BUILDING REPRESENTATIONS FROM FUSIONS OF MULTIPLE VIEWS

Ernest W. Kent, Michael O. Shneier, and Tsai-Hong Hong

Sensory-Interactive Robotics Group
National Bureau of Standards
Gaithersburg, MD 20899

ABSTRACT

A robot sensing system is described that uses multiple sources of information to construct an internal representation of its environment. Initially, object models are used to form the basic representations. These are modified by processes that operate on sequences of sensory information, obtained from sensors that move about in the environment. Two representations are constructed. One is a description of the spatial layout of the environment, represented as an octree, while the other is an object- and feature-based representation. The system handles both expected and unexpected objects, and attempts to register its internal representation with the external world using a variety of predictive, sensory-processing, and matching procedures.

1. Introduction

A characteristic of robot applications is that most of the system's sensory processing time is spent on sensor-guided position servoing, rather than on object identification. This is because almost all of the sensory data with which the robot deals are encountered within a slowly-changing historical context. In this respect, the problem domain is very similar to that of animal sensory processing. After objects are initially acquired by the sensory system, its principal job is to provide continuous relative position information to guide the control system.

In our approach, an internal description of the world is maintained and continuously servoed to the sensory data. All sensory information required by the control system is obtained from this internal description, which represents the system's "best guess" about the external world. A major advantage of interposing such a description between the sensory and control systems is that it permits a decoupling of the two systems. The sensory system can decide for itself where best to use its resources, instead of having to attend to acquiring the specific data required by the control system, and the control system need not wait on the sensory analysis of recently sampled data to receive the data it needs. In our implementation, there is some low-level servoing that does directly rely on sensing, for example, when approaching an object using range information. This is necessary to avoid collisions, and to account for rapid changes in the environment. It can be considered a part of the internal representation that changes as fast as the sensors can operate.

The world model is built using all available sensory input, including senses for external data, such as vision, touch, and proximity, and senses for internal data, such as joint angle and force. Internally stored knowledge is also employed. It consists of general knowledge about the environment (such as descriptions of classes of known objects), and particular knowledge, such as the names and locations of objects which are expected in the current context. The representation problem thus includes both the fusion of information from multiple sensory sources, and the fusion of these data with object models and other *a priori* knowledge.

The representation scheme employed consists of two, linked representations. One of these is a spatially-indexed representation of occupied and empty space in the working volume of the robot. The other is a linked structure of tables and lists, indexed by objects and features, which represents the system's knowledge of objects in the working volume.

The spatial representation is organized as an octree, a regular decomposition of a cube into octants ([10], [13]). Each octant may be split if it is not homogeneous, giving rise to a tree representing the workspace. Homogeneous nodes in the tree, called leaves, may represent empty space, where it is known that no objects lie, or object volumes, where it is possible that an object exists, or unknown regions, where no sensory information is available. The spatial representation is useful in computing free paths for trajectory analysis, and for answering questions about the identities of objects or features in given locations. The representation is also used to simplify the task of two-dimensional prediction of sensory data by explicitly representing the spatial relationships between objects.

The feature-based representation is linked to both the octree and the generic object models. It has entries for each object known or hypothesized to be in the workspace, including objects that have not yet been recognized. Each object is associated with the set of features that verifies its identity, and recognized objects are linked to their geometric models.

The feature-based representation is especially suited to answering questions about objects or features by name or by description. Some questions rely on both the representations for their answers. For example, deciding if an object is occluded from a particular viewpoint involves first finding the entry for the object in the feature-based representation, and then following the links to the spatial representation to find the answer.

2. Constructing the Internal Representations

The goal of the system is to construct and maintain an internal representation of the world that is as close as possible a reflection of the actual situation. There are two aspects to the accomplishment of this goal. The first deals with verifying expectations, while the second deals with describing and representing unexpected events. Much of the processing is the same for both aspects, but each has some special requirements.

