CMU-CS-78-113

REPRESENTATION OF THREE-DIMENSIONAL OBJECTS

D. R. Reddy and S. Rubin

April, 1978

Department of Computer Science
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

This work was supported by the Advanced Research Projects Agency of the Office of the Secretary of Defense under contract number F44620-73-C-0074 and is monitored by the Air Force Office of Scientific Research.

REPRESENTATION OF THREE-DIMENSIONAL OBJECTS

D. R. Reddy and S. Rubin

ABSTRACT

This paper describes a system for generation and representation of computer models of arbitrary three dimensional objects. Three representations are explored which have varying time and space tradeoffs. These representations, which are tailored for irregular 3-D objects, store every input point exactly by using a tree structuring of the object space. Their advantage over more common representations is that they make perspective and hidden-surface elimination less difficult. In addition, one of the representations simplifies the storage of dynamic objects and objects with redundant sub-parts.

I. INTRODUCTION

Many tasks require a three dimensional computer model of an object in order to perform examination and simulation. A typical example of this need is the modelling of biological processes where a computer model can facilitate detailed examinations from viewpoints and distances that are otherwise impossible to obtain. In addition, these computer models allow experimentation and simulation that would normally be infeasable. This paper describes a system that can generate and represent computer models of highly irregular objects. Actual system experimentation was done with the neuronal structures of a lobster (see Figure 1), but we feel that these techniques are applicable to all computer modelling tasks.

Generation of the model is accomplished with a set of serial sections that describe the object. Each serial section is a cross sectional view of the intersection of the object with a plane running through the object. The planes are parallel to each other and are equidistant through the object. In addition, the planes are spaced

