3D Object Recognition: Interpretation Tree Search on a MIMD Machine *

Timothy S. Newman Anil K. Jain Richard J. Enbody

Michigan State University, Department of Computer Science
East Lansing, MI 48824 USA
email: newman@cps.msu.edu

Abstract

This paper presents a parallel implementation of a tree search strategy for recognition of three-dimensional objects in range images. A coarse-grained and a medium-grained implementation of the tree search algorithm is presented and analyzed. The performances of the two implementations are analyzed on a 96 node BBN GP-1000 and on a 48 node BBN TC-2000, and, for comparison, on a Sun SPARCstation 2.

1 Introduction

Automatic identification and localization of 3D objects from a scene is a major research issue in computer vision. The 3D object recognition problem has been tackled using many different approaches, most of which use some form of search technique to match objects to models (e.g., [3, 5]). Nearly all of the 3D object recognition systems have been implemented on sequential machines. In spite of the different forms of sensory data and visual processing paradigms that have been utilized, there is no generally acceptable approach to 3D object recognition.

This paper examines one popular high-level computer vision paradigm, the interpretation tree (IT) search [5, 6], for matching observed scene features to known model features. The interpretation tree paradigm is a method that enumerates all possible mappings between features on a model with features derived from the sensed image of the model. A variety of constraints and heuristics are commonly used to prune the tree in order to reduce the computational complexity of the IT search.

Video, or intensity, images gathered by standard CCD cameras have been a popular data source from which 3D objects have been recognized. But deriving 3D structures from a single 2D image continues to be a difficult and challenging problem. To circumvent this problem, computer vision researchers have directly acquired range images. In our experiments, a triangulation-based range sensor, the Technical Arts 100X White Scanner in the Pattern Recognition and Image Processing (PRIP) laboratory at Michigan State University, was used to acquire range images.

*Research was supported in part by the National Science Foundation under Grant CDA-88-06599.

Parallelism in Object Recognition

Object recognition has been pursued most often on sequential machines. Many low-level vision tasks such as image convolution are ideally suited for implementation on SIMD machines like the Connection Machine or on systolic architectures [2]. Parallel approaches to higher level computer vision tasks such as matching or verification have received far less attention. One exception is Bhanu and Nuttall [1], who have implemented a quadric surface classification algorithm on an 18-node BBN Butterfly.

Interpretation tree search is one high-level vision task that is particularly promising for parallel implementation. This paper presents a parallel technique for matching image objects to models using the interpretation tree matching paradigm. The technique has been implemented on a 96 node BBN GP-1000 at Michigan State University and on a 48 node BBN TC-2000 at Argonne National Laboratory. For comparison, we implemented the same technique on the SPARCstation 1 and 2. Results are presented for two implementations having different levels of granularity. A coarse-grained implementation achieved relative speedups of 4 or less and a medium-grained implementation achieved relative speedups of 9 to 17 for the test objects.

The GP-1000 has 96 M68020 processing elements, each of which has 4MB of memory that can be accessed by all processors over an interconnection network. The TC-2000 has 48 M88000 processing elements, each with 16MB of memory. Programs written using BBN's Uniform System for parallel programming are portable between these machines. Interpretation tree search seems suitable for implementation on these MIMD machines because nodes of the tree can be mapped onto different processors, each of which has enough power and memory to independently evaluate the pruning constraints on the relevant data.

2 Interpretation Tree Search

Each node in the interpretation tree represents an association between one scene and one model feature. The features in our interpretation tree are 3D object surfaces. Object recognition can be viewed as matching an object's features against an interpretation forest consisting of a tree for each object model in the database. The set of valid

0-8186-2910-X/92 $3.00 © 1992 IEEE

mappings from the object features to the corresponding model features that survive the pruning constraints form a collection of hypotheses indicating which object is present in the scene. Since each valid mapping contains evidence for the position of object features vis-à-vis model features, the interpretation tree is useful for both object recognition and pose determination. In practice, the interpretation tree usually has to be at least four levels deep to allow accurate estimation of pose.

A complete matching between all image surfaces and corresponding model (or NULL association) surfaces is considered an hypothesis which must be verified. The verification step used was to compute the total angular difference between the mean surface normals of sensed object surfaces with the estimated surface normals of the transformed model.