The assumptions made about the environment are as follows. The robot operates in a constrained environment (a metalworking factory cell, in our implementation). An outside knowledge base supplies CAD descriptions of the objects expected to be in the world before the start of each task. Instances of these objects are expected to appear in specified locations. The sensors are mobile, being mounted on the wrist of the robot, and their positions can be established at any time by reading the joint positions of the robot. From an initial expectation about the state of the world, the sensors

must confirm or modify hypotheses about the positions and identities of objects, recognize and describe unexpected objects, and describe the regions of free space suitable for navigation.

A collection of algorithms has been implemented to accomplish these tasks. They are described below by following the sequence of events during the execution of a task.

2.1. Initial Processing

When the factory management system activates a cell to manufacture a particular set of parts, it extracts the CAD descriptions of the parts from a database, and sends them to the sensory system. The descriptions include a representation of each part in terms of surfaces, edges, and vertices, and an object-centered octree representation of the volume occupied by the part. Each description is generic, in that there can be many instances in the world at the same time. The initial set of instances and their expected positions and orientations are supplied to the sensory system by a materials-handling workstation when it delivers the parts.

The sensory system accepts the generic object descriptions and specific expectations about instances of the objects. Its first task is to set up its internal representation of the world. This has two parts. It must build the initial spatial representation, and the initial object- and feature-based representation.

The initial octree representation is set up by projecting the object-centered octrees associated with each object-instance into the world octree. This involves an arbitrary rotation and translation of each tree. A top-down algorithm has been developed for this purpose [8]. This algorithm is used extensively throughout the task. Each time a new position and orientation of an object is established through sensory feedback, a new projection is performed. The algorithm works by projecting a rotated and translated version of the octree back onto the untransformed object tree, and labeling nodes in the transformed tree according to those in the underlying, untransformed tree. An example of the operation of this algorithm is shown in Figure 1.

Construction of the initial object-based representation is relatively simple. A row in the table is constructed for each object instance. Each instance is given a unique name, which is cross-indexed to the octree. Columns in the table are created for significant feature types. Currently only boundary features, such as corners, and holes in surfaces are implemented. We are considering extending this set to include the cranks, beaks, and ends described in [2], and the shapes of surface patches.

2.2. Processing Sequences of Sensory Data

Once the initial representation is set up, the sensory system has a current best guess about the world. It can immediately use this to respond to questions from the control system that guides the robot in performing the task.

The next step is to use the sensors to register the internal representation with the world. This involves first generating predictions from the internal representations about what each sensor should observe, and then matching the predictions with the actual data. Predictions are generated first in 3D world coordinates using the position of the sensor and its field of view to locate significant features that should be visible. Only those small regions of the field of view that contain the predicted features need be processed, and a complete prediction of the sensor's response is not necessary.

The selected features are screened for visibility in two passes. First, they are examined for self-occlusion, and then for occlusion by other objects in the world. Features associated with surfaces or sets of surfaces that point away from the sensor are pruned first. Next, the remaining features are checked to see if a ray from the sensor to the feature intersects another part of the object before it reaches the feature. Finally, the remaining features are checked for occlusion by other objects. This is done by projecting rays through the octree from the sensor to each feature. Hidden features are once again removed.

The remaining features are projected into the sensor's image plane, and become the predictions. Each feature is enclosed in a window indicating the uncertainty of its position, and has an associated label giving its name and the name of the object instance to which it belongs. This makes later matching at the 2D level very simple.

Sensory processing involves extracting the expected feature types within the predicted windows, and matching them with the expectations. When a match is found, it is trivially labeled from the prediction, but its position is established from the sensory data, and will be used to update the internal representation of the object's position.

It is also necessary to deal with unexpected objects, that is, with sensed data that were not predicted. Currently, this is done by performing a connected-components analysis on the whole image, and treating as unexpected those components that do not intersect with any of the predicted windows. These components are processed generically to extract a standard set of features, each of which is given a unique label. Substantial processing involving multiple images from different viewpoints is necessary to describe these unexpected objects.