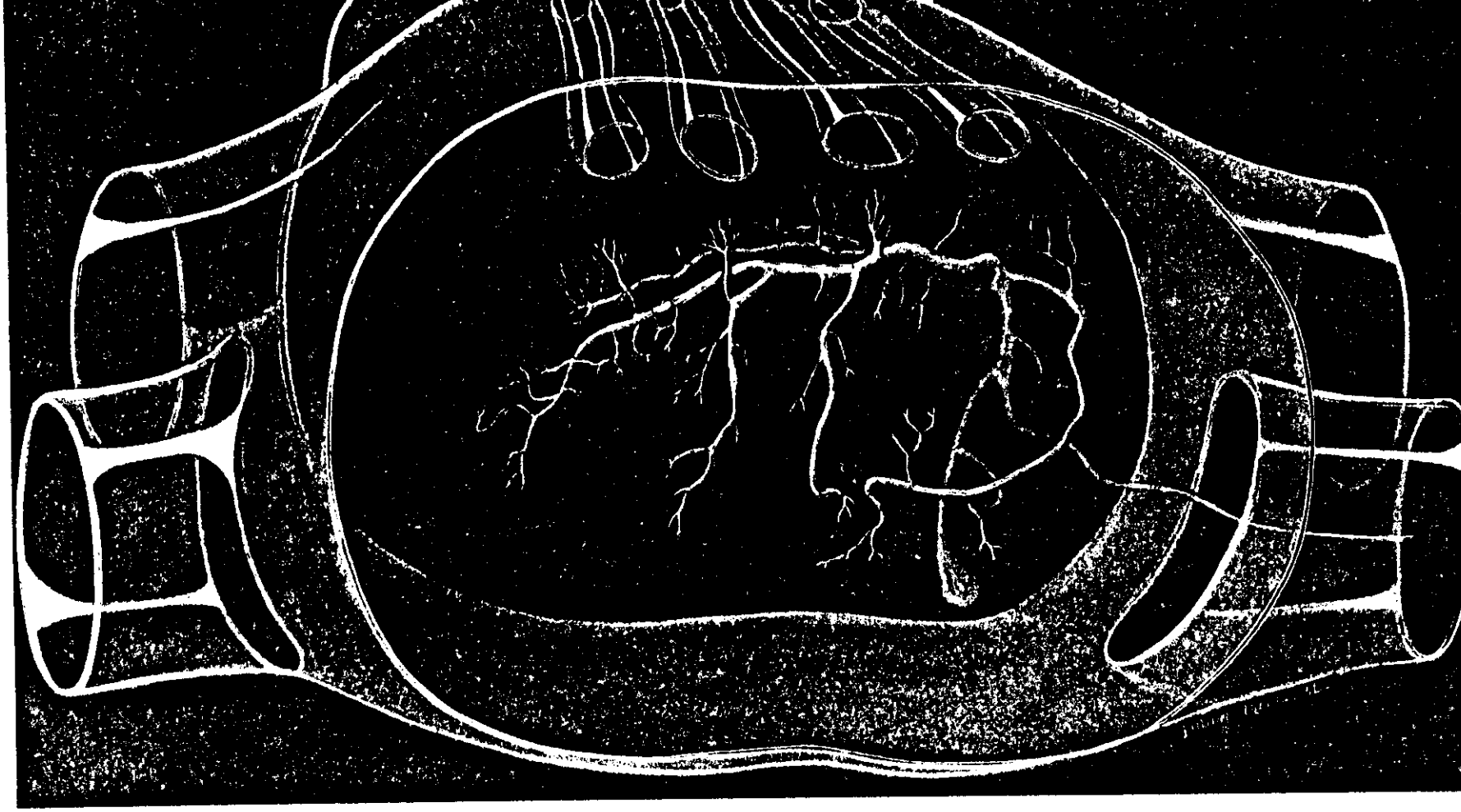

Figure 1. Lobster Neuronal Structure

This is an artist's conception of the swimmerette ganglion of a Lobster. This neuronal structure consists of an outer shell, four large fibers, and about 75 dendrites (only one shown).

closely enough to be able to give full resolution along their normal axis. Since this type of input describes every point in the object, it is well suited to highly irregular scenes and can be used as input when modelling the external *and* internal details.

There are many ways to extract serial sections from a real object. The object can be physically sliced and the slices photographed (Reddy et al, 1973; Selverston, 1973; Woolsey et al, 1972). In addition, the object can be X-rayed in order to construct serial section information without harming the object (Prewitt, 1976). Another method of obtaining three dimensional information is to take stereo views of the object and re-construct from there (Gennery, 1977). The main limitation of this approach is that it does not provide a complete description of the object.

Our representation of an object is based on a precise storage of every point in the object. It uses tree structured data to isolate the object from its surrounding empty space thus allowing the modelled object to be stored efficiently and accessed quickly. The data structure facilitates random accessing so the model is well suited to raster displays and yields inexpensive hidden-surface elimination. In addition, the nature of the tree structuring allows for dynamic object modelling, ease of storage for common sub-parts, and very inexpensive perspective transformations. The model falls short only when high resolution is demanded from low resolution input. This is because the model stores object points, so the high resolution image will be extremely grainy. However, we feel that this model is powerful enough to be used for any three dimensional representation task where accuracy is required.

Generating views of three-dimensional objects has been a topic of interest for some time. Roberts (1963) explored hidden-surface elimination in the object space. In addition, there have been many scan line algorithms for hidden-surface elimination

(Wylie et al, 1967; Bouknight, 1970; Watkins, 1970). Sutherland, Sproull, and Shumacker (1974) discuss these in more detail in their paper on hidden-surface algorithms.

The remainder of this paper describes the features of our system for modelling three dimensional objects. Section II presents some of the problems associated with serial section input. Sections III and IV are discussions of the three representations that were examined and an evaluation that shows why one is best. Section V is a description of the display techniques that can be used with the optimal representation.

II. ASPECTS OF MODEL GENERATION

When an object is to be entered into the computer via serial sections, it is sliced and the slices are photographed. In cases where physical destruction of the object is not possible, X-ray diffraction techniques can be used to obtain the position of all points in the object. In our case, however, serial section photographs are more readily available, so we have explored their use.

