CAD-Based Robotics

Thomas C. Henderson, Eliot Weitz, Chuck Hansen,
Rod Grupen, C.C. Ho and Bir Bhanu

Department of Computer Science
The University of Utah
Salt Lake City, Utah 84112

Abstract

We describe an approach which facilitates and makes explicit the organization of the knowledge necessary to map robotic system requirements onto an appropriate assembly of algorithms, processors, sensors, and actuators. In order to achieve this mapping, several kinds of knowledge are needed. In this paper, we describe a system under development which exploits the Computer Aided Design (CAD) database in order to synthesize:

- recognition code for vision systems (both 2-D and 3-D),
- grasping sites for simple parallel grippers, and
- manipulation strategies for dextrous manipulation.

We use an object-based approach and give an example application of the system to CAD-based 2-D vision.

1. Introduction

The overall objective of this project is to explore the use of various kinds of knowledge, and in particular CAD (Computer Aided Design) representations and models as a basis for the visual recognition and manipulation of objects for robotic applications. Thus, CAD-based robotics applies to real-world analysis and manipulation of workpieces, including Computer Aided Manufacturing (e.g., loading NC machines, process control, etc.), Computer Vision for Inspection, as well as assembly. We have developed techniques and algorithms which allow the interactive and automatic generation of computer vision representations and geometric models of complicated realizable 3-D objects in a systematic manner. These representations and models are obtained using available Computer Aided Geometric Design (CAGD) techniques. We have also developed algorithms and reasoning techniques which permit the automatic synthesis of recognition strategies. An object-based approach is used (built on COMMON Lisp), and these objects are then executed to analyze a scene. We describe a 2-D inspection application below.

This work is a step forward in the direction of bridging the gap between the fields of CAGD and computer vision in so far as the issues related to representation and modeling of 3-D objects are concerned. This work provides a fundamental understanding of the requirements and the role of boundary/surface models in computer vision. It provides a systematic way of building object models, as opposed to the *ad hoc* techniques currently in use. These models can be used in the recognition of objects in 2-D images for tasks such as photo interpretation, navigation and guidance. We are also exploring their use for finding the orientation and position of 3-D objects in space for their manipulation by robots. The long-term goal is the automated assembly of objects with parts designed and manufactured using the knowledge in a CAGD system.

Studying such issues will also take us a step closer to computer aided prototyping. The rapid design of embedded electromechanical systems is crucial to success in manufacturing and defense applications. In order to achieve such a goal, it is necessary to develop design environments for the specification, simulation, construction and validation of multisensor systems. Designing and prototyping such complex systems involves integrating mechanical parts, software, electronic hardware, sensors and actuators. Design of each of these kinds of components requires appropriate insight and knowledge. This in turn has given rise to special computer-based design tools in each of these domains. Such Computer Aided Design (CAD) systems have greatly amplified the power and range of the human designer. To date, however, it is still extremely difficult to address overall system issues concerning how the components fit together, and how the complete system will perform.

It is crucial to develop a design environment in which these multiple facets of system design can take place in a coordinated way such that the description of one component can be easily interfaced to another component, even when they are radically different kinds of things (e.g., a control algorithm, a mechanical linkage and an actuator). The designer should have the freedom to try out ideas at different levels of detail; i.e., from the level of a sketch to a fully detailed design. Figure 1 shows the general set of components required in the system we envision. The use of the CAD database provides part of the solution to developing such an environment.

The system described in Figure 1 consists of four major components:

1. **the Multisensor Knowledge System (MKS)** - this specifies the knowledge of all available algorithms, processors, actuators, sensors which can be used in system construction; in addition, other knowledge, such as environmental constraints (e.g., lighting, obstacles, etc.) can also be specified,
2. **the Computer Aided Geometric Design (CAGD) System** - this is an interactive design system which provides geometric design and analysis capabilities,
3. **the Requirements Specification Interface** - this is essentially a primitive task specification language for the user to identify the task (e.g., 2-D visual inspection for structural defects), and
4. **the Application Specific Rules** - these rules take as input the task requirements specification and use the knowledge in **MKS** and the CAGD system to synthesize a system which satisfies the requirements. Thus, the rules themselves encode domain-specific knowledge for the application of interest.

