REPRESENTING SOLIDS FOR A REAL-TIME ROBOT SENSORY SYSTEM

Ronald Lumia
Building 220, Room A-127
Industrial Systems Division
National Bureau of Standards
Washington, D. C. 20234

ABSTRACT

The goal of the sensory system is to supply sufficient information to the robot control system, which actually moves the robot, to accomplish a desired task. The sensory system constantly updates a model of the 3-D workspace to reflect reality. This model is decoupled from the sensory processing so that the control system can be given responses without having to wait for sensory processing. A real-time world model is presented which incorporates both CAD descriptions of known parts and information about each specific object in the workspace. In order to enhance its speed, the 3-D world model supports several representations. The world model predicts the 3-D features for the objects in the workspace of the robot which are used by other modules in the sensory system.

KEYWORDS: automated manufacturing, knowledge bases, part representation, real-time processing, sensory and vision systems, solid modelling.

Commercial equipment is identified in this paper in order to adequately describe the systems that were developed. In no case does such identification imply recommendation by the National Bureau of Standards, nor does it imply that this equipment was necessarily the best available for the purpose.

1. INTRODUCTION

The goal of the robot sensory system is to provide the robot control system with the information required to perform a task. The real-time robot control system and the methodology for programming are described in (Barbera et al., 1984). Questions of concern to the robot control system include the location of desired parts, existence of freespace for trajectory planning, range from the desired part, etc. The robot control system should experience as little delay as possible from the sensory system if the robot is to operate in real-time. The way in which information is represented can have a large impact on the ultimate efficiency and speed of the system. This paper discusses some of the compromises involved in the design of a 3-D database which describes objects within the robot's workspace. The sensory system is multi-modal in nature; it accepts data from image sensors, proximity sensors, touch sensors, etc. Although most of the examples given in this paper involve images, any other sensor data could have been used.

The sensory system, which is hierarchical in nature, is divided into two parts: the lower vision levels and the upper vision levels. The lower vision levels are responsible for gathering and pre-processing data from the sensors. Much of this processing is generic in nature, i.e., it is independent of the image data. For example, run-length encoding and connected components analysis are performed on all images. The details of some of the lower level vision modules are described in (Kent, 1982).

The processing in the upper levels of the the sensory system, described by (Shneier et al., 1984a), is a function of the current goals of the system. The sensory system is continually trying to servo a world model to reality. While the robot moves, sensor data are used to update the world model to reflect the true nature of the robot workspace. A 3-D model database is used to create predictions of features which should be detected. Predictions are matched with sensor data and the pose of each object in space can be modified to reflect reality.

The basic problem addressed by this paper is one of representation. What is the best way to represent the object models in the 3-D database for use with the sensory system? There are several representation schemes available for solids. Each method has advantages and disadvantages. This paper develops the NBS sensory system 3-D model requirements, explores the choices for object representation and presents the representation adopted for the sensory system.

The next section describes the different ways in which solids are represented. This is followed by a discussion of the operation of the sensory system. Finally, the representation adopted for the sensory system is presented and is compared with other schemes.

2. REPRESENTING SOLIDS

Many areas of science and technology (including architecture, computer graphics, design of metal parts, etc.) are employing techniques where solids must be represented in a computer. This is shown by the number of systems available for solid modeling (Requicha et al., 1983. Baer et al., 1979. Myers, 1982). Unfortunately, the best representation depends on the intended application. Many factors influence the choice of the representation. The representation itself should be complete, concise, and unambiguous. However, using the representation should be fast and easy.

2.1. PROPERTIES OF REPRESENTATIONS

All of the methods for solid representation can be discussed in terms of several properties (Requicha, 1980). Validity is concerned with the creation of a representation which corresponds to a real solid and not a nonsense object (Waltz, 1975). Completeness requires that each representation correspond to only one object. Uniqueness expects one and only one representation for each object. Conciseness refers to the memory requirements for storing the representation. Efficacy implies how the representations are used. They are sources of data for other algorithms. Some representations store data in a more usable form than other representations for a given algorithm.