A prototype object-based table representation is also constructed, with rows for each expected and unexpected object, and columns containing the extracted features. Errors in the predictions are sent to the prediction processes to supply feedback to improve later predictions.

Included in the 2D predictor is a tracking algorithm for unexpected features. It is important to be able to retain the identity of a feature across a sequence of images if 3D information is to be extracted. The tracker uses the motion of the sensor, and past estimates of the motions of features to predict their next positions. It construct predictions on this basis, and uses the error feedback to refine its estimates in the absence of 3D information.

The 2D matches and unexpected features are also sent to a 3D matching procedure. For expected objects, this procedure uses a least-squares technique to register the models with the labeled features, resulting in updated position and orientation information about each object. This information is used in the octree to reposition the objects, using the translation and rotation algorithm, and in the table to update the positions of the features.

For unexpected features, the main tool for description is consistency across frames of data. For example, a point feature in one view is constrained to lie on a vector projecting from the position of the sensor towards the feature. When the sensor moves to another position, a second vector can be drawn, given that the feature can be tracked and put into correspondence with its earlier manifestation. The intersection of these vectors is an estimate of the 3D location of the point. Naturally, there is a lot of error in this estimate, so it is desirable to track points across many frames, and use a pseudo-intersection of all the vectors as an estimate of the true position.

The 3D unrecognized features are sent to a special recognition module that uses a graph-matching algorithm to try to identify groups of features with parts of objects whose models are in the database. This is not a real-time process, but, when it succeeds, it sends its results to the 3D prediction module, which can then create more accurate predictions, and to the octree and table representations to update the understanding of the world.

Even without recognition, however, the features must be described, and predictions made about their appearance in later sensory data. The 2D tracker can accomplish some of this but, as the 3D positions of features become better established, 3D predictions are made as well. In the case of unexpected features, the world octree is used exclusively in determining the visibility of features.

Thus, there are two feedback loops that serve to update the internal representation as sequences of sensory data are processed. The first uses the results of 2D matches and tracking to predict new feature positions. it works very fast because the sensory processing is restricted to small windows, and the matching is straightforward. The second involves the recomputation of the positions and orientations of objects, based on matching features with models. This takes longer than the 2D process, but provides more useful information and allows better 2D predictions for further processing. It also builds up consistent descriptions of unexpected objects allowing them to be manipulated as single objects.

2.3. Updating the Octree

In parallel with the above processing is a second process that operates to update the octree representation. It works directly from images of the world, taken from known positions. The 2D silhouettes of objects in the images are projected into the world octree as cones. When an object is seen from several viewpoints, the intersections of the cones constrain the position and size of the object (Figure 2). After several views, the representation begins to resemble the true shape of the object. At all times, the representation of an object occupies a volume at least as large as the object. This is important because the octree is used for path planning as well as spatial representation [7], so the free space represented must be guaranteed to be empty. The algorithm used for the projection is described in [9]. An example of its use is shown in Figure 3.

The two paths to constructing the spatial representation complement each other. If a cone intersects with a known object, the generic object octree is used to refine the resulting occupied volume. For unexpected objects, the volume provides evidence for the existence of single objects or multiple discrete objects. Feature that intersect with a volume can be grouped for the purposes of recognition. If it is discovered that an object must be split, the octree can help in dividing the features among the newly-created sub-objects. The octree is also essential to the hidden-feature analysis, and for path planning.

3. Implementation

The system is implemented in a network of microprocessors, operating asynchronously, and communicating through common memory. This allows each module to work at its own pace. There is one module for each major processing step (Figure 4). Information is passed when it is ready. If a module is not ready to receive it, it may queue it for later processing, or ignore it, and receive more up-to-date information later. This is useful if some processes run much faster than others, and if the slower processes do not need to be updated very frequently. An example of a process that requires frequent updating is the 2D tracker, which must process data fast enough to ensure that features do not disappear or merge. A slower process, like the 3D matcher can afford to miss some data, because very closely spaced data points do not significantly improve its position and orientation estimates.

