THE AUTOMATIC SENSING AND ANALYSIS OF THREE-DIMENSIONAL SURFACE POINTS FROM VISUAL SCENES

HENRY FUCHS

THE AUTOMATIC SENSING AND ANALYSIS OF THREE-DIMENSIONAL SURFACE POINTS FROM VISUAL SCENES

by

Henry Fuchs

A dissertation submitted to the faculty of the
University of Utah in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

Department of Computer Science

The University of Utah

August 1975

UNIVERSITY OF UTAH GRADUATE SCHOOL

SUPERVISORY COMMITTEE APPROVAL

of a dissertation submitted by

Henry Fuchs

I have read this dissertation and have found it to be of satisfactory quality for a doctoral degree.

June 3, 1975

Date

Robert P. Plummer
Chairman, Supervisory Committee

I have read this dissertation and have found it to be of satisfactory quality for a doctoral degree.

Date

Steven V. Coons
Member, Supervisory Committee

I have read this dissertation and have found it to be of satisfactory quality for a doctoral degree.

June 3, 1975

Date

Elliott I. Organick
Member, Supervisory Committee

UNIVERSITY OF UTAH GRADUATE SCHOOL

FINAL READING APPROVAL

To the Graduate Council of the University of Utah:

I have read the thesis of Henry Fuchs in its final form and have found that (1) its format, citations, and bibliographic style are consistent and acceptable; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the Supervisory Committee and is ready for submission to the Graduate School.

June 30, 1975
Date

Robert P. Plummer
Member, Supervisory Committee

Approved for the Major Department

Anthony C. Hearn
Chairman/Dean

Approved for the Graduate Council

Sterling M. McMurrin
Dean of the Graduate School

ACKNOWLEDGMENTS

I am deeply grateful to my thesis supervisor, Robert Plummer, who has been a constant source of assistance and encouragement throughout the course of this project.

I want to express my appreciation to the other members of the committee, Elliott Organick and Steve Coons, for many hours of discussions, suggestions and insights -- as well as for responding so quickly and constructively to early drafts of this dissertation.

I would like to thank Ivan Sutherland, not only for being the initial inspiration for the project, but also for serving as a committee member while he was still at Utah.

I would like to express my appreciation to Rich Riesenfeld and Elaine Cohen for help and enthusiasm, especially at crucial periods in this project.

It should be noted that it is Mike Milochik to whom all praise is due for the excellent photography, in spite of some less-than-excellent sources from which to work.

Thanks are due to the many friends and fellow students, who willingly lent their bodies to be digitized in a none-too-comfortable environment.

Finally I would like to thank my parents for decades of faith, support, and encouragement -- without which none of this would have been possible.

TABLE OF CONTENTS

ABSTRACT

Described are the design and implementation of a new range-measuring sensing device and an associated software algorithm for constructing surface descriptions of arbitrary three-dimensional objects from single or multiple views.

The sensing device, which measures surface points from objects in its environment, is a computer-controlled, random-access, triangulating rangefinder with a mirror-deflected laser beam and revolving disc detectors.

The algorithm developed processes these surface points and generates, in a deterministic fashion, complete surface descriptions of all encountered objects. In its processing, the algorithm also detects parts of objects for which there is insufficient data, and can supply the sensing device with the control parameters needed to successfully measure the uncharted regions.

The resulting object descriptions are suitable for use in a number of areas, such as computer graphics, where the process of constructing object definitions has heretofore been very tedious. Together with the sensing device, this approach to object description can be utilized in a variety of scene analysis and pattern recognition applications which involve interaction with "real world", three-dimensional objects.

CHAPTER 1

INTRODUCTION

1.1 Problem Statement

Researchers in seemingly-diverse areas are often concerned with the acquisition of object descriptions. In artificial intelligence, for instance, a large part of most scene analysis systems is devoted to generating a description of objects in the system's working environment, whether this be a table-top scene of toy blocks, a rocky Martian surface, or a work-station on an auto assembly line.

In computer graphics, much time is spent attempting to create accurate pictorial images of real and imaginary objects. While the descriptions of imaginary objects are often created with the aid of an associated computer-aided design system, the descriptions of real objects usually has to be generated by laborious, largely-manual measurement techniques.