These properties cannot be optimized simultaneously. For example, a representation which is concise may not satisfy the uniqueness property or the efficacy property. This is not cause for alarm. It is reason for selecting the representation most appropriate for the expected algorithm.

2.2. SOLID REPRESENTATION METHODS

There are several representation schemes for solids: Pure primitive instancing, spatial occupancy enumeration, constructive solid geometry, and boundary representation.

In pure primitive instancing, the basic assumption is that parts fall into families which differ only in a small number of parameters. For example, a block may be specified completely by length, width, and height. This method is concise and easy to use but the domain over which parts can be created is relatively small. Consequently, it is rarely used as the only representation but as a feature within another representation such as in PADL-2 (Brown, 1982).

Spatial occupancy enumeration is a scheme where the list of volumes occupied by an object are listed. This method can be quite verbose unless a hierarchical coding scheme such as an octree, a regular decomposition of a cube into subcubes, is used. In spite of the fact that the representation is unique and valid, it is relatively

incompatible with most algorithms because it is difficult to extract useful information from the representation in a reasonable time. It is quite useful to have an indication of the volumes taken up in the workspace. The NBS sensory system uses a coarse octree as an indication of spatial occupancy rather than as an accurate model.

Constructive solid geometry (CSG) is a popular representation which is used in several extant systems, e.g. PADL-2 (Brown, 1982), GE-SOLID (Boyse and Gilchrist, 1982), and Synthavision (Goldstein and Malin, 1979). In CSG, solids are represented as combinations of primitive solids or "building blocks" using the Boolean operations of union, intersection, complement, difference, etc. This results in a concise, unambiguous, complete representation. Furthermore, valid objet representations are guaranteed. Although the CSG representation appears to be excellent for creating objects, it seems less suited to any algorithms which need data in a different form. Such algorithms include display routines, where projection is a key operation. Unfortunately, the higher vision levels predict the projected location of a feature and attempt to match it with the feature extracted from the empirical data. Since the CSG representation is complete, it is clearly possible to calculate the required projection. However, such calculations are time-consuming.

A boundary representation views a solid in a hierarchical manner. The solid is composed of a set of surfaces. Each surface is built from a list of bounding edges. Finally, each edge is represented by bounding vertices. For relatively complex objects, the representation can become rather long. The size of the representation is often exacerbated by the intentional storage of redundant information. Also, it is possible to create nonsense objects with dangling edges or faces. With all of these problems, it is reasonable to ask why this representation is so popular.

For a real-time system, the time delay incurred in using the representation is more important than the size of the representation. This is especially true as the cost of memory continues to decrease. As a result, redundant information is intentionally incorporated into the model in order to minimize run-time calculations.

The validity of the representation is an important consideration but does not need to be dealt with in real-time. Computationally expensive validity checking algorithms can be run on the representation once off-line. The validity does not need to be checked each time the representation is used.

The major advantage of this representation is that the data can be organized around the *features* of the object. These features can extend the minimal vertex, edge, surface information to include the needs of the algorithms. Edge length, surface normal vector, and parametric equations describing curves and surfaces can be

incorporated directly into the representation. This results in a database where algorithms can operate in real-time.

3. SENSORY SYSTEM ARCHITECTURE

Figure 1 shows the architecture for the sensory system. Information about the task arrives from the factory. This includes CAD descriptions of each object and an expected location of each object in the workspace of the robot. This information is stored in the 3-D world model, which is called world model-II (WM-II). The world model determines a list of features which best differentiate the expected parts. The expected values of these features are calculated and stored for each object in the workspace. After this initialization step, all processes must run as rapidly as possible.

The 3-D world model stores the current best guess about the position of objects in the robot's workspace. Since other modules in the system require different types of information from this database, the way in which information is represented can have a significant impact on system speed.