Sensors that are currently supported are a camera, providing full-field images, and a structured-light sensor that projects two planes of light into the world and uses triangulation to measure range and orientation of visible surfaces [1]. The octree projection algorithm uses only the camera data (although it could be modified to use range information). The structured light data are used to good effect by the 3D matching routines. The 3D matching routines for structured light data use models of the objects, which they attempt to rotate and translate to fit over the actual data. This is done using a steepest descent method, minimizing an error function that measures the distance between surfaces in the model and surfaces in the data. The technique separates out the rotational and translational components and solves them separately ([14], see also [5]).

4. Discussion

The ability to describe scene contents, irrespective of the ability to name things, is fundamental to a sensory system intended for sensory-servoed robot guidance. An understanding of the spatial and temporal structure of the environment is basic to the ability to physically act in it. It is only when the physical structure of the environment is understood that the system can act on information gained by recognition of objects.

Dealing with unrecognized objects is also of crucial importance. The robot must avoid collision with unrecognized objects and maneuver around them. Additionally, it will usually need to inspect unknown objects in an attempt to identify them, and this may involve actually manipulating them. Unrecognizable objects may have to be removed from the workspace or relocated within it.

For guiding a manipulator, a model of the spatial and temporal structure of the environment is necessary. A representation that explicitly represents space and the relationships between the objects it contains is essential. In our system this is accomplished with the octree representation of the workspace. The nodes of the octree indicate whether the volume that each represents is occupied, empty, or unexamined, and, via pointers to the table-based representation, by what it is occupied.

While there has been substantial earlier work on constructing representations from multiple views (for example, [3], [4], [6], [9], [11], [12], [15]), previous research has usually involved subsets of the current goals. We are attempting to build, in real-time, both spatial and object-based representations of a complex environment, containing many objects, some of which are unexpected. In addition, these representations must be available for answering questions about the world posed by a robot control system, so must always contain valid information. The problem has been attacked by dividing it into a number of modules, each of which is an expert at a particular task. Heavy use is made of prediction to reduce the amount of computation, and even unexpected objects can give rise to predictions after they have been observed across several frames of data.

The methods also lend themselves to the use of multiple sensors, so long as conflicts in readings (for example, different ranges to the same surface from different sensors) can be resolved. Most of the matching is independent of the sensory modality, so long as labels can be attached to features, and the same feature always receives the same label. Unless it is of importance, the internal representation need not even maintain a record of how a piece of information was obtained.

The initial implementation, while including all the described modules, is impoverished in a number of ways. It would benefit from using more feature types and more sensor types. There is currently little control over the goals of the system. A supervisor module is being constructed that will examine the internal representation and decide where the system resources would best be spent, given the task specification. The octree can help in deciding where to point the sensors, because it explicitly represents unseen space, but deciding which sensor will provide the most useful information at any time is an open problem. It would also be useful to be able to automatically generate an optimal set of features to extract for a given mix of parts so that processing could be minimized. These problems are the subject of ongoing research.

References

1. J. S. Albus, E. Kent, M. Nashman, P. Mansbach, L. Palombo, and M. Shneier, A 6-D vision system. Proc. SPIE Technical Symposium, Crystal City, Virginia, May 1982.

2. H. Asada and M. Brady, The curvature primal sketch. IEEE Trans PAMI *8*, 1, January 1986, 2-14.

3. C. I. Connolly, Cumulative generation of octree models from range data. Proc. International Conference on Robotics, Atlanta, Ga, March 1984, 25-32.

4. F. P. Ferrie and M. D. Levine, Piecing together the 3D shape of moving objects: an overview. Proc. CVPR '85, San Francisco, CA, June 1985, 574-584.

5. W. E. L. Grimson and T. Lozano-Perez, Model-based recognition and localization from sparse range or tactile data. *International Journal of Robotics Research 3,* 3, 1984, 3-35.