There are some interesting problems that arise when isolated serial section photographs are presented to the computer. For example, suppose that after all of the serial sections have been digitized, additional information is required. Each slide must be placed on the scanner for further digitization. The problem is that the physical placement must be correct so that the new information can be registered properly with the old information. To achieve this, two fiducial marks are placed on each slide so that the machine can generate the proper rotation and translation offset for the new data. Once the fiducial marks have been identified, it is trivial to generate a transformation matrix that will register the new information with the old.

The most serious problem involved with separate photographs of serial sections is aligning them with each other. Since each photograph is independant of the others, the complete collection must be lined up before the internal model is built. This process is automated to a modest extent, but requires human interaction to double-check the alignment. The machine can line up two serial sections by matching the outlines of the object on each section. This is done by choosing a center point and drawing a series of lines from that point to the edge of the object. If the lines are evenly distributed about the center, then the set of distances to the edge will define an orientation for that section. The alignment of two sections is accomplished by finding the least-squares fit between their orientations. Formally, the machine first finds the center of gravity of the object and draws N radii about that point to the edge. Assuming that the distances to the edge are Di_1, Di_2, ..., Di_N for serial section i, then two serial sections, j and k, have the following least-squares fit for an angular offset of r radians:

$$F(r) = \sum_{m=1}^{N-x} (Dj_m - Dk_{m+x})^2 + \sum_{m=N-x+1}^{N} (Dj_m - Dk_{m-N+x})^2$$

where $x = rN/2\pi$. The optimal angular offset is r, such that $\forall s \neq r$, $F(r) \leq F(s)$. Our system uses N = 180 and the alignment proceeds very quickly when all integral values of r are tested. It usually requires no adjustment and only fails when the object outline changes rapidly between slides. Figure 2 shows a sample result of the automatic alignment process on two serial sections.