The interest in object descriptions is not limited to computer users. A prosthesis manufacturer may want to match the new artificial leg with the user's natural one, but they may not have the facilities to take more than a few basic measurements.

Researchers in artificial intelligence (specifically robotics) have been among the ones most intensely involved in the development of systems for the automatic acquisition of object descriptions. Most of their systems, however, have relied on a picture-oriented sensor, usually a TV camera. This report hopes to demonstrate that a significantly different kind of sensor, a computer-controlled rangefinder, may also prove useful for

some of these tasks. The design and implementation of such a rangefinding system is described. To demonstrate the feasibility of this approach, a simple scene-analysis algorithm is implemented, which can generate, solely from the range data, descriptions of objects in the sensor's field of view. It is hoped that this research will stimulate other attempts at sensing systems more readily adaptable to the computer than the human-oriented TV camera.

1.2 A Sampling of Previous Methods

1.2.1 Direct Manual Measurement

The most elementary method of digitizing objects is by direct, manual measurement. With the aid of yardsticks, plumb lines and calipers, a great many solid objects can be successfully measured, and the set of values later input to a computer system.

This idea of being able to specify an arbirarily complex three-dimensional object with a set of simple measurements is hardly a recent development. The Renaissance artist Leon Battista Alberti, in his book *Della Statua* , published in 1440, describes a method for the accurate measurement of the human form (Figure 1-1). He claimed that by using his method, different parts of the same statue could be constructed at separate places and would still be able to fit together [10].

Today's approach, still basically the same, is often to mark all points of interest -- "key" points -- on the surface of the original object and then measure the distance of each of these points from a common reference position (Figure 1-2). The surface is then defined as a topological net over these key points. Of course, many tedious hours must be spent to carefully measure the position of each selected point on the original

Fig. 1-1: Alberti's "Definer" (from [10])

Fig. 1-2: Manual measurement today (from [18])

object. The results are, however, often surprisingly effective. Although this method is not practical for serious, large-scale digitizing, it should be noted that it has several advantages over the other more sophisticated methods. Obviously, it requires almost no equipment -- hence no cost, except of course for manual labor. The resulting descriptions also tend to be very compact, since the user naturally wants to minimize the number of points that he has to measure. Although the less compact descriptions resulting from the more sophisticated automatic methods can often be trimmed in size according to some algorithm, it turns out that the subjective criteria used by humans are usually more effective.

1.2.2 Mechanical Moving Devices

An obvious next step to the simple manual approach is to substitute some computer sensing device for the user's calipers and yardsticks. The human user still has to define the surface points and their interconnections, but now he can just move some pointer around the object and tell the computer when the "current" position is of interest.

One device of this type is the so-called three-dimensional "crane" (Figure 1-3). This is a mechanical arrangement of rods and gears which allows sliding movement in each of the three axial directions. Through the amount of turning of each of three gears, the computer can calculate the distance extended along each axis. The user simply positions the tip of the crane's arm to a point of interest and instructs the computer --through a foot switch in this case-- to note the current position. Although this method is much faster than the completely manual approach, it is still very time-consuming. More serious is the severe limitation to the range of object sizes which can be measured. Being a mechanical device, there are also considerations of its bulkiness and the inertia and slippage of its moving parts.

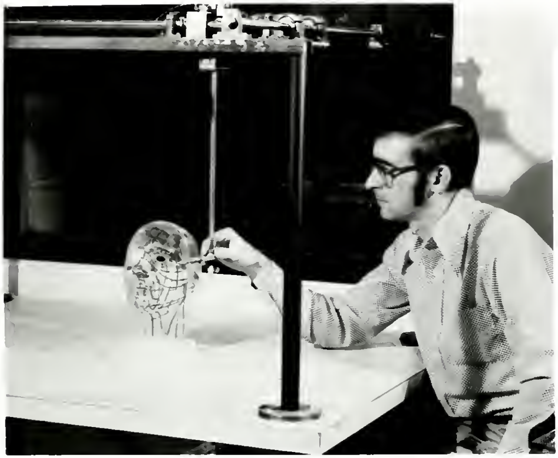


Fig. 1-3: Direct measurement with a 3-D "crane" (from [18])

Fig. 1-4a: Detail of fishing-line digitizing unit (from [20])

Fig. 1-4b: Fishing line-connected 3-D digitizing handle ("wand") resting on tripod (from [20])

A large variety of devices of this same general type have been constructed. One such device has been in use at the University of Utah for a number of years [20]. It uses three separate spring-loaded fishing-reel/fishing-line units (Figure 1-4a). These three assemblies are placed around the top corners of the working volume and the ends of the three fishing lines are all connected to the tip of a pointing device (Figure 1-4b). From the amount of rotation on the shaft of each reel, the length of fishing line rolled out can be calculated. Assuming that the lines are unobstructed, the three-dimensional position of the pointer tip can be calculated from the three separate lengths of the fishing lines.