6. M. Herman, Matching three-dimensional symbolic descriptions obtained from multiple views of a scene. Proc. CVPR '85, San Francisco, CA, June 1985, 585-590.

7. M. Herman, Fast, three-dimensional, collision-free motion planning. Proc. International Conference on Robotics and Automation, San Francisco, CA, April 1986.

8. T-H. Hong and M. O. Shneier, Rotation and translation of objects represented by octrees. Robot Systems Division, National Bureau of Standards, 1985, submitted to 8th ICPR, Paris, October 1986.

9. T-H. Hong and M. O. Shneier, Describing a robot's workspace using a sequence of views from a moving camera. IEEE Trans. PAMI *7,*
6, November 1985, 721-726.

10. C. L. Jackins and S. L. Tanimoto, Oct-trees and their use in representing 3D objects. *Computer Graphics and Image Processing 14* 1980, 249-270.

11. H-S. Kim, R. C. Jain, and R. A. Volz, Object recognition using multiple views. Proc. International Conference on Robotics and Automation, St Louis, MO, March 1985, 28-33.

12. W. N. Martin and J. K. Aggarwal, Volumetric descriptions of objects from multiple views. IEEE Trans. PAMI *5,* 2, March 1983, 150-158.

13. D. Meagher, Octree encoding: a new technique for the representation, manipulation and display of arbitrary 3D objects by computer. Tech. Rep. TR-IPL-111, Dept. Electrical Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY, 1980.

14. W. S. Rutkowski and R. Benton, Determination of object pose by fitting a model to sparse range data. Robot Systems Division, National Bureau of Standards. Also in Intelligent Task Automation Interim Technical Report No. 4, January 1984.

15. S. A. Underwood and C. L. Coates, Visual learning and recognition by computer. TR123, Information Systems Research Laboratory, University of Texas, Austin, 1972.

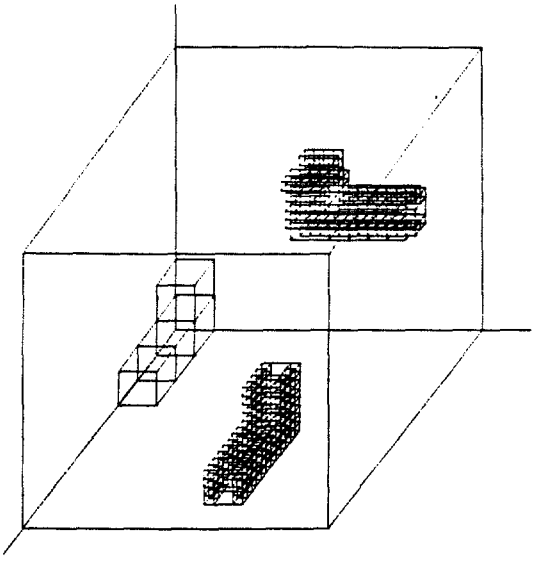

Figure 1. Three transformed instances of an l-shaped object.