Finally, the synthesized systems are packaged as "Logical Sensors" and are then available in the **MKS** system as future system

[1]This work was supported in part by NSF Grants MCS-8221750, DCR-8506393, and DMC-8502115.

resources. For example, an edge detector may become a separate Logical Sensor.

2. CAD-Based Robotics

In this section, we describe the components of our CAD-Based Robotics approach. The two major components are the Multisensor Knowledge System (**MKS**) and the CAGD System. The current CAGD system that we are using is the Alpha_1 system, while work is underway on the construction of **MKS**. We are developing the system in the context of several applications projects, including 2-D visual inspection [8], 3-D computer vision [10] and object manipulation with both simple parallel grippers and more sophisticated dextrous hands [11].

2.1. The Multisensor Knowledge System

Much of our previous work on multisensor systems has concentrated on the specification of such systems and reasoning about their properties. It is necessary to be able to describe both the parameters and characteristics of individual components of multisensor systems, and to be able to deduce global properties of complete systems. Although it may be possible to deduce such properties (especially static properties like complexity, data type coercion, etc.), we believe that many interesting properties can only be determined by simulating the operation of the complete system.

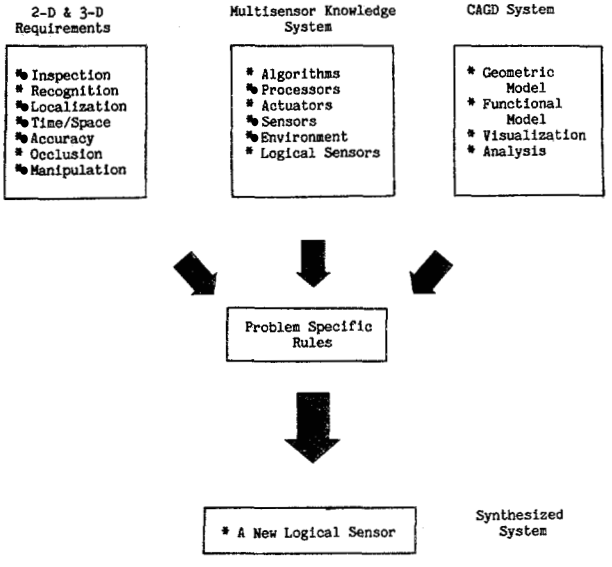

Figure 1. General Components of a CAD-Based Robotics System

Logical Sensor Specifications (LSS) permit an implementation independent description of the required sensors and algorithms in a multisensor system. Figure 2 gives a pictorial description of the basic unit: a *logical sensor.*

Sensor data flows up through the currently executing program (one of $program_1$ to $program_n$) whose output is characterized by the *characteristic output vector.* Control commands are accepted by the *control command interpreter* which then issues the appropriate control commands to the Logical Sensors currently providing input to the selected program. The programs 1 through n provide alternative ways of producing the same characteristic output vector for the logical sensor. The role of the *selector* is to monitor the data produced by the currently selected program and the control commands. If failure of the program or a lower level input logical

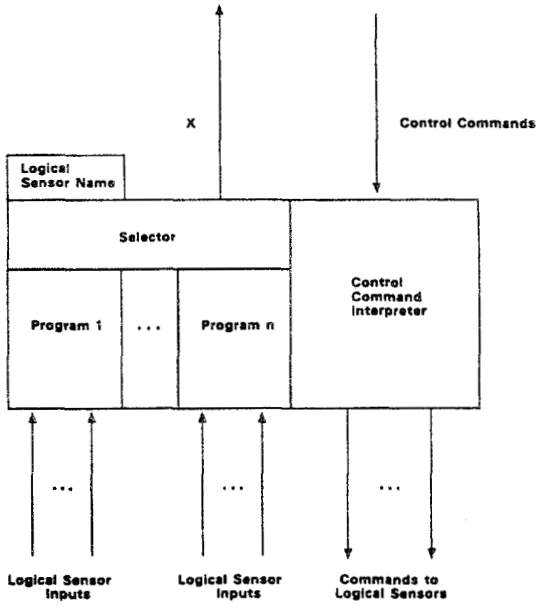

Figure 2. Logical Sensor Specification Building Block: The Logical Sensor

sensor is detected, the selector must undertake the appropriate error recovery mechanism and choose an alternative method (if possible) to produce the characteristic output vector. In addition, the selector must determine if the control commands require the execution of a different program to compute the characteristic output vector (i.e., whether dynamic reconfiguration is necessary).