All processing performed by the sensory system during a task is centered about keeping the world model in registration with the world. As the robot and sensors move about in the workspace, information is updated in the model to reflect reality. This servo loop is now described in more detail.

The supervisor, which transcends all levels of the sensory hierarchy determines how to use the resources of the sensory system to update the world model. This includes deciding the types of sensors to use, choosing the operating parameters of the sensors, etc. In the case of vision, it initiates the commands to the low-level modules to take and to process an image. The supervisor must be aware of the global goals of the control system to be able to predict where the robot will be at some future time.

The 3-D world model predicts the state of the workspace at some future time. Given an expected position and orientation of the camera by the supervisor, a perspective projection of the features of each object within the camera's field of view can be calculated. The major problem is occlusion. Some of the features of an object may be occluded by itself or by some other object. This is partially resolved using aspect graphs (Koenderink et al., 1979).

The 2-D world model (WM-I) now uses the expected location of the features in the image plane and the confidence about the workspace contents to calculate windows within which each feature is expected. Figure 1 shows that the lower levels of the sensory system have two paths: the vision path and the other sensors path.

The other sensors include proximity, force/torque, etc. At present, they require only one level of processing. The vision path, on the other hand, requires several hierarchically organized levels. First stage vision (FSV) controls the camera hardware and maintains synchronization between the camera and the flash units. Second stage vision A (SSVA) performs a connected components analysis on the image.

The next level, second stage vision B (SSVB), extracts features. This is the first level that is model driven. Windows, which are calculated by WM-I, are passed to SSVB. It is within these windows that SSVB searches for the expected features.

The features detected by SSVB are matched with those predicted by WM-I in the first multi-level database (MLD-I), which deals only with 2-D data. The information from vision as well as from the other sensors is combined to form a coherent representation. The matched predictions and empirical data are then sent to MLD-II, the 3-D multilevel database.

MLD-II is responsible for 3-D feature matching. Given the expectation from WM-II, MLD-II assigns the third dimension to the empirical data passed from MLD-I. From the matches, MLD-II calculates the pose error for each part viewed. The pose updates are sent to the 3-D world model to reflect the true state of the workspace. In this way, the servo loop is closed. WM-II now has better information with which future predictions can be made.

Occasionally, there may be unmatched features. This situation can occur if an object is truly unknown (e.g. wrench dropped in workspace), if the actual object location is significantly different than its prediction, or if the extraction algorithms erroneously detect or miss features. In these cases, the data is passed to the recognition module which attempts to match the empirical data with each of the known object models. This procedure takes longer than the normal object verification mode. It is hoped that the recognition module should be required infrequently. When the recognition module is successful, the 3-D world model is informed of the object identity and is used in subsequent predictions. For the case of a truly unknown object, the recognition module will never be able to assign a model. However, the object description from the empirical sensor data is sufficient to move the object.

In parallel with the matching operations used for naming objects, a separate process attempts to describe the environment in terms of the space that is occupied. This is done using an octree representation to decompose the volume of the work area (Hong et al., 1984). Each camera image is projected into the octree. The objects are intersected with objects already represented, while the background is projected to carve out volumes known to be empty. There are links between the spatial representation and a feature-based representation. They allow object names and attributes to be associated with regions in space.

The operation of the sensory system is described in greater detail in (Shneier et al., 1984a) while the workspace representation is described in (Shneier et al., 1984b). In the implementation of the sensory system, each of the modules in the architecture is a microprocessor which operates asynchronously. Since the crucial parameter of the sensory system is time, the organization of the 3-D world model must support all of the modules expecting data as rapidly as possible. The next section describes the model itself in detail.

4. 3-D WORLD MODEL

The description of the 3-D world model is divided into three parts. First, the philosophy of the representation is presented. This is followed by a description of the information stored in the representation. Finally, a description of the processing and implementation of the module is provided.

