation algorithm (that perhaps apples model-specific knowledge to the problem), the resulting segmentation maps could be substantially improved.

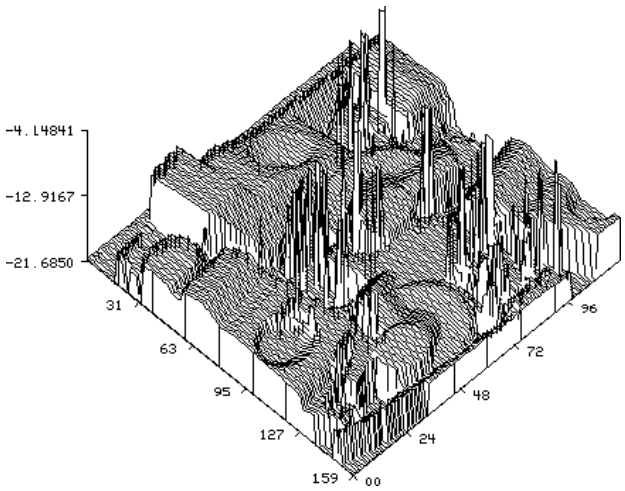


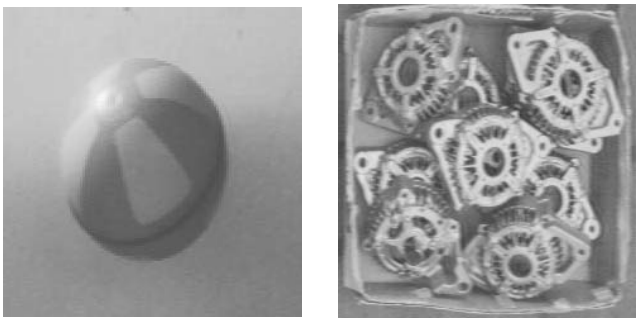
Figure 5: *Range map showing the 3D location of detected points from the scene in Fig. 1(a).*

The second root cause of failure is also a consequence of implementational expediencies. As discussed in Section 2.2, the purpose of a LFS is to group individual features together in such a manner that powerful relational constraints can be used to limit the number of scene-to-model match candidates. The particular instantiation of the LFS concept that is implemented in both 3D-POLY and MULTI-HASH requires a convex vertex and the surrounding surfaces for LFS formation. However, this restrictive defini-

tion of a LFS is not fundamental to MULTI-HASH in any way; it was simply a convenient implementational shortcut. By using a less restrictive class of LFSs that still impose relational constraints of equal or greater power, the second failure mode should be greatly reduced as well.

Therefore, we believe that for the domain of 2nd-order objects, the failure modes exhibited by MULTI-HASH are primarily consequences of implementational shortcuts rather than fundamental limitations of the approach; in a real industrial application, the robustness of MULTI-HASH could most likely be raised to a much higher level.

On the other hand, as one moves beyond the relatively simple 2nd-order objects used in our current experiments toward more complex entities of industrial interest, the MULTI-HASH system does indeed begin to show fundamental limitations that are not likely to be alleviated through minor enhancements. As a simple example, consider the multi-colored ball shown in Fig. 6(a). MULTI-HASH can easily represent this object's shape as a single spherical surface with a certain radius attribute. Unfortunately, the color attribute of this single surface is not homogenous and cannot be represented by a single RGB triple. Instead, the representation of the ball's surface appearance must somehow take into account the two distinct color regions.



(a) (b)

Figure 6: *(a) A simple example of a spherical object that the MULTI-HASH representational scheme is incapable of describing. (b) A jumbled bin of alternator housings present a very challenging recognition problem in terms of model representation.*

Although the example of Fig. 6(a) is almost trivial, it demonstrates the fundamental limitation of the current MULTI-HASH system: the model representation scheme does not possess the flexibility to deal with the arbitrary shape or surface appearance characteristics that are often found on complex objects. Once again, proper model representation plays a crucial role in the success or failure of object recognition systems. A more compelling example of this premise are the cast aluminum alternator housings shown in the scene of Fig. 6(b). The geometric complexity of the alternator objects are representative of a wide variety of industrial objects. The most noteworthy property of the alternator object is that it cannot be represented simply as a collection of distinct primitive surfaces. Rather, it basically consists of a smooth, continuous surface with complex topology. How can one effectively represent this surface for use in a recognition task? Clearly a MULTI-HASH-like segmentation into distinct, well-defined surfaces would seem inappropriate. Furthermore, how can the various non-geometrical attributes of the alternator's surface, such as its texture variations, be represented if the surface itself is continuous and unsuitable for segmentation into distinct regions? This object seems to present a very difficult representational problem, yet a humans routinely recognize and grasp such objects almost effortlessly.