Logical Sensor Specifications are useful then for any system composed of several sensors, where sensor reconfiguration is required, or where sensors must be actively controlled. The principle motivations for Logical Sensor Specifications are the emergence of significant multisensor and dynamically controlled systems, the benefits of data abstraction, and the availability of smart sensors. (For more on the various aspects of Logical Sensors, see [5, 4, 6, 7, 9, 10].) Related work includes that of Albus [1] on hierarchical control, Bajcsy et al. [2] on the Graphical Image Processing Language, Overton [15] on schemas, and Chiu [3] on functional language and multiprocessor implementations. For an overview of multisensor integration, see Mitiche and Aggarwal [13].

In exploring these issues, we have found that the specification of multisensor systems involves more than just sensor features. It is true that knowledge must be available concerning sensors, but it is essential to also be able to describe algorithms which use the sensor data and the hardware on which they are executed. In addition, the geometric knowledge provided by the CAD database helps to focus the synthesis of recognition and manipulation strategies. In the rest of the paper, we describe the components of an object-based approach to developing a knowledge system to support these requirements.

An object-based style of programming requires that the logical sensor of Figure 2 be re-described in terms of objects and methods. We shall next give the general flavor of this style, but it must be remembered that any particular sensor is actually an instance of some object class, and, in fact, inherits properties from many levels up.

Each logical sensor is completely specified as an object with slots for the *logical sensor name, selector function,* and the *logical sensor description.* There are also two methods defined on logical sensor

objects. In order to get data from a logical sensor, the *characteristic output vector method* must be invoked. Likewise, to issue control commands to the sensor (e.g., camera pan and tilt parameters), the *control commands method* must be used. The role of the *selector* is still the same as in previous logical sensor implementations, however, it now, in essence, is invoked to produce the characteristic output vector.

Such a representation makes it very easy to design sensor systems. Moreover, such specifications allow for replacement of sensors and dynamic reconfiguration by simply having the *selector* send messages to different objects. Given current object-based programming technology, such systems can be rapidly developed and permit dynamic typechecking (on objects).

Figure 3 shows the Multisensor Knowledge Base, and below the dashed line, a set of particular instances of various algorithms, sensors, etc. (drawn as circles). A logical sensor specification (indicated as a blocked in subset of the circles) defines a grouping of algorithms, sensors, etc. This newly created logical sensor is an instance of the *logical sensor object* and can be sent messages. As mentioned above, there are two methods defined on logical sensors: the *characteristic output vector method* and the *control commands method*. Thus, any logical sensor can be defined recursively in terms of other logical sensors (including itself).

Currently, our main interest is in the automatic synthesis of logical

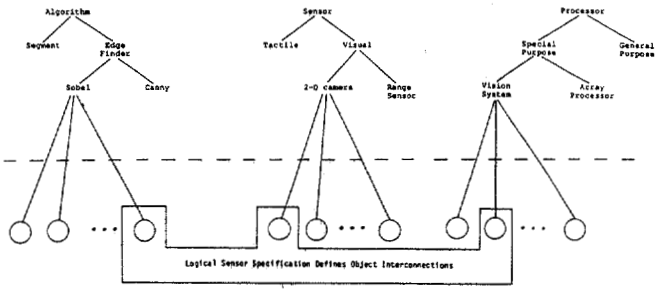

Figure 3. Logical Sensor Specification Using Object Instances

sensor specifications. Given a CAD model of an object, we would like to synthesize a specific, tailor-made system to inspect, recognize, locate or manipulate the object. Note that the synthesis of a logical sensor specification consists, for the most part, of interconnecting instances of sensors and algorithms to perform the task. This is done by writing the selector to invoke methods on other logical sensors. Given certain constrained problems, most notably the CAD/CAM environment, such a synthesis is possible.