1.2.3 A Holographic Method

Gara, Majkowski and Stapleton of General Motors Research Laboratories report the development of a novel new digitization technique [8]. Although their system does not have direct applicability to the interaction-oriented scene analysis applications, it may provide a solution to the off-line object-digitization problem.

Their method consists of first taking a controlled, high-quality holograph of the object of interest, then extracting surface measurements from the developed holograph with a special-purpose computer-controlled video viewing system. The surface measurements are calculated by moving the video detection sytem about the object's holographic real image. As seen in Figure 1-5, there is a large angular orientation between the face of the video detector and the object's surface image to allow both in and out of focus parts of the image to hit the video detector's surface. The intersection of the object's surface and the video detector's face is the locus of points on the detector face at which the image is in sharpest focus. Figure 1-6 shows a typical video image from the detector. (To aid in this focus-determination process, an optical

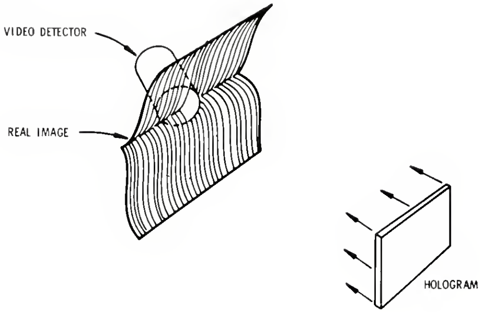


Fig. 1-5: Orientation of video detector to holographic real image

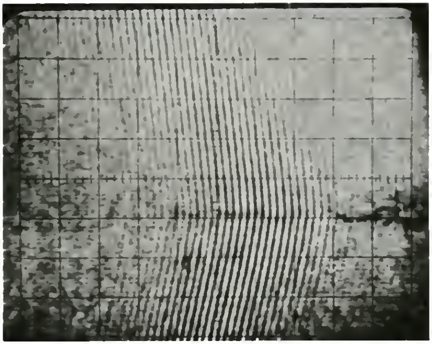


Fig. 1-6: Image from video detector showing in- and out-of-focus areas

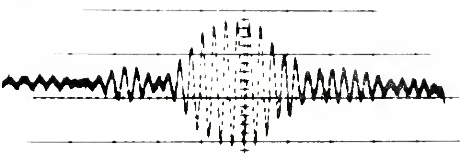


Fig. 1-7: Signal from part of one scan line, showing a region of focus (all three figures from [8])

interference pattern was projected onto the object's surface when the holograph was taken.) Figure 1-7 shows the video signal from one scan line, the point of optimum focus being at the peak of the signal's envelope. After the optimum-focus locations are determined in a single video image, the system incrementally moves the video detector in an attempt to track the object's surface contours. In this way, all visible surfaces of the object can eventually be measured.

1.2.4 A Moire Method

Speight, Miles and Moledina report the application of a Moire method (suggested by H. Takasaki [19]) to the 3-D measurement of slaughtered animal carcasses [17]. Figure 1-8 shows an overhead view of the geometric arrangement of camera, flash-gun light sources, sliding grid and the carcass of interest. The resulting photographic image contains contour lines each of which is of equal depth from the grating plane (Figure 1-9). Although the actual digitization in the reported system was largely a manual operation, there do not seem to be any theoretical obstacles to the automatic processing of these contour maps. The main limitation to applying this method to the more general object-description problem may lie in the method's difficulty in accurately capturing complex, detailed, rapidly-varying surface structures.

1.2.5 Multiple 2-D Images

Acquisition of three-dimensional information from multiple two-dimensional photographic images is not just a widely used digitization method, but is the basis for an entire technical field, stereo-photogrammetry -- most likely inspired by the human stereo vision system.