We have concentrated on the performance of parallel interpretation tree search on polyhedral objects. Characteristics of each polyhedral facet were used as pruning constraints; a match between image surface i and model surface j was only considered if the constraints were satisfied. The facet constraints used were surface area, surface adjacencies, angles of intersection between surfaces, visibility attributes, and vertex angles of face vertices. Surface area and vertex angles can be considered to be unary constraints. The other features are binary constraints that capture inter-surface relationships.

3 Images and Models

The experiments reported in this paper were conducted on images of three polyhedral objects. These objects are labeled as Block1, Block2, and Block3. A range image of Block1 is displayed as a pseudo-intensity image in Figure 1. Images of Block2 and Block3 are shown using similar renderings in Figures 2 and 3, respectively. Experiments were carried out on two real range images of Block1, on five real range images of Block2, and on one synthetically generated range image of Block3.

The segmentation of the range images and extraction of resulting surface features were done off-line on a SUN SPARCstation 1 or 2. Our segmentation scheme is based on the algorithm first proposed by Hoffman and Jain [8] and later modified by Flynn and Jain [3]. This segmentation performs a clustering of surface normals and (x, y, z) coordinates at every pixel. A surface segmentation of the range image of Figure 2 is shown in Figure 4. Various features of the surface patches were then extracted and downloaded to the Butterfly.

The object models were also generated off-line. The I-DEAS CAD package's GEOMOD solid modeler was used to build the CAD models of these objects. GEOMOD creates a constructive solid geometry (CSG) representation, a boundary representation (B-rep), and a polyhedral representation of the objects. The polyhedral representations of the Block1, Block2, and Block3 objects, built by Flynn and Jain [4], were used for the work presented here.

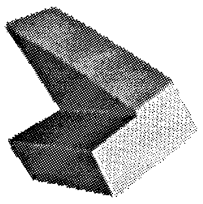

Figure 1: Block1 range image.

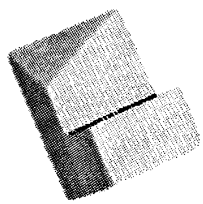

Figure 2: Block2 range image.

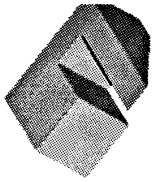

Figure 3: Block3 range image.

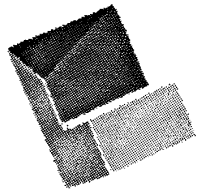

Figure 4: Segmentation of range image in Figure 2.

4 Methodology and Experiments

Two implementations, one of coarse granularity, and one of medium granularity, were implemented using the BBN C Uniform System. Each program consists of about 1700 lines of code.

4.1 Coarse-Grain Parallelism

Our first experiment was to test a coarse-grain parallelism, where one task was generated for each node at the top level of the interpretation tree. Each task performed a recursive depth-first search on its own subtree. Subtrees were evaluated in a top-down manner, since if any constraints were violated at a tree node, then it was unnecessary to examine any descendants of the node.

Speedup graphs for this approach applied to the Block2 images are shown in Figure 5. All speedups shown in this paper are from timings on the BBN GP-1000. Similar results were achieved on the TC-2000, although the speedups were always slightly lower on that machine. The total number of hypotheses to be verified for Block2 depended on the specific range image and varied from 12 to 229 for the five Block2 images tested.

The speedups observed were acceptable, especially for Image 5 which had a linear speedup for clusters of five or fewer processing elements, but not impressive—we never observed a speedup of greater than 4. This is because early pruning of the tree resulted in most of the processors running out of work relatively quickly. A few of the processors, particularly the one searching the NULL branch, executed lengthier searches and generated the majority of the matchings. In fact, the speedup could never be greater than $n + 1$, where n is the number of model surfaces, since

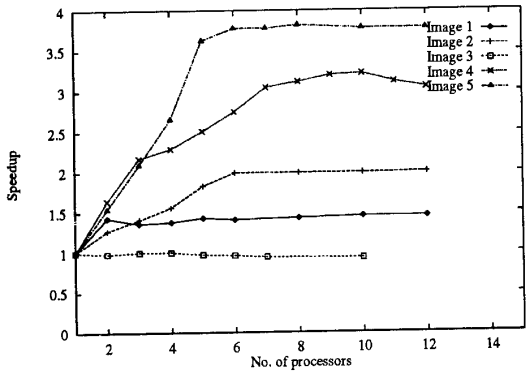

Figure 5: Speedups for Coarse-Grain Parallelism for Interpretation Tree Search for Block2 on the GP-1000.