2.4. The Alpha_1 CAGD System

The CAGD design system that we use is Alpha_1, an experimental solid modeling system developed at the University of Utah. For the past few years the Computer Aided Geometric Design group has been involved in a concerted effort to build this advanced modeler. Alpha_1 incorporates sculptured surfaces and embodies many theoretical and algorithmic advances. It allows in a single system both high-quality computer graphics and freeform surface representation and design.

It uses a rational polynomial spline representation of arbitrary degree to represent the basic shapes of the models. The rational B-spline includes all spline polynomial representations for which the denominator is trivial. Nontrivial denominators lead to all conic curves. Alpha_1 uses the Oslo algorithm for computing discrete B-splines. Subdivision, effected by the Oslo algorithm, supports various capabilities including the computation associated with Boolean operations, such as the intersection of two arbitrary surfaces. B-splines are an ideal design tool, they are simple, yet powerful. It is also the case that many common shapes can be represented exactly using rational B-splines. For example, all of the common primitive shapes used in CSG systems fall into this category.

Other advantages include good computational and representational properties of the spline approximation: the variation diminishing property, the convex hull property and the local interpolation property. There are techniques for matching a spline-represented boundary curve against raw data. Although the final result may be an approximation, it can be computed to any desired precision (which permits nonuniform sampling).

3. An Example Application: CAD-Based 2-D Vision

A simple example which demonstrates some of the power of the Multisensor Knowledge System approach is that of CAD-Based 2-D Vision. The goal is to automate visual inspection, recognition and pose determination of parts using pattern recognition techniques on features extracted from binary images. Figure 4 shows the scheme pictorially.

The Multisensor Knowledge System stores knowledge about the algorithms, sensors, processors, etc. This knowledge is used by application specific rules. The systems to be synthesized here require that a model be created for the part to be inspected, and that a robust and (perhaps) independent set of features be chosen along with an appropriate distance metric.

A special Vision Shape Editor (VSE) interface was added to Alpha_1 for the class of parts in this application (2-D polygons with circular or polygonal holes). VSE automatically generates a toolpath for a 5-axis mill. The path is displayed (superimposed over the designed object) upon exit from VSE.

The lower left side of the figure shows the offline training component. The new part is designed using VSE; then a set of images is rendered by the CAGD system giving a sample of various views of the part in different positions, orientations, and scales. These serve as a training set to the Multisensor Knowledge System.

A set of rules (or productions) performs an analysis of the views of the part to select a subset of the total set of possible features. Features are used if they are robust, independent and reliable. Once these features have been chosen, a new logical sensor object is created whose only function is to recognize the given part based on an analysis of the selected features. The part detector is then linked into a particular application (e.g., an inspection task at a specific workcell) by sending a message to the appropriate camera.

As a specific example, consider the object shown in Figure 5. This is the workpiece as designed using VSE. Figure 6 shows the NC path overlaid on the object. The object was rendered at orientations of 0, 22.5 and 45 degrees (see Figure 7), and these images were analyzed. A standard set of possible features was used for object modeling, including area, perimeter, etc., as well as the seven invariant moments given by Hu [12]. Figure 8 shows the actual milled workpiece. It was milled as an island (in wax stock) and then sliced off. The model produced from the off-line analysis of the rendered images is packaged into a distinctly executable object.

The synthesized logical sensor object merely sends a message to the segment program for Camera 1 (a Fairchild 3000 CCD camera, see Figure 9), then sends a message to each of the features used, then sends a message to the distance function object with the appropriate weights. The system has been implemented in FROBS [14] (FRames and OBjectS, a system which supports frame objects and is implemented in Common Lisp) using objects and methods. The feature calculations are performed by running C code called from within the instances of the feature objects. Recognition

experiments have been successfully run on several designed and milled objects. Future work includes the incorporation of more robust recognition schemes, e.g., Local Feature Focus.