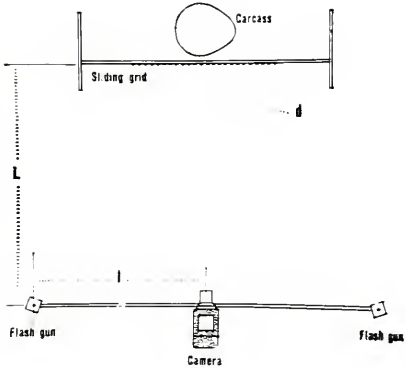


Fig. 1-8: Geometry for generating Moiré patterns (from [17])

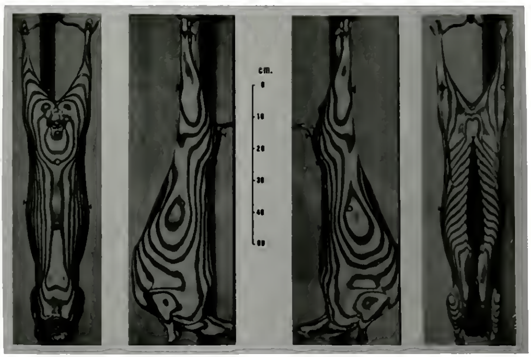


Fig. 1-9: Moiré contour maps of 4 views of a lamb carcass (from [17])

The geometry is beautifully simple. If a picture of a three-dimensional environment is considered to be an image drawn on a window-pane in front of the viewer's eye, then a point on that picture must correspond to a spot in the environment somewhere along the line defined by the viewer's eye and the point on the image.

Given another eye-image pair at a different orientation to the object, and assuming the point of interest is in view in both images, the point's three-dimensional position is simply at the intersection of the two lines of sight (Figure 1-10).

A variety of methods are based on this simple idea. A common technique consists of marking the points of interest on the object itself, then taking pictures from at least two different viewpoints (Figure 1-11). If the camera/eye positions and orientations are not known, they can be calculated from the correspondence between the picture positions and the known 3-D positions of a number (at least 6) of "reference" points in the object's environment [14].

When marking the subject is not practical, other methods can be employed. The common practice is to take two pictures from locations only a small distance from each other -- similar to the two human-eye views. An operator then looks at these images through a suitably adjusted stereo viewer and perceives the three-dimensionality of the object. By moving a pointer in each view until they "merge" in the virtual three-dimensional environment, he performs the correspondence which previously consisted of manually marking the object. From the X,Y distances of the pointers in each image, the three-dimensional position of the perceived point can be calculated [10]

An obvious advantage of this viewing approach is that an indefinitely large number of points can be digitized, since with the marking method, only the actual points marked can be measured. But with a stereo viewer, the accuracy of the measurements depends not

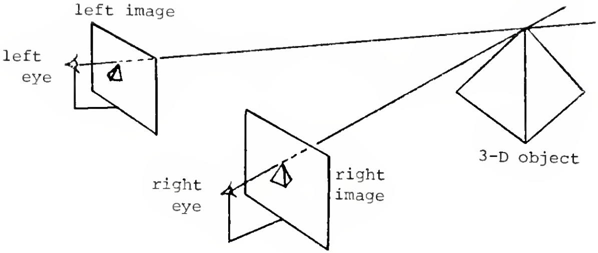


Fig. 1-10: Triangulation from two images

Fig. 1-11: Actual digitization using two images with a data tablet
(from [18])

only on the interocular distance, but also on the visual acuity -- and the patience ! -- of the operator.

Several attempts have been made to eliminate the need for the human operator to specify the corresponce for each point to be measured. Levine et. al. [12] , expand on an earlier algorithm of Julesz[11] which is based on the observation that when two views of a scene are taken from nearby positions, relative to the objects, then the differences between corresponding parts of the two images is largely an X-axis offset, with the amount of offset related inversely to the distance of the object from the viewer (see Figure 1-12).

The technique then, is to digitize the two images and cross-correlate parts of corresponding scan lines. The X offset of the best fit can be used to calculate the three-dimensional position of the point defined by the center of the two matching scan-line segments.