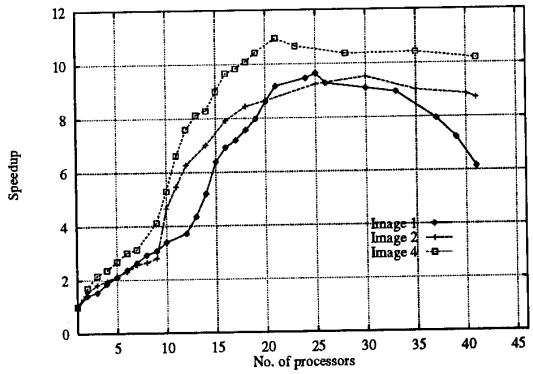

Figure 6: Speedups for Medium-Grain Parallelism for IT Search on Block2 Images 1, 2, 4 on the GP-1000.

only $n + 1$ tasks will be generated. The sublinear behavior may also reflect the strength of the pruning constraints or that very strong evidence exists for the presence of only a few of the model surfaces.

All timings and speedups reported here consider only the IT search, verification, and calculation of the transformation that maps the model onto the sensed object. The time for image input, building the internal object models, and initializing the Uniform System were not included.

4.2 Medium-Grain Parallelism

Our second experiment sought to balance the tasks more evenly among the processing elements. IT search was implemented in a medium-grain parallelism by mapping each node of the search tree onto a processor that evaluated the pruning constraints for that tree node. Tasks were generated in a top-down fashion, so matchings were only attempted for image surface i after image surfaces $1, 2, ..., i - 1$ had been matched to some model surfaces. Figure 6 shows the speedup graphs for three of the Block2 images.

Figure 7 shows the speedup graphs for the Block1 images, the other two Block2 images, and for the Block3 image. Twenty-four hypotheses were generated for Block1 Image 1 and six hypotheses were generated for Block1 Image 2. Block2 Image 5 and Block3 Image 1 achieved the greatest speedups, probably because they contained more surfaces than the other images. (In fact, Block2 Image 5 was over-segmented by the segmentation scheme.) The large number of surfaces resulted in a deeper and wider interpretation tree, thus creating more tasks for the team of processors. All of the speedup graphs exhibit slightly sublinear behavior, although the region in the graph between 10 and 20 processing nodes is approximately linear. These results do not reflect any finetuning of the implementation. Through more careful coding, perhaps speedup could be ten to twenty percent higher.

Graphs of the execution times on the GP-1000 are shown

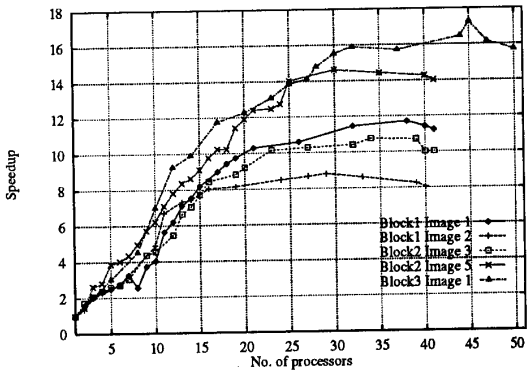

Figure 7: Speedups for Medium-Grain Parallelism for IT Search on Block1, Block2 Images 3 and 5, and on Block3 Image 1.

in Figure 8. Block2 Images 1, 2, 3, and 4 exhibited minimum times of between 0.08 and 0.23 seconds. Image 5 took a minimum of 0.95 seconds to execute.

Another set of experiments was conducted that consisted of executing the search and verification without producing any output. The speedups were roughly the same as those observed above, although execution times were reduced by at least twenty percent. For Block2 Image 3, execution time for a single processing element was reduced from 1.37 seconds to 0.43 seconds and the time for 12 processing elements was reduced from 0.25 seconds to 0.08 seconds.