4.1. REPRESENTATION PHILOSOPHY

4.1.1. GENERIC OBJECT vs INSTANCES

A generic object is defined as the description of an object in object space. An instance, on the other hand, provides information unique to a specific part in the world. The reason for this distinction is memory space. The description of the object is relatively long while the instance need only have a pointer to the generic object description (CAD description) and some information concerning its precise location and orientation in the world. Consequently, each instance has a homogeneous matrix that transforms a part location from the standard orientation called "object" space into a position in the real world.

Every object must have a description in the model at all times. There may be several intermediate stages in the machining of a part. Each is considered to be a different part and must be given a description from the external database. This should not be too inconvenient since the system must know precisely how it intends to process the part.

4.1.2. MULTIPLE REPRESENTATIONS

One of the features of the generic object description is that it supports multiple representations. The main representation form is a boundary representation. Surfaces are built from edges which are in turn constructed from vertices. However, other representations are also supported to facilitate processing.

The first "extra" representation is a geometric representation. For many simple shapes, this is the representation of preference. For example, any quadric surface can be represented with a point, two vectors, and three scalars (Goldman, 1983). This is quite compact and it easy to use. Similarly, for conic sections, the curves can be represented by a point, two vectors, and two scalars. This representation is especially useful in matching. The parameters of the representation make geometric sense; the user can easily comprehend the meaning of the parameters.

For more complex shapes, the geometric representation method becomes cumbersome because many quadric patches would be required for adequate representation. Consequently, a parametric representation should also be provided. Rational bicubic b-splines have been chosen because they seem to offer a reasonable compromise between representational accuracy and size.

In order to accelerate the projection of the CAD description into the octree, the 3-D world model also stores an octree representation in "object" space. This is transformed to the real world by the pose matrix of the instance and projected into the octree during initialization.

4.2. THE SENSORY SYSTEM WORLD MODEL REPRESENTATION

There are two parts to the object representation. The first is concerned with the description of generic objects. This information is in "object" coordinates and never changes. The second form represents information unique to a particular instance of a generic part. The information for the instances of any object can change at any time since the pose is updated constantly. In all cases, the representations are dynamically allocated in memory to ensure efficient memory use. Otherwise, the worst case size of an object representation must be used for all objects.

4.2.1. GENERIC PART DESCRIPTIONS IN THE SENSORY SYSTEM

The object representation shown in Table I is the highest hierarchical level in the boundary representation. It stores the generic part number which differentiates one object from all others. Furthermore, it contains the previous and the next identity of the object, i.e., the generic number of the part before the current machining step and its generic part number after the planned machining step.

The object level of representation contains the pointers to the beginnings of each of the hierarchically organized representations. Each representation is essentially a singly linked list. The last entry in the first vertex structure, for example, is a pointer to the next vertex. This is continued until all of the structures in the representation have been exhausted. Then, as is the common method for database processes, the pointer is NULL.

The object representation supports several representations aside from the vertex, edge, and surface representations. While these representations have redundant information, they accelerate the real-time aspects of the system. Aspects graphs (Koenderink et al., 1979) can be used to solve the hidden surface problem off-line. This greatly simplifies the problem of predicting which features should be detected by the sensors.

Another representation is concerned with features. Given a set of objects in the robot workspace, a subset from the set of all extractable features is chosen which best differentiates the objects. Each of the features in this set has an expected value in "object" space. A pointer is stored to the first feature/value pair.

Each of the other representations will be described using a bottom-up approach. The main representations,vertex, edge, and surface, will be described first. This will be followed by the aspect graph and the feature representations.

The vertex representation stores a vertex number, the 3-D location of the vertex, the length of the vertex, and a list of edges connecting to the vertex. This is shown in Table II.

The edge representation is slightly more complicated because it stores multiple representations as well as flags which speed up processing. Table III shows the contents of the representation. An edge can be represented as a "vertex" representation or a "parametric" representation. The vertex representation stores the starting and ending vertices of the edge, which can either be straight or curved. When an edge has a vertex representation, it is likely that information about the specific vertices will be needed. A pointer to the information can be accessed by other means. However, explicitly storing the vertex pointers accelerates the data access.