Some initial success has been reported with this method. The obvious difficulty is that the viability of the offset-difference assumption (due to the depth-variation of the object surface) is often inversely related to the distance of the object from the viewing position; the assumption is reasonable for distant or flat regions where the view from both eyes is essentially the same, but it is often invalid for close-by objects, as with the face of a person , for whom one view may contain one side of the nose, and the other view may contain the other side. On the other extreme, if the image of the local surface is too similar in both images -- e.g. flat side of a building, a new sidewalk -- then there will also be difficulties due to a lack of characteristic features with which to achieve a high cross-correlation.

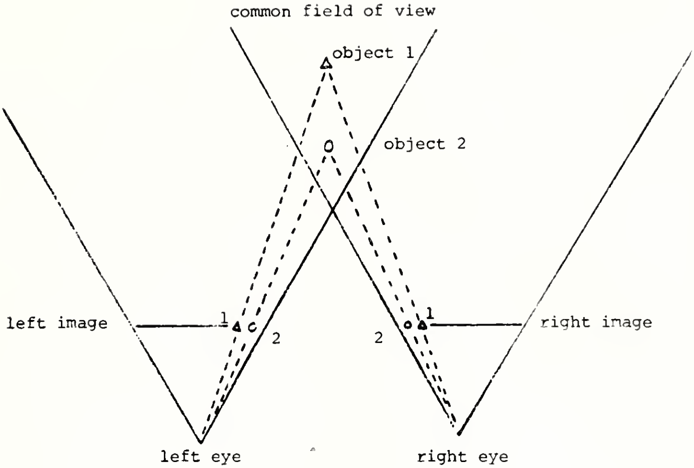
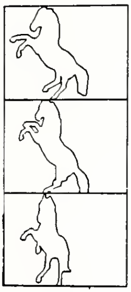
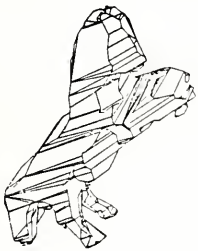


Fig. 1-12: Cross-correlation of stereo images for depth determination



(inputs)



(output)

Fig. 1-13: Object reconstruction from projected silhouettes (from [4])

As part of a more extensive project on computer vision, Baumgart[3,4] demostrates a technique of reconstructing 3-D objects from their image silhouettes. The geometry is similar to the triangulation technique in Figure 1-10, except that in this case instead of line-of-sights being projected, the cone-shaped projections of a silhouette are mapped into the object space. The object by definition is constrained to lie entirely within each and every one of these projections. Baumgart describes it as being like "the old joke about carving a statue by cutting away everything that does not look like the subject." Figure 1-13 shows 3 silhouettes of a plastic horse and a view of the reconstructed object. It is of interest to note that the input views were all from the horse's left side, while the view of the reconstructed object is of the horse's right side. Due to the projective nature of this method, however, surfaces with full concavities cannot be adequately reconstruted.

1.2.6 Controlled Illumination on Objects

Methods for extracting three-dimensional information from multiple two-dimensional projections are not limited to considerations of photographs only. If the geometry of Figure 1-10 is reconsidered, it can be observed that a pencil-beam of light can replace one of the two lines-of-sights used in the triangulation process. In this way, one of the photographs could be eliminated; the beam of light would be seen -- if not obscured by some object -- as a bright reference point in the remaining photograph. This kind of a system yields itself naturally to automatic processing; the origin and orientation of the pencil-beam of light can be placed under computer control and the photograph can be input as a video picture. If the object under investigation can be examined at length, then an arbitrary number of points on its surface can be digitized (Figure 1-14).

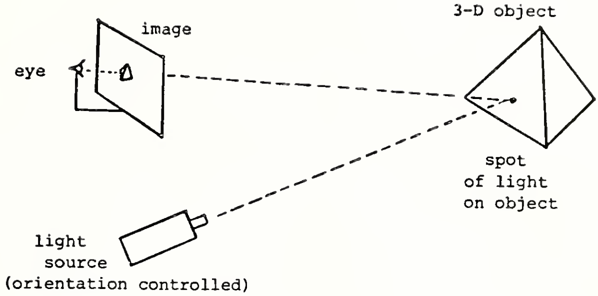


Fig. 1-14: Digitization with one image and a pencil-beam of light

Fortunately, the geometry of such a system is over-solved, and thus can be simplified. Instead of two lines, a plane and a single line are sufficient to uniquely define a surface point (Figure 1-15). A number of investigators have used this kind of a system [1,2,5,13,16]; a plane of light, rather than a pencil-beam, is projected onto the object's surface. In this way, from a single video image, the system can extract not just one surface position, but rather a large number of points along the visible intersections between the plane of light and the object's surface.