The problem then is to move beyond the restrictive model representation schemes used by MULTI-HASH toward a more general and flexible approach capable of dealing with complex objects such as the alternator housing, *while still attempting to retain the efficiencies exhibited by current data structures*, such as the feature sphere, Local Feature Sets, and multiple-attribute hash tables. In particular, we would like the more general and flexible model representations of the future to continue to take maximum advantage of the constraints inherent to the problem at hand.

Some of the most promising work along this direction has been done by Medioni. A representational scheme suitable for arbitrary object geometries that uses a form of hashing to preserve some measure of efficiency is described in [12] More recently, [4] presents a system for generating model descriptions of complex objects using range data. Their approach is one of several "inflating balloon" methods that proceed to inflate a fine mesh representation, starting at a point in 3-space corresponding to the interior of the object, and continuing until the mesh collides with the registered range data associated with the object's surfaces. The Robot Vision Lab has recently begun work on a model-building scheme similar in concept to this inflating balloon approach.

4 Potential for Industrial Applications

This section will briefly address the potential applications of the state of the art bin-picking systems that have been developed at the Purdue Robot Vision Lab. In particular, we will focus on two systems: a recognition system developed for the Nippondenso Corporation, and the MULTI-HASH system for recognizing objects composed of 2nd-order surfaces.

The Nippondenso system uses range data, obtained from a structured light sensor, to identify and locate heater tubes from a jumbled bin. Following recognition, these tubes are to be individually grasped and positioned for welding with the other components of the heater assembly. The tubes are approximately 0.5 inches in diameter and consist of two or more straight segments connected by joints; there are several different tube configurations, depending upon the segment lengths and their adjoining angles. Fig. 1(b) shows a typical scene. Some of the salient points regarding the design of this system are discussed in [14].

The model representation scheme is quite simple and is specific to the recognition of tubular objects only. Essentially, a tube model is stored as a sequence of linear segment lengths along with their adjoining angles. Following the acquisition of the range data, a sequence of low-level processing operations is performed. The result is the set of segmented features shown in Fig. 2(b). These features are then skeletonized and fit to line segments, which are then matched with the tube models. In a typical scene, between 1 and 3 complete model matches are usually recognized. In addition, between 6 and 10 partial matches are usually found, in which a tube fragment is identified, but not enough of it's segments are matched to completely determine its pose. Partial matches provide enough information to successfully grasp the tube and place it on a light table for complete pose determination.

In this system, the non-optimized scanning and recognition times are approximately 3 and 7 seconds, respectively. In terms of robustness, our experiments have demonstrated that the system invariably finds at least one graspable partial tube fragment in each scene, and only occasionally fails to find at least one complete tube. The current performance of the Nippondenso system is near the level neccesary for industrial implementation. In addition, we feel that with an improved sensor and limited optimization work, both the efficiency and robustness of the system could be further enhanced. In fact, the system is currently undergoing evaluation testing by Nippondenso engineers for installation in an actual manufacturing line in Japan.

In contrast to the recognition results that the Nippondenso system has achieved in the case of very simple tubular objects, the performance of the MULTI-HASH system is still inadequate for factory use. However, with the proper enhancements (discussed in Section 3) for alleviating the two primary failure modes of the system, as well as a substantial amount of optimization work, we believe that it is quite conceivable that the MULTI-HASH approach could find use in an industrial setting today. Of course, the effectiveness of MULTI-HASH has only been demonstrated for relatively simple, 2nd-order objects of the type shown in Fig. 1(a). Unfortunately, the majority of industrial objects of interest are considerably more complex in nature, such as the alternator housing presented in Section 3. Bin-picking of such objects is not currently feasible, nor is it likely that the current experimental systems and methods in use at the Robot Vision Lab may be easily extended or enhanced to achieve such capabilities. The primary obstacle to the efficient recognition of such objects lies in the development of model representation schemes that provide the flexibility to describe complex features yet still take sufficient advantage of problem-specific contraints to retain adequate efficiency.

To summarize the potential for industrial applications of the bin-picking technology recently developed at the Robot Vision Lab, we divide the domain of all object types into three broad categories of increasing complexity. Simple geometric objects of homoegenous surface appearance, such as the heater tubes, can now be recognized at a level of performance suitable for industry. Another class of simple objects in this category would include standard rectangular cardboard boxes used in packaging; for example, it is probably safe to say that the Robot Vision Lab's current structured-light technology is sufficient for the factory implementation of a cardboard box palletizing/depalletizing system. Objects of the second category are comprised of distinct 2nd-order surfaces; the current bin-picking algorithms of the Robot Vision Lab, specifically the MULTI-HASH system, are probably sufficient for limited factory implementation with regards to this class of objects. Finally, for objects that are too complex to be described using a set of distinct 2nd-order surfaces, such as the alternator housings of Section 3, even laboratory experiments (much less indusrial-grade systems) demonstrating efficient bin-picking are still well beyond the reach of the state of the art technology,