For comparison, we have implemented our algorithm on the GP-1000, TC-2000, and on a SPARCstation 1 and 2. The algorithm ran in 1.76 seconds on a SPARCstation 2 for Block2 Image 5. On the GP-1000, the same search executed in 1.22 seconds with 20 processors and in 0.98 seconds using 40 processors. It executed in 0.42 seconds using 24 processing elements of the TC-2000. In gen-

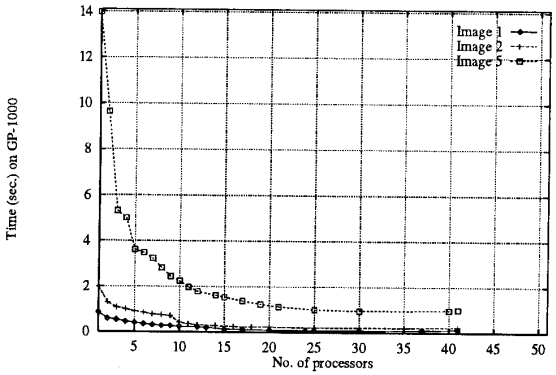

Figure 8: Time for Medium-Grain Parallelism for IT Search on Block2 Images 1, 2, and 5 on the GP-1000.

Machine	CPU Time
SPARC 1	4.05 s
SUN 4/390	2.84 s
SPARC 2	1.76 s
GP-1000	0.98 s
TC-2000	0.42 s

Table 1: CPU times for IT search for Block2 Image 5 on various architectures.

eral, the GP-1000 performed IT search twice as fast as the SPARCstation 2. The TC-2000 was approximately 4-4.5 times faster than the SPARC 2. The various execution times are shown in Table 1.

Gustafson has suggested that speedups should be measured using a bounded speedup, in which the total amount of time a process takes to execute is kept constant by scaling up the work as the number of processors increases [7]. Potentially, we could scale up the amount of work by using models or images which have more surfaces or by using different constraints, although this would necessitate measuring the scale-up in work.

5 Conclusions and Future Directions

Interpretation tree search is a technique that is well-suited to parallel implementation. We explored a coarse-grain and a medium-grain implementation of the interpretation tree search on the BBN Butterflys. Speedups of less than 4 were observed for the coarse-grain implementation and speedups between 9 and 17 were achieved for the medium-grain implementation. The speedups on the TC-2000 were slightly lower, although, as expected, both the Butterflys performed the IT search faster than a SPARCstation2. The system developed here presents empirical evidence that the search procedure can be done quickly and fairly efficiently by an MIMD machine having 10 to 20 processing elements.

There are several extensions to this project which are being pursued. We are fine-tuning our implementation and exploring the effects of memory contention on speedup. We are also examining how different constraints affect speedup and minimum execution time for interpretation tree search. Weaker constraints would result in a deeper search tree, creating more work for the processors and hopefully greater speedups. This probably would not be detrimental since the processing elements are currently underutilized.

Finally, we are developing implementations of interpretation forest search to allow matching of scene objects to multiple models. We believe this scale-up in work should deliver greater speedups for clusters of 20 or more processors since our current IT search does not fully utilize large clusters of processors.

Acknowledgments

Access to the TC-2000 was provided by the Advanced Computing Research Facility, Mathematics and Computer Science Division, Argonne National Laboratory. We are grateful to Mr. Narayan S. Raja for his helpful suggestions.

References

[1] Bir Bhanu and Lawrence A. Nuttall, "Recognition of 3-D Objects In Range Images Using a Butterfly Multiprocessor," *Pattern Recognition*, Vol. 22, No. 1, 1989, pp. 49-64.

[2] Alok N. Choudhary and Janak H. Patel, *Parallel Architectures and Parallel Algorithms for Intergrated Vision Systems*, Kluwer Academic: Boston, 1990.

[3] Patrick J. Flynn and Anil K. Jain, "BONSAI: 3-D Object Recognition Using Constrained Search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, 1991, pp. 1066-1075.

[4] Patrick J. Flynn and Anil K. Jain, "CAD-Based Computer Vision: From CAD Models to Relational Graphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, 1991, pp. 114-132.

[5] W. Eric L. Grimson and Tomàs Lozano-Perèz, "Model-Based Recognition and Localization from Sparse Range or Tactile Data," *International Journal of Robotics Research*, Vol. 3, No. 3, Fall 1984, pp. 3-35.

[6] W. Eric L. Grimson, "The Combinatorics of Heuristic Search Termination for Object Recognition in Cluttered Environments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, No. 9, 1991, pp. 920-935.

[7] J. Gustafson, "Reevaluating Amdahl's Law," *Communications of the ACM*, Vol. 31, 1988, pp. 523-533.

[8] R. Hoffman and A. K. Jain, "Segmentation and Classification of Range Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 9, 1987, pp. 608-620.