The method developed for the present system is in some ways the inverse of the above "plane-of-light and single video image" design. The system is described in Chapter 3.

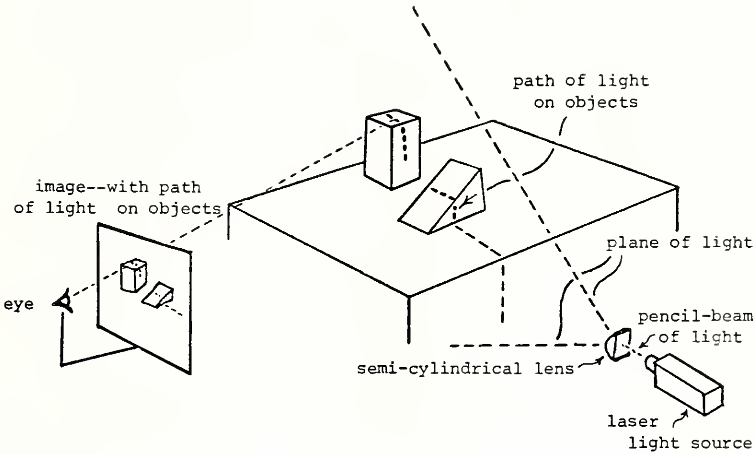


Fig. 1-15: Digitization with one image and a plane-beam of light

CHAPTER 2

DESIGN PHILOSOPHY

It is important in design considerations to review not only what the present limited system may be able to do, but also to delineate the scope of capabilities desired for the eventual "ideal" system.

is the basic goal of this research? It is to design a system which can easily acquire three-dimensional data and use it to construct surface descriptions of arbitrary three-dimensional objects. The general idea is to build a system in which the hardware sensor(s) and the software analysis algorithms interact to produce a more capable system than would be possible without this interaction.

The desired system would have a sensor whose orientation to the object(s) could be altered to allow input data from various views of the object. This could be accomplished in a number of different ways. There could be several sensors mounted at strategic locations around the system's environment. There could be a system-controlled device -- an arm, a turntable -- which could move the object. The sensor itself could be movable -- on a track, on a computer-controlled arm, or mounted on a moving platform. Of course, the particular application would influence the configuration design. For example, the "moving platform" model would be the one most likely to be used for a robotics application.

The scenario would go something like this. The object to be scanned is placed in the system's environment. The sensor starts scanning the environment according to some initial control parameters -- scanning the entire environment at a cursory level of

detail, or perhaps scanning only until a close-by object is encountered. The analysis system -- let's call it Analyzer -- processes this initial scan data and begins to contruct its object descriptions. These not yet being complete, Analyzer calculates the control parameters the sensor needs to obtain the additional input data. The sensor again gathers some data, now according to the new specifications. Analyzer processes the new scan data and integrates it into its developing object-description structures. It again determines whether it needs additional input data. If it does, it again calculates the sensor control parameters. Again, the sensor is instructed to obtain more data, according to its new set of control specifications. This interaction between the sensing device and Analyzer continues until some "completeness" criterion in Analyzer is satisfied.

This approach has several advantages over a simpler method. First, only the data which is needed is actually acquired by the sensor. In this way, neither the sensor nor Analyzer is burdened with unnecessary data. The level of detail can now vary with the specific application. If the task is to give object descriptions to help navigate a robot through an obstacle-filled environment, then one or two requests to the sensor may be sufficient. If, on the other hand, the task is to digitize some arbitrary object for a computer graphics system, then Analyzer could successively direct the sensor to those regions of the object which have yet to be charted with adequate detail. The level of input detail could also be context-dependent. If the objective is to inspect an assemblage for missing bolts, then the level of detail could be modified as Analyzer directs the sensor towards the regions of interest -- after, of course, it has located these regions from earlier inputs.