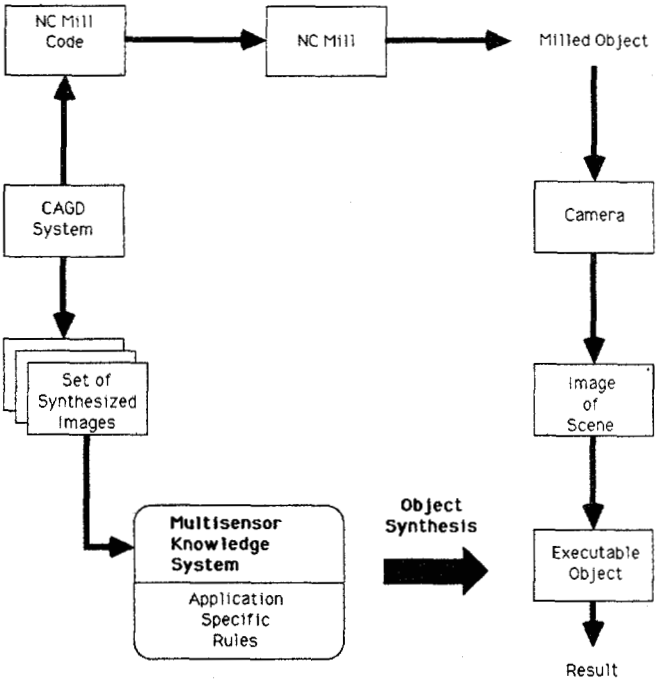

Figure 4. Synthesis of NC Path and Part Detector

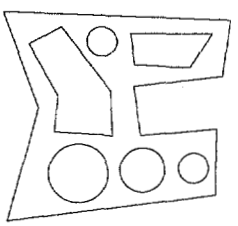

Figure 5. An Example Workpiece Design

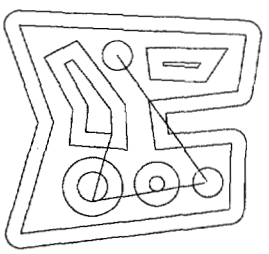

Figure 6. NC Path Overlaid on Workpiece Design

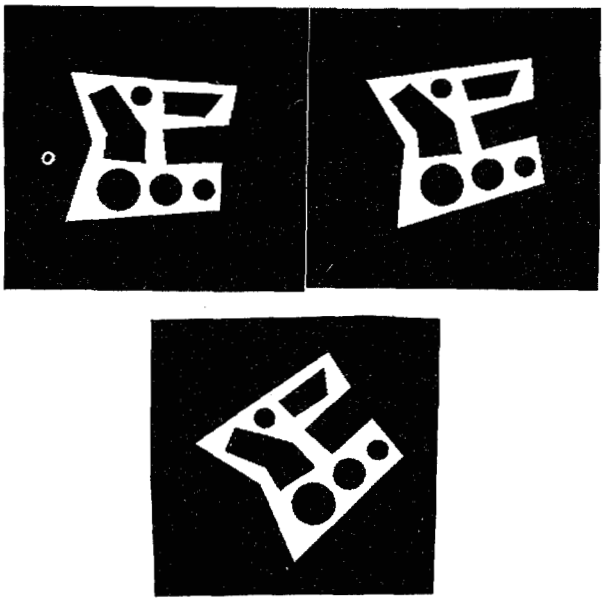

Figure 7. Example Workpiece Rendered at Three Orientations

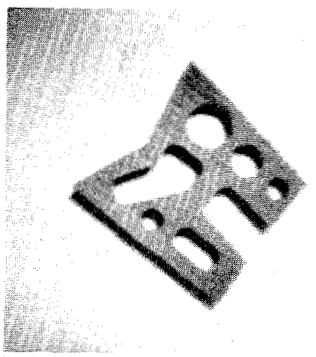

Figure 8. Milled Workpiece

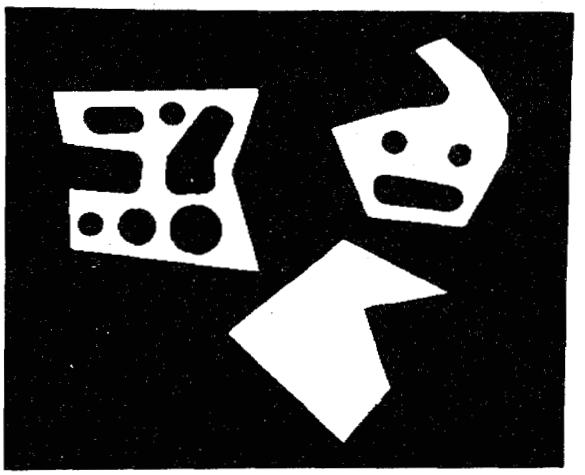

Figure 9. An Image of Actual Scene to be Analyzed

4. Summary and Future Work

CAD-based robotics offers many advantages for the design, construction, and simulation of robotics systems. We have described many of those. We are currently working on CAD-Based 3-D vision system. That is, we are developing a set of rules which will evaluate the 3-D geometry and function of any part designed with the Alpha_1 CAGD system. In this way, weak recognition methods can be avoided and specially tailored logical sensor objects can be synthesized automatically. Another area of current research interest is the simulation of multisensor systems. We believe that our approach can lead to very natural, straightforward, and useful simulations which can include native code running on the target processors. Finally, we are also investigating the organization of knowledge in the Multisensor Knowledge Base. Certain structuring of the data may lead to improved or simplified analysis.

Acknowledgment: We would like to thank the Alpha_1 group for their cooperation in this project, and in particular, Russ Fish for his technical assistance.

References

[1] Albus, J.
Brains, Behavior and Robotics.
BYTE Books, Peterborough, New Hampshire, 1981.

[2] Bajcsy, R.
GRASP:NEWS Quarterly Progress Report.
Technical Report Vol. 2, No. 1, The University of Pennsylvania, School of Engineering and Applied Science, 1st Quarter, 1984.

[3] Chiu, S.L., D.J. Morley and J.F.Martin.
Sensor Data Fusion on a Parallel Processor.