There are some edges, such as drill holes, which do not have vertices. In this case a parametric edge representation is more appropriate. There are two parametric forms to describe an edge. For edges which are first or second order curves, the easiest parametric form is "geometric" in nature. All conic section curves can be represented by one point, two orthorgonal unit vectors and two scalars. More complex curves, on the other hand, can be approximated by splines. A current research project involves organizing spline representations hierarchically so that a suitable level of representational accuracy be be used by a given module. For example, inspection modules may require a great deal more accuracy than vision modules.

Table IV shows the information stored in the surface representation. The surface number is followed by the containment surface. This information is useful if a hole is to be represented. The hole is actually considered as a surface contained inside of

another surface.

4.2.2. INSTANCE DESCRIPTION IN THE SENSORY SYSTEM

The generic object description stores the information which is constant for the object in "object" space. However, a mechanism is needed which describes the objects in the workspace. It would be prohibitively expensive to store a separate representation for each object. Consequently, a more economical solution has been adopted.

The generic object description stores information about the part in "object" space. The instance description provides the information which transforms "object" space into "world" coordinates. Consequently, relatively little information needs to be stored for each instance.

Each instance of a generic object in the workspace is unique. It contains information concerning its instance number. It also includes the time when this instance was instantiated in the workspace and the last time it was modified. Then there are three time/transformation pairs. A homogeneous tranformation is needed which transforms "object" space into "world" coordinates. Associated with this transformation is the time when it was valid. This is useful in predicting the location of moving objects. Some of the changes in the expected values of the sensors occur because the sensors move with the robot. Knowledge of the robot motion is sufficient to predict this change. However, if the objects are also moving, the expectations of object location must take this motion into account, also.

4.3. IMPLEMENTATION AND PROCESSING IN THE 3-D WORLD MODEL

The three dimensional world model is written in "C" and takes advantage of the use of structures in the language. There are several implementation considerations which effect the speed at which the processor can supply information.

The three dimensional world model has several functions which it must perform. First, it must accept information from the factory database which declares the parts which will be of interest during the next task. Then, for those generic parts which are not in the model, it must accept the model data and place into memory. The process of placing the model into memory will be called instantiation.

The speed at which information can be retrieved from the model is the most important consideration. Consequently, the models must be stored in memory not on disk. This obviously limits the number of models which can simultaneously reside in memory. At the beginning of a task, the factory database informs the sensory system of the models which will be used. The three dimensional world model checks to see if there is enough room to allocate space for each model. If there is enough space, the model is allocated. If not,

the model searches for old information to delete before the current model is allocated.

At any time, new instances of any of the generic part types may need instantiation into the model. This can occur because of an initial prediction of what is expected in the world based on information from the factory database or when a "new" part is created by a machine process. Recall that there must be a model for every partially created part. It can also occur when an unknown object is recognized. Also, when the robot moves parts out of the world, these objects must be removed from the world model.

The 3-D world model predicts the pose of all objects in the world. Initially, the factory database provides an initial estimate of the pose. This pose is updated by the data processed from the sensors. Since several pose matrices with the time at which they were valid are stored in the instance representation, it is possible to predict the new pose taking both the object and robot motion into account.

The 3-D world model also predicts the features of the workspace. These are the features which are matched with the empirical features extracted from the sensory data. In order to do this, the 3-D world model must determine which features are occluded. This is done using the aspect graphs to determine self-occlusion, i.e., the object itself occludes the projection of a feature. It also requires the octree to obtain the spatial relationships between objects so that occlusion of features by other objects is taken into account.