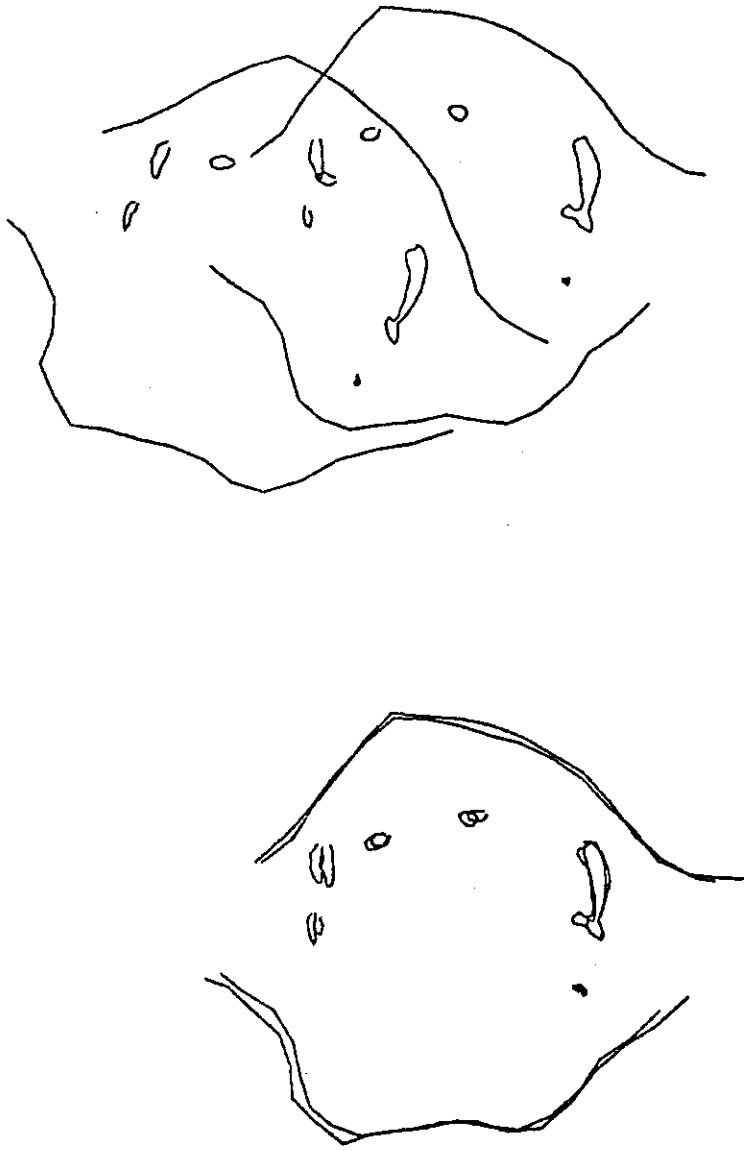

Figure 2. Automatic Alignment

The top figure shows two sections of the neuronal structure of a lobster as they are digitized. The bottom figure shows them after machine alignment.

III. THREE DIMENSIONAL MODELS

In our search for a powerful three dimensional model, we examined three different techniques which can store every point in the object space by using tree structures. Three-dimensional models which store every object space point are frequently excessive in their space requirements. For example, if an object can be modeled by dividing it into 1000 parts along each axis, then the internal representation requires 10^9 points. This is not feasible because of the high storage costs and relatively low resolution. The three techniques presented here are variations of this scheme that are storage efficient yet can obtain accuracy equal to or greater than the 10^9 example. They are based on the assumption that since most of the object space is empty, storage requirements will not be excessive if representation of empty space is avoided. This can be viewed as "object coherence" (Sutherland et. al. (1974)). The models impose no unnatural structure on the object and they retain the precision of detail that is present in the input. In addition, these models are designed for random access of points and so are tailored towards display on raster screens. The random access yields hidden-surface elimination at virtually no cost since display is done by searching along the line of sight into the object. (Note that a random access structure can be viewed in the image space, where all object points are projected to the screen, or the object space, where all screen points are projected into the object. We have chosen the object space algorithm because it is more independant of the object complexity and does not require a frame buffer for hidden-surface elimination.) The remainder of this section describes the three models in detail.

POLYGONAL SURFACES (AN ASIDE)

Our task domain is that of highly complex objects. Therefore, we have rejected

the polygonal surfaces approach which stores the object as a collection of plane segments. Display time for such object spaces grows exponentially with the number of surfaces, and we are assuming that the object to be modelled is highly complex. For this reason, we have chosen alternate models that can store equivalent representations and display with much more ease.

EQUAL SUBDIVISIONS OF THE OBJECT SPACE

The first representation uses a successive subdivision method to isolate and eliminate empty spaces (see Figure 3). This subdivision has the Warnock (1969) flavor, but deals with object space subdivison rather than image space subdivision. Given a sparsely occupied cubic space, it is subdivided into a number of sub-cubes. Each of these sub-cubes will either be empty or have useful information. Those sub-cubes that are not empty are further subdivided into sub-sub-cubes and tested for content. The subdivision process can proceed for as many levels as desired, however one usually stops when the size of the points in the lowest level of subdivision is on the order of .1% of the object size. The properties of the object are then stored at the bottom level. This is the simplest of the three representations, and is the only one that contains no special knowledge about the object.

To see why this model is so simple to generate and use, let us examine the accessing technique for a single point in the object space. We will assume that each level of subdivision divides the three axes in half so that each sub-cube has one eighth of the volume of its parent cube. Given that the object space co-ordinates range from 0 to 1023 in all three axes, we will obtain the contents of the point at (X, Y, Z) where X, Y, and Z are all ten-bit integers. At the top level of the model is a vector of eight elements, one for each sub-cube in the object space. Indexing this vector can be done