In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 1629-1633. San Francisco, CA, April, 1986.

[4] Henderson, T.C. and E. Shilcrat.
Logical Sensor Systems.
Journal of Robotic Systems 1(2):169-193, 1984.

[5] Henderson, T.C., E. Shilcrat and C.D. Hansen.
A Fault Tolerant Sensor Scheme.
In *Proceedings of the International Conference on Pattern Recognition*, pages 663-665. August, 1984.

[6] Henderson, T.C., C.D. Hansen, and Bir Bhanu.
The Specification of Distributed Sensing and Control.
Journal of Robotic Systems 2(4):387-396, 1985.

[7] Henderson, T.C., Chuck Hansen and Bir Bhanu .
A Framework for Distributed Sensing and Control.
In *Proceedings of IJCAI 1985*, pages 1106-1109. Los Angeles, CA, August, 1985.

[8] Henderson, T.C., Chuck Hansen and Bir Bhanu.
The Synthesis of Logical Sensor Specifications.
In *Proceedings of the SPIE Conf. on Intelligent Robots*, pages to appear. Boston, MA, September, 1985.

[9] Henderson, T.C. and Steve Jacobsen.
The UTAH/MIT Dextrous Hand.
In *Proceedings of the ADPA Conf. on Intelligent Control Systems*, pages to appear. Ft. Belvoir, Va., March, 1986.

[10] Henderson, T.C., Chuck Hansen, Ashok Samal, C.C. Ho and Bir Bhanu.
CAGD Based 3-D Visual Recognition.
In *Proceedings of the International Conference on Pattern Recognition*, pages to appear. Paris, France, October, 1986.

[11] Henderson, T. and R. Grupen.
A Survey of Dextrous Manipulation.
Technical Report UUCS-TR-007, The University of Utah, Department of Computer Science, July, 1986.

[12] Hu, M.K.
Visual Pattern Recognition by Moment Invariants.
IRE Transactions on Information Theory IT-8:179-187, 1962.

[13] Mitiche, A. and J.K. Aggarwal.
An Overview of Multisensor Systems.
SPIE Optical Computing 2:96-98, 1986.

[14] Muehle, Eric.
FROBS Manual.
Technical Report PASS-note-86-11, University of Utah, October, 1986.

[15] Overton, K.
Range Vision, Force, and tactile Sensory Integration: Issues and an Approach.
In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 1463. San Francisco, California, April, 1986.