5. DISCUSSION

One of the main applications for representing solids is in the development of CAD descriptions of solids. There are many systems available which interactively create part descriptions. These include Computervision, Synthavision, McAuto, PADL, etc. In most cases, the goal of the system is to create a part description which is sufficient to manufacture the part. Since there is human interaction, the speed of the processing is not crucial. It only needs to be rapid enough so that the operator does not experience annoying delays. The requirements for the NBS sensory system are quite different. The "other user" is a computer not a person. The rate at which data can be accessed determines the servo rate of the system, i.e., the rate at which the sensory system can servo the model of the workspace to the actual condition of the workspace. The model always lags in time. A rapid servo rate ensures that the 3-D model represents the nature of the workspace in the not too distant past.

Badler et al., (1978) were concerned with the representation of 3-D objects for computer graphics and computer vision applications. It was clear that the choice of representation and the intended application were strongly coupled. Each representation had particular advantages and it would clearly be desirable to convert one representation into another. Unfortunately, many of these conversions were not obvious. In their paper, the authors discussed many of the alternative representations but drew no conclusions about the best set of representations for a system.

Aggarwal et al., (1981) discussed methods for representing 3-D objects for computer vision. They discussed both the methods for acquiring 3-D data and the ways in which the objects can be represented. Representations were divided into two broad classes: "object centered" representations and "observer centered" representations. Most methods which acquired 3-D data, such as laser scanning, photometric stereo, etc., employed "observer centered" representations. However, most solid modelling methods, such as CSG or boundary representations employed "object centered" representations. They recognized the need for efficient conversion between the two representations before a complete system can be developed but they did not supply a method for the conversion. The NBS sensory system bridges these two representations by using the generic object concept as the "object centered" representation and the instance concept as the "observer centered" data.

Recently, Shapiro et al., (1984) presented a hierarchical relational model for part inspection. After the part is placed in a jig, vision and tactile sensors perform measurements for quality control. This is similar to the NBS sensory system in two ways. The system employs several sensors and uses a boundary representation. However, there are several major differences. First, the description is not adaptive. If the part is slightly misaligned, it is not clear how the system can correct itself. The NBS sensory system, on the other hand, is constantly servoeing the model with the empirical data obtained from the sensors. Second, the system can only work with known parts. The NBS system is capable of representing unknown parts. Third, it is not clear how rapidly the system can access the data stored in the database. One of the major objectives of the NBS system is real-time operation.

6. CONCLUSIONS

A representation for the 3-D workspace of a robot has been presented which combines a verbose boundary representation with an instance representation. The 3-D model predicts features about the workspace. It then projects these features to 2-D where they are matched with sensory data. In this way, the sensory system employs

model-driven processing to keep the 3-D model of the workspace in registration with the true nature of the workspace.

7. REFERENCES

J. K. Aggarwal, L. S. Davis, W. N. Martin, and J. W. Roach, "Survey: representation methods for three-dimensional objects," Progress in Pattern Recognition, L. N. Kanal and A. Rosenfeld, eds., (New York: North-Holland, 1981), pp. 377-391.

N. Badler and R. Bajcsy, "Three dimensional representation for computer graphics and computer vision," ACM Computer Graphics 12, 3, Aug., 1978, pp. 153-160.

A. Baer, C. Eastman, and M. Henrion, "Geometric modelling: a survey," Computer-Aided Design, 11, 5, Sept., 1979, pp. 253-272.

A. J. Barbera, M. L. Fitzgerald, J. S. Albus, and L. S. Haynes, "RCS: the NBS real-time control system," Proc. Robots 8 Conference, Detroit, June, 1984.

J. W. Boyse and J. E. Gilchrist, "GMSolid: interactive modeling for design and analysis of solids," IEEE Trans. on Computer Graphics and Applications, 2, 2, March, 1982, pp. 27-40.

C. M. Brown, PADL-2: a technical summary. IEEE Computer Graphics and Applications, 2, 2, March 1982, 69-84.

R. N. Goldman, "Two approaches to a computer model for quadric surfaces," IEEE Trans. on Computer Graphics and Applications, 3, 6, Sept., 1983, pp. 21-24.