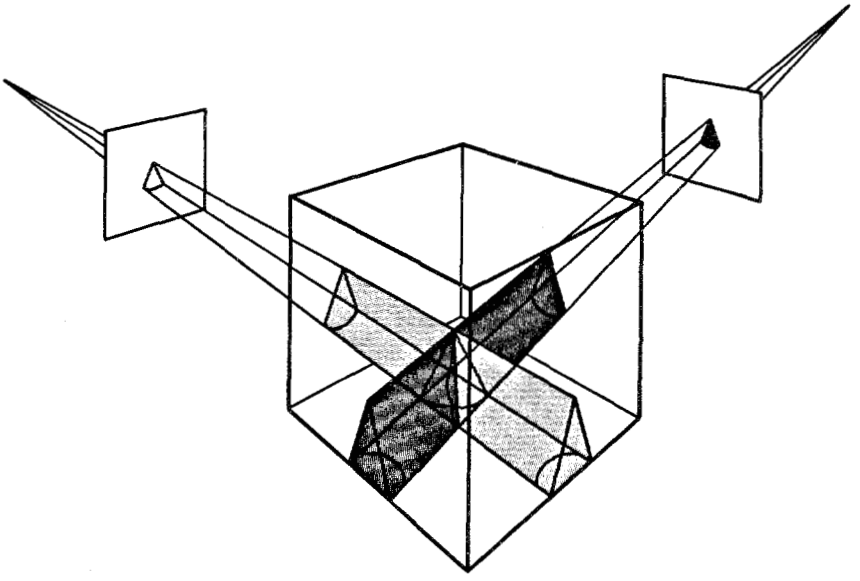

Figure 2. The effect of intersecting the cones arising from two views of an object.

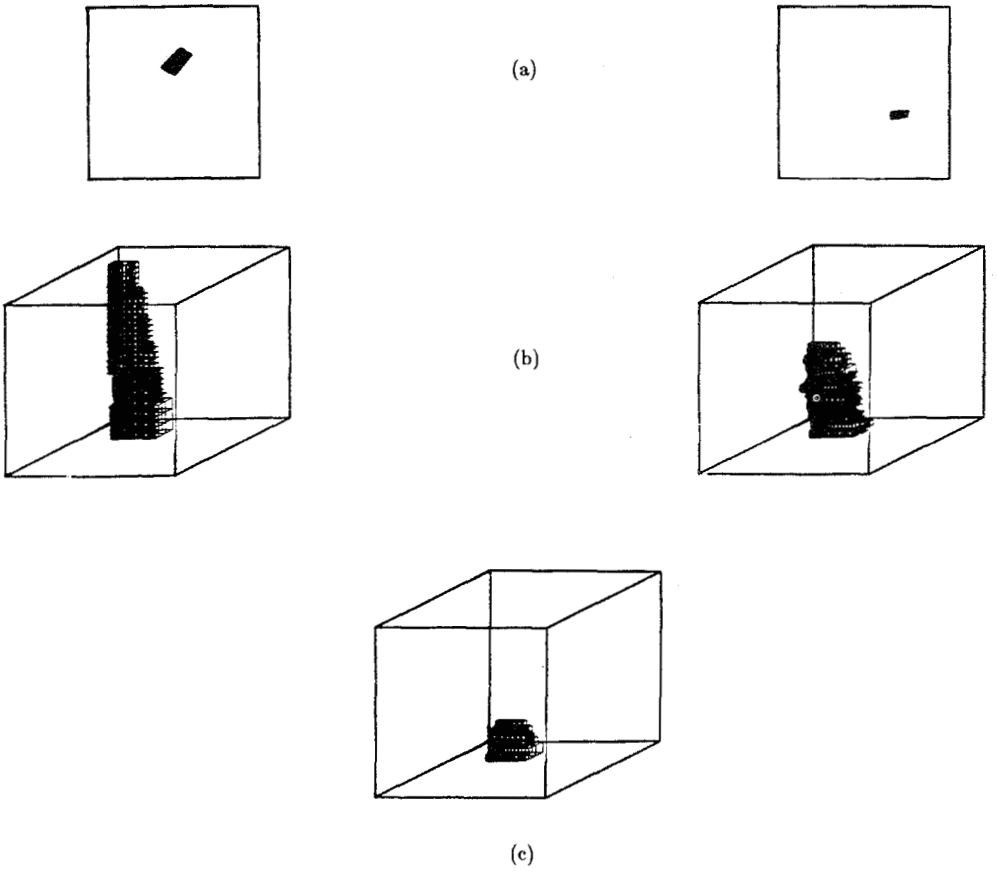

Figure 3. Two views of an object (a), the cones resulting from these views (b), and the intersection of their volumes (c).