5 Conclusions

The state of the art for recognizing 3D objects from 3D data has undergone a marked improvement during the last decade. It is now possible to design systems that can handle non-convex and non-polyhedral objects and do so with a good measure of robustness and speed on ordinary computing hardware. In this paper, we reviewed some of the salient aspects of these systems designed at the Purdue Robot Vision Lab. In particular, we have emphasized the influence that proper object representation and feature organization can play on recognition efficiency. Through our discussions of the feature sphere data structure, Local Feature Sets, and multiple-attribute hash tables, we have provided three examples of how problem-specific constraints can be embeddded into an object representation to yield significant reductions in the time complexities associated with hypothesis generation and verification.

Furthermore, we have argued that the two principal failure modes associated with the MULTI-HASH system are consequences of implementational expediencies rather than fundamental weaknesses of the approach. We have also pointed out that the MULTI-HASH approach is inadequate for recognizing many complex industrial objects, such as alternator housings. This limitation is again directly related to the method of object representation used by MULTI-HASH, which is too rigid to describe many objects whose shapes cannot easily be segmented into distinct 2nd-order surfaces with homogenous attributes.

Finally, we have discussed the potential for implementing our algorithms in industrial settings. We believe that our current algorithms are adequate for the industrial-grade recognition of tubular entities and objects of a fairly simple nature (i.e., composed of distinct 2nd-order surfaces), but that the recognition of more complex objects is still well beyond our capabilities.

References

[1] R. Bolles and R. Cain, "Recognizing and locating partially visible objects: The local-feature-focus method," *Intl. Journal Robotics Research*, Vol. 1, No. 3, pp. 57-82, 1982.

[2] R. Bolles and P. Horaud, "3DPO: A three-dimensional part orientation system," *Intl. Journal Robotics Research*, Vol. 5, No. 3, pp. 3-26, 1986.

[3] C. Chen and A. Kak, "A robot vision system for recognizing 3-D objects in low-order polynomial time," *IEEE Trans. Systems Man Cybernetics*, Vol. 19, No. 6, pp. 1535-1563, 1989.

[4] Y. Chen and S. Medioni, "Description of complex objects from multiple range images using an inflating balloon model," *Computer Vision and Image Understanding*, Vol. 61, No. 3, pp. 325-334, 1995.

[5] P. Flynn and A. Jain, "3D object recognition using invariant feature indexing of interpretation tables," *CVGIP: Image Understanding*, Vol. 55, No. 2, pp. 119-129, 1992.

[6] L. Grewe and A. Kak, "Interactive learning of a multiple-attribute hash table classifier for fast object recognition," *Computer Vision and Image Understanding*, Vol. 61, No. 3, pp. 387-416, 1995.

[7] B. K. P. Horn, "Extended Gaussian Image", *Proceedings of IEEE*, Vol. 72, No. 12, pp. 1671-1686, 1984.

[8] L. Hyafil and R. Rivest, "Constructing optimal binary decision trees is NP-complete," *Inform. Process. Lett.*, Vol. 5, No. 1, pp. 15-17, 1976.

[9] W. Kim and A. Kak, "3-D object recognition using bipartite matching embedded in discrete relaxation," *IEEE Trans. Pattern Analysis Machine Intelligence*, Vol. 13, No. 3, pp. 224-251, 1991.

[10] Y. Lamdan and H. Wolfson, "Geometric hashing: A general and efficient model-based recognition scheme," in *Proc. IEEE Int. Conf. Robotics and Automation*, Philadelphia, PA, Apr. 1988, pp. 1407-1413.

[11] M. Oshima and Y. Shirai, "Object recognition using three-dimensional information," *IEEE Trans. Pattern Analysis Machine Intelligence*, Vol. 5, No. 4, pp. 353-361, 1983.

[12] F. Stein and G. Medioni, "Structural indexing: Efficient 3-D object recognition," *IEEE Trans. Pattern Analysis Machine Intelligence*, Vol. 14, No. 2, pp. 125-145, 1992.

[13] A. J. Vayda and A. C. Kak, "Robot vision system for recognition of generic shaped objects," *CVGIP: Image Understanding*, Vol. 54, No. 1, pp. 1-46, 1991.

[14] S. H. Wang, R. L. Cromwell, A. C. Kak, I. Kimura, M. Osada, "Model-based vision for robotic manipulation of twisted tubular parts: Using affine transforms and heuristic search," in *Proceedings of IEEE Interational Conference on Robotics and Automation*, 1994.