R. Goldstein and L. Malin, "3-D modelling with the Synthavision system." Proc. 1st Annual Conf. Computer Graphics in CAD/CAM Systems, Cambridge, MA., April, 1979, pp. 244-247.

T-H. Hong and M. Shneier, Describing a robot's workspace using a sequence of views from a moving camera. Industrial Systems Division, National Bureau of Standards, Washington, D.C., 1984. (in preparation).

E. W. Kent, A hierarchical, model-driven, vision system for sensory-interactive robotics. Proc. Compsac '82, Chicago, November 1982.

J. J. Koenderink and A. J. van Doorn, The internal

representation of solid shape with respect to vision, Biol. Cybernetics 32, 1979.

W. Myers, "An industrial perspective on solid modeling," IEEE Trans. on Computer Graphics and Applications, 2, 2, March, 1982, pp. 86-97.

A. A. G. Requicha, "Representations for rigid solids: theory, methods and systems", Computing Surveys, 12, 4, Dec. 1980, pp. 437-464.

A. A. G. Requicha and H. B. Voelcker, "Solid modeling: current status and research directions," IEEE Trans. on Computer Graphics and Applications, 3, 7, October, 1983, pp. 25-37.

L. G. Shapiro and R. M. Haralick, "A hierarchical relational model for automated inspection tasks," Proc. Int. Conf. on Robotics, Atlanta, March, 1984, pp. 70-77.

M. O. Shneier, R. Lumia, and E. W. Kent, "Model-based strategies for high- level robot vision," (in preparation, 1984a).

M. O. Shneier, E. W. Kent, and P. Mansbach, Representing workspace and model knowledge for a robot with mobile sensors, Proc. 7th International Conference on Pattern Recognition, Montreal, 1984b, 199-202.

D. Waltz, "Understanding line drawings on scenes with shadows," Psychology of Computer Vision, P. Winston, ed., (New York, McGraw-Hill, 1975), pp. 19-59.

Table I -- Object representations

1. generic part number
2. old generic part number
   previous processing identity
3. new generic part number
   identity after current operation
4. number of instances in the world
5. pointer to a linked list of instance information
6. number of vertices
7. pointer to a linked list of vertex representations
8. number of edges
9. pointer to a linked list of edge representations
10. number of surfaces
11. pointer to a linked list of surface representations
12. number of aspect graphs
13. pointer to a linked list of aspect graph representations
14. number of features
15. pointer to a linked list of features
16. pointer to the next object in the world

Table II -- Vertex Representation

1. vertex number
2. 3-D location (x,y,z)
3. pointer to linked list of edges which connect to this vertex
4. pointer to the next vertex rep

Table III -- Edge Representation

1. edge number
2. type of representation
   VERTREP -- vertex rep
   PARAREP -- parametric rep
3. shape of edge -- STRAIGHT or CURVED
4. starting vertex (if using VERTREP)
5. ending vertex (if using VERTREP)
6. pointer to starting vertex rep
   (if using VERTREP)
7. pointer to ending vertex rep
   (if using VERTREP)
8. edge length
9. pointer to geometric PARAREP
10. pointer to spline rep of edge
11. pointer to surface left of edge
12. pointer to surface right of edge
13. angle between adjacent surfaces
14. pointer to the next edge rep

Table IV -- Surface representation

1. surface number
2. containment surface
   useful for hole representation
3. type of surface--SOLID or HOLE
4. shape of surface--PLANAR or CURVED
5. reflectance
6. surface area
7. pointer to the surface normal
   (for PLANAR surfaces)
8. number of bounding edges
9. pointer to linked list of edge
   numbers which bound the surface
10. pointer to geometric surface rep
11. pointer to parametric surface rep
    (bicubic b-splines)
12. number of holes in this surface
13. pointer to a linked list of holes
14. pointer to the next surface rep