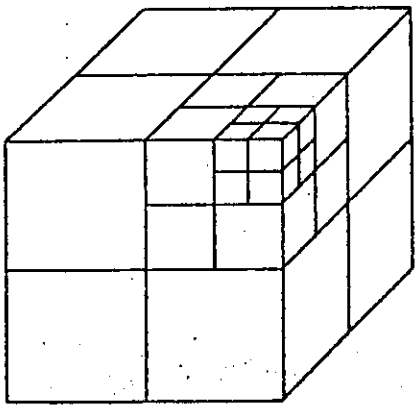

Figure 3. Equal Subdivisions

Each level of subdivision divides the remaining object space equally along the X, Y, and Z axes

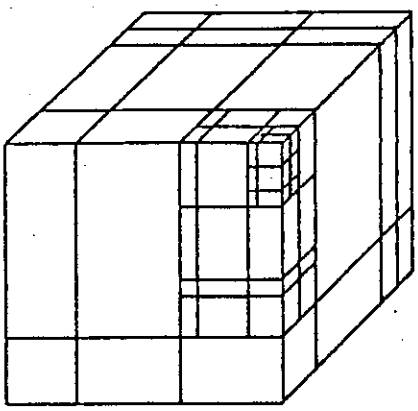

Figure 4. Unequal Subdivisions

Each level of subdivision divides the remaining object space with arbitrarily placed planes perpendicular to the X, Y, and Z axes

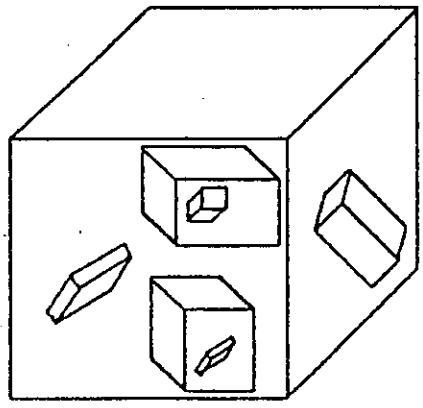

Figure 5. Arbirtarily Oriented Parallelopipeds

Each level of subdivision divides the remaining object space into arbitrarily oriented parallelopipeds which are defined as transformations within the current level of subdivision

by concatenating the high bit of the X co-ordinate with the high bit of the Y co-ordinate and the high bit of the Z co-ordinate. This three-bit value selects a sub-cube from the vector. If that sub-cube is to be further sub-divided, then the second highest bits of the X, Y, and Z are concatenated to select a sub-sub-cube. The bit manipulation is simple to implement at a hardware level and proceeds very quickly since it requires no searching. It need only proceed until it finds an empty cube or a full cube.

UNEQUAL SUBDIVISION OF THE OBJECT SPACE

The next representation is an extension of the first one. It employs some knowledge about the particular scene being represented so that its data structure can be more concise. In this model, the space to be subdivided is broken into rectangular parallelopipeds as opposed to cubes. Division of the space is done with planes perpendicular to the X, Y, and Z axes, however the planes are not equally spaced and there can be a different number of planes along each axis (see Figure 4). As with the previous model, the top level of subdivision does not provide the final detail, just a gross segmentation of the structure. Because the subdivision is variable along the axis, storage can be saved by intelligent subdivision. For example, if a very small object is to be represented in the middle of a large empty area (i.e. a fly in a room) the equal subdivision model will have to traverse many levels of the tree before it gets to the detail of the fly. Using unequal subdivisions, two closely spaced planes along each axis will exactly single out the fly so that the next level of subdivision can begin at the proper detail.

Point accessing is more difficult with this model because search techniques are required to move down the correct branch of the tree. Each level of sub-division

needs three vectors that indicate the location of the X, Y, and Z planes which make up the sub-spaces. The X, Y, and Z components of the desired point must be located in these vectors so that the indices can be used to select the proper sub-space.

ARBITRARY GROUPING OF THE OBJECT SPACE

The final representation also allows for intelligent subdivision with the advantage of permitting easy modification to the structure and allowing repetition without wasted space. In this model, which we feel is the best, the structure is broken down into rectangular parallelopipeds that are not necessarily aligned with an axis (see Figure 5). The parallelopipeds can be any size and orientation but ***any point that is not in one of these parallelopipeds is empty.*** In this scheme, therefore, one finds a point's attributes by determining which parallelopiped it falls in. If the point is found in a parallelopiped, then the next level of detail is examined. This level has another set of parallelopipeds that isolate the non-empty parts within it. Sub-levels in this model operate in the co-ordinate space of the parallelopiped that encloses them, so accessing of points is done recursively.