2.1 Hardware Design Considerations

For an integrated system such as the one just described, the hardware system should be as flexible as possible. Desired was a sensor which could acquire three-dimensional surface data simply, and directly under computer control. An accurate time-of-flight rangefinder would have been ideal, but, alas, too expensive. This kind of system needs very high speed electronics which are capable of sensing the delay between the time a pulse of laser light is started and the time its reflection returns from the objects's surface. This may be as little as a few nano-seconds (billionths of a second !). Time-of-flight rangefinders have been used to measure distances from a few meters to thousands of kilometers.

For the present purposes, the most reasonable of the previous approaches seems to be the "single image and a plane of light" method shown in Figure 1-15. Even with this system, however, the degree of interaction between the data acquisition and the analysis system is limited by the actual ("wall-clock") time and the computer-processing time commitment for an entire video image. Even if only one or two 3-D points are needed, an entire video image has to be input. In order to extract 3-D information, these systems must process the large amount of data inherent to a TV image. Although they can extract many 3-D positions from each TV image, the rigid input pattern required to do this -- for example, having all points be co-planar -- may discourage experimentation with analyses utilizing more context-dependent patterns of inputs, for instance, those analyses for which the density of input data varies with the level of interest in the local surface region. A system with efficient, explicit, single-point measurement capability was judged more suitable forthe present effort.

2.2 Analysis System Considerations

Continuing the flexible approach to system design, it was decided that although Analyzer may specify control parameters to the given input device, it should not depend on interaction only with a particular kind of input device. Thus, for instance, when Analyzer is connected to the proper input device, it can be constructing object descriptions of buildings, automobiles, or microscope specimens. Also, while it can always request more data for its description-building process, it should always be ready to quit -- i.e., the *form* of its object descriptions should be the same after the analysis of one scan input as after ten. (These descriptions structures may, of course, abound with "unknown" and "not sure" markers for much of the object's surface.) It is also unrealistic to suppose that after each request to the input device, Analyzer will get the exact data it requested. Due to obstacles in its way, or because of difficult surface characteristics, the device may not be able to obtain the desired data; so Analyzer should be able to integrate any new data which it gets.

The system should also be able to deal concurrently with more than one object in the scene, and of course, there should be as little restriction as possible about the kinds of objects which are acceptable; restrictions to planar-faced solids or simple geometric shapes would be regrettable. As with any system operating in the "real world", the analysis process should not fail simply because it gets some conflicts about its environment -- e.g. different measurement values for the same surface from different views. In short, a system was sought which would be simple, yet flexible and powerful enough to allow implementation of a variety of experiments.

CHAPTER 3

THE HARDWARE SENSING SYSTEM

The present sensor implementation is a simple, computer-controlled triangulating rangefinder consisting of a mirror-deflected laser beam and spinning-disc detectors. The spinning-disc detectors were previously developed by Robert Burton as part of a PhD dissertation [6]. A basic description of his system will aid in understanding the present implementation.

The objective of Burton's project was the rapid digitization of multiple three-dimensinal points of interest. The tip of a wand, the fingertips of a designer, the key body points of a dancer can all be defined by the physical placement of very small light bulbs -- actually Light-Emitting-Diodes (LED's) -- connected by thin wire to the computer. With the room darkened, the computer, in sequence, turns on each of the lights, at which time several widely-spaced detectors are asked to note the position of the (only!) spot of light. By triangulating the data from several detectors, the three-dimensional position of the small light can be determined. A naive approach would have been to use TV cameras as detectors and find in each of the images the position of the single spot of light. The actual implementation uses much simpler, more efficient detectors than TV cameras. Each detector consists of a rapidly spinning disc set between a wide-angle camera lens and a light-sensitive Photo-Multiplier (PM) tube (see Figure. 3-1). The disc has radial slits cut at regular intervals, which at the proper orientation allow the light passing through the lens to reach the PM tube. A reference PM tube and reference light unit is added to monitor the slits as they pass by. From this information, the exact position of a slit at any instant can be calculated. Now, since

the room is entirely dark except for the illumination of the one small LED of interest, the only instant at which the PM tube senses any light is when the tube, a slit in the disc, the lens and the LED are all lined up. Since the position of the slit at that instant can be calculated from the reference PM tube data, the (unknown) position of the LED must be somewhere on the plane defined by the positions of the slit, the lens, and PM tube (see Figures 3-1,2,3). Since more than one of these sensors around the room is expected to "see" the LED, its position is simply at the intersection of all these detector-defined planes.