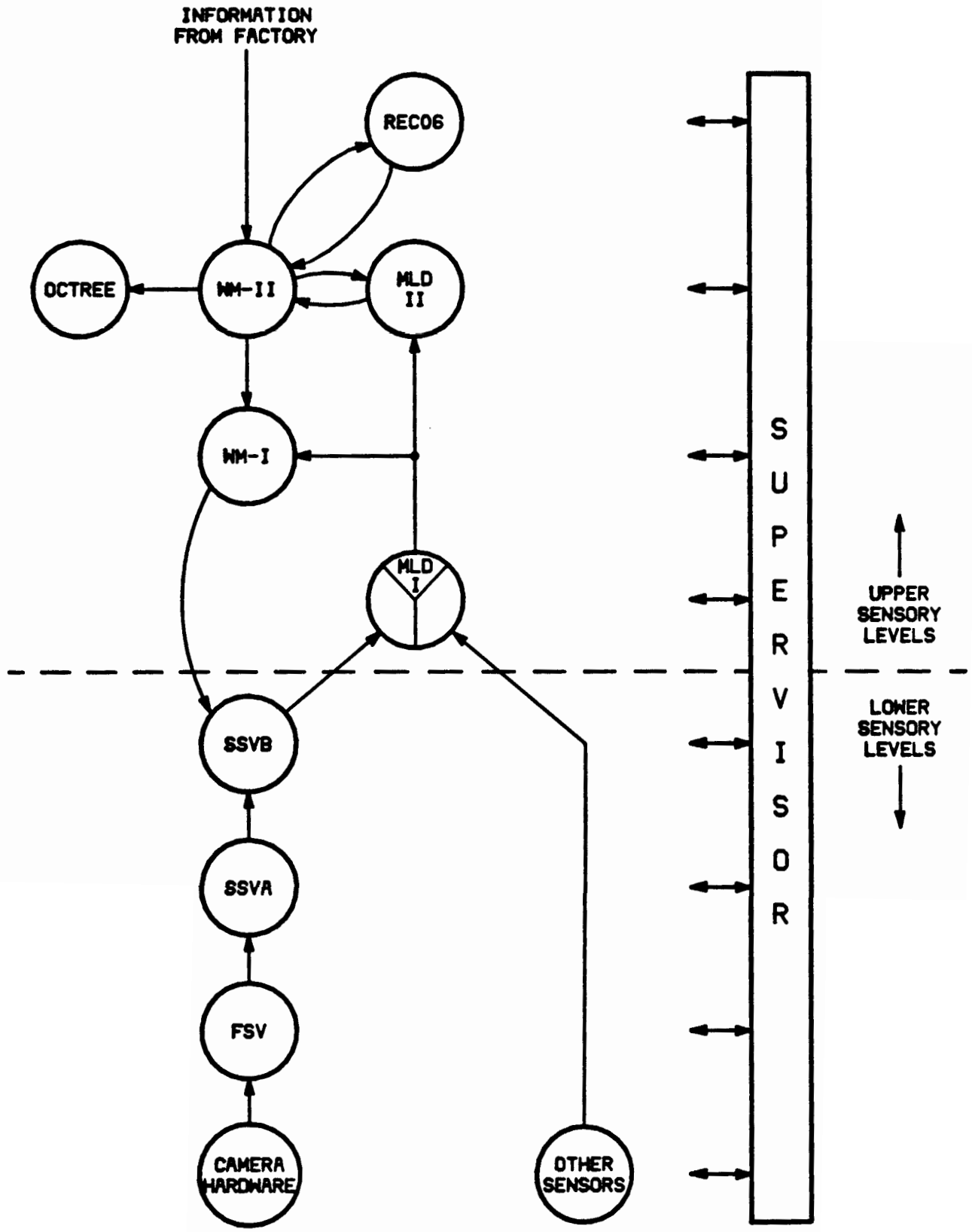

FIG. I - SENSORY SYSTEM MODULES