Let us examine the point accessing method of this representation in more detail. The system is presented with a point (X, Y, Z) that lies in the object space. At the top level of the structure, there are N transformation matrices, T_1 through T_N, and each is a 4x4 transformation that converts the object space point into the co-ordinate system of its parallelopiped. In this new system, the point (0, 0, 0) is at one corner of the sub-space, and the point (UX, UY, UZ) is at the diagonally opposite corner. The object space point is within parallelopiped i if, after transforming (X, Y, Z) through T_i to become (X', Y', Z'),

$$0 \le X' \le UX_i \wedge 0 \le Y' \le UY_i \wedge 0 \le Z' \le UZ_i$$

If none of the parallelopipeds at a given level is found to contain the requested point, then that point is reported to be empty. If the point falls within one of the parallelopipeds, then there is a possibility that it is non-empty. To find out, all of the sub-parallelopipeds within this new object space must be searched. The algorithm recurses and the point (X', Y', Z') is extracted from the sub-space. This recursion continues until a level is reached where there is no more detail to the object. At this level, a point that falls within a parallelopiped is non-empty, and the properties of the point are extracted.

IV. EVALUATION

There are a number of tradeoff issues involved in storing and accessing the above representations. Each model requires successively more complex operations to access a point, yet successively less storage to represent an object because it encodes more information about the structure. In addition, if the more complex models are properly built, point accessing and image building actually takes less time due to the efficiency of storage. We have found that storage is more critical than accessing time because it is usually necessary to use secondary storage to hold an entire object representation: the cost of swapping a large but simple data structure is higher than that of a small and complex structure and accessing considerations become secondary.

We have found that the three representations require linearly increasing amounts of time to access a point and that the most complex representation takes five times as long as the least complex one. However we are still able to get better performance from the most complex representation when run on our PDP-10 computer. Another issue that indicates use of the most complex representation is display side-

effects. The next section will show a number of display side-effects which make the most complex representation optimal.

V. DISPLAY

We have chosen the complex representation as the most appropriate technique for accurately storing a three dimensional object because it is the most efficient. Like the others, it has the advantage of complete representation which allows views to be constructed from any point, even within the object. In addition, there are a number of display issues that are simplified when using this model. In fact, this model has so many useful features, that we can place it at the top level of detail and place other models within it at lower levels of detail. Using this composite representation, our model imparts its display advantages and allows the special features of the sub-representation to be used.

When using the complex representation to display a view of an object on a raster screen, the display program projects lines from the viewpoint, through the screen point to be illuminated, into the object space. The equation of the line is first transformed to find out which of the top level parallelopipeds it intersects. When a hit is found, the transformed line is matched against all of the sub-parallelopipeds in a recursive manner until the screen point is filled. To display with hidden-surfaces eliminated, all of the parallelopipeds at a level must be examined to see which one is closest to the viewer. This is quite inexpensive since there are usually very few parallelopipeds on a level (on the order of ten).

Other side effects of our model are common storage of redundant sub-representations, dynamic modelling, and inexpensive perspective. Common storage of

redundant sub-representations is implemented by allowing sub-objects that appear frequently in an object to be represented once. Each sub-object is then identified by its own transformation matrix and the common sub-description. Needless to say, the selection of parallelopiped enclosings must be done carefully so that logical units in the object space are enclosed separately and completely.

Our model also allows dynamic objects to be represented by enclosing the moving parts in their own parallelopiped. When a part of the object moves, the model can be updated by changing a few values in the transformation matrix. When two parallelopipeds intersect, the search must proceed in parallel down to the lowest level of detail (assuming that the object points themselves do not overlap). Alternatively, the internal model can be restructured to eliminate overlapped parallelopipeds.