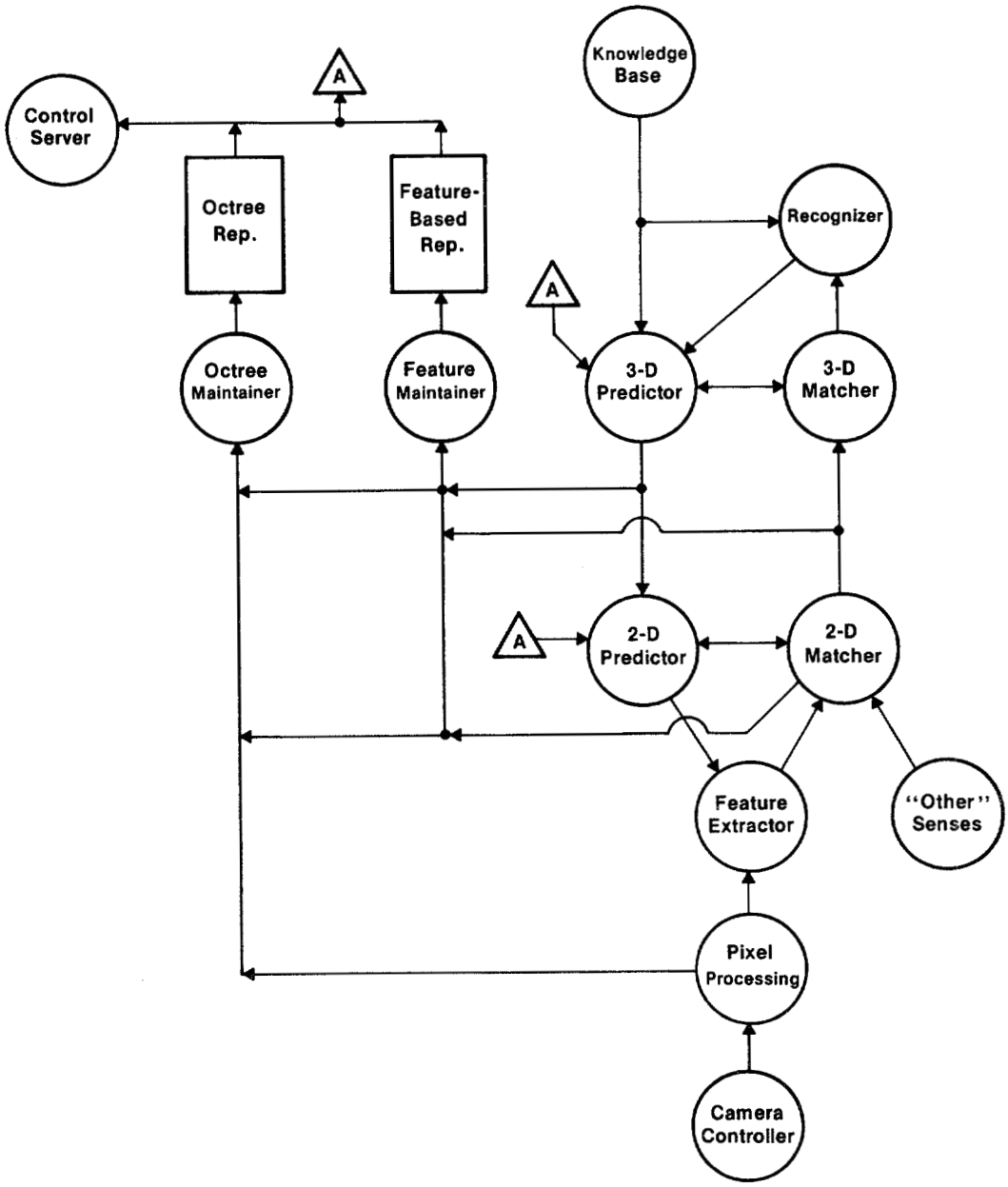

Figure 4. A schematic representation of the sensory system. The circular icons represent processors, while the rectangular icons represent the internal representations.