Perspective transformations are virtually free in this system because the perspective matrix can be pre-multiplied into the transformation matrices at the top level of the object description. This is relatively inexpensive due to the small number of matrices at a level. To generate a perspective display, each point on the viewing screen is searched in an object space that has already been transformed: the object points can be plotted directly. In addition, other transformations of the object space can also be done in this step for a fraction of the cost normally required.

There is only one drawback to our model, a problem that is encountered in all of the representations presented here, and that is finite resolution. When using polygonal surfaces to represent an object, views can be taken at arbitrary magnifications and the details of the polygons will be the same. When using complex representation models however, the resolution of the screen must not be greater than the resolution of the object space or else adjacent screen points will be drawn from

the same object space point. This will result in jagged lines on the screen at the edges of each feature. One solution to this problem is to store great detail in those parts of the object that are subject to high magnification. Thus, an object might be stored with five levels of subdivision, but a facet of the object that is subjected to detailed examination would employ two or three more levels of detail before the object point descriptions are stored. Another solution is to use some interpolation algorithm to smooth the jagged edges. The best solution, however, is to use our model at the top levels of detail and to use some other representation at the lower levels. This avoids the "jaggies" problem without damaging the effectiveness of the model.

VI. CONCLUSIONS

We have developed a complete system for storing and retrieving three dimensional objects. This includes some new techniques for entering the description of an object using serial sections and some new models for storing the object.

Alignment of serial sections has been automated quite successfully using object outlines. We have developed a precise internal representation that is well suited to the currently emerging technology of raster displays. This model can handle dynamic objects, repetitive sub-objects, and inexpensive perspective and hidden-surface elimination. We feel that this storage model is powerful enough to work well with our task, be useful for other tasks, and tackle some of the hardest problems in three dimensional modelling.

VII. REFERENCES

Bouknight, W.J. (1970). A Procedure for Generation of Three-Dimensional Half-toned Computer Graphics Representations, CACM, 13, 9, 527.

Gennery, D.B. (1977). A Stereo Vision System for an Autonomous Vehicle, IJCAI-77, 576-582.

Prewitt, J. (1976). New Vistas in Computer Tomography, from Digital Processing of Biomedical Images, Preston, K. and Onoe, M., Eds, Plenum Press.

Reddy, D.R., Davis, W.J., Ohlander, R.B., Bihary, D.J. (1973). Computer Analysis of Neuronal Structure, from Intracellular Staining in Neurobiology, edited by Kater, S.B. and Nicholson, C., Springer-Verlag, 227-253.

Roberts, L.G. (1963). Machine Perception of Three-Dimensional Solids, MIT Lincoln Laboratory, TR 315.

Selverston, A.I. (1973). The Use of Intracellular Dye Injection in the Study of Small Neural Networks, from Intracellular Staining in Neurobiology, edited by Kater, S.B., and Nicholson, C., Springer-Verlag, 255-280.

Sutherland, I.E., Sproull, R.F., and Shumacker, R.A. (1974). A Characterization of Ten Hidden-Surface Algorithms, Computing Surveys, Volume 6 Number 1, 1-55.

Warnock, J.E. (1969). A Hidden-Surface Algorithm for Computer-Generated Halftone Pictures, Computer Science Department, University of Utah, TR 4-15.

Watkins, G.S. (1970). A Real-Time Visible Surface Algorithm, Computer Science Department, University of Utah, UTECH-CSC-70-101.

Woolsey, T.A., Cowan, D.F., Dierker, M.L., and Linn, C.M.S. (1972). Computer Analysis of Golgi Impregnated Neurons, Washington University, St. Louis, presented at the Second Annual Meeting of the Society for Neuroscience, Houston, Texas.

Wylie, C., Romney, G.W., Evans, D.C., Erdahl, A. (1967). Halftone Perspective Drawings by Computer, AFIPS Conference Procedings, Vol. 31, 49.