To modify this system into a computer-controlled rangefinder, a computer-deflected laser beam was added to replace the LED's . The amount of laser light reflected from most light surfaces was found to be sufficient to be noticed by the PM tubes in the detectors. Now any surface point within the laser's (and the detectors') field of view can be measured simply by aiming the laser beam in that direction. The deflection of the beam is accomplished by reflecting the beam off two small front-surface mirrors which are connected to the shafts of galvanometers mounted perpendicular to each other (Figure 3-4). The control signals to drive the galvanometer electronics come directly from the computer's digital-to-analog converters.

Although there are eight detectors in the present system -- two at each corner of the room -- it is easy to show that only one detector is actually necessary to obtain 3-D surface positions. Since the laser deflection system is under computer control, and the physical position of this unit in the room can be determined, the definition of the laser-beam line can be calculated directly from the current X and Y deflections. Now, since each detector identifies a plane through the room in which the spot of light must lie, the spot has to be at the intersection of this plane and the laser-beam line. (Because the present implementation has the luxury of using eight detectors, knowledge

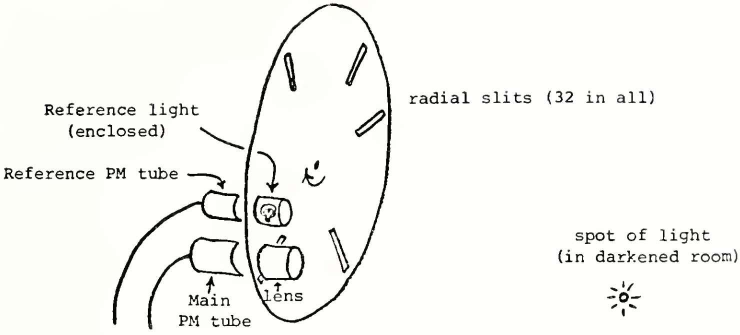


Fig. 3-1: A spinning disc detector and a spot of light

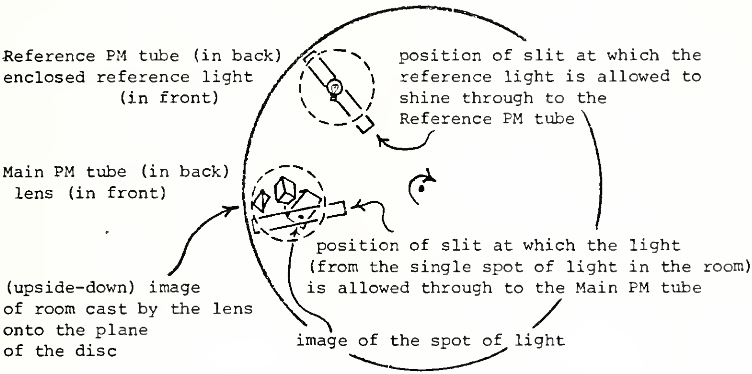


Fig. 3-2: Details of spinning disc detector

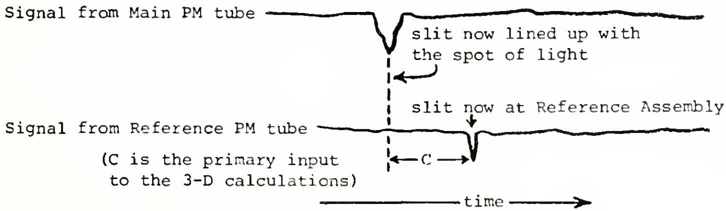


Fig. 3-3: Primary signals from the detector system

of the laser-beam of light in the environment is not used in the 3-D calculations.)

It is important to distinguish this kind of a digitizing system from the similar ones described in Chapter 1. First, in place of the spinning disc detectors, TV cameras could have been used, similar to the "single spot of light and the single image" method shown in Figure 1-14. The primary advantage of the present disc detectors over TV cameras is their speed and simplicity. A disc (with 32 radial slits) spinning at 3500 r.p.m. scans the environment approximately 1900 times each second, as compared with the approximately 30 frames each second acquired by the standard TV camera. Also, as can be seen from Figure 3-3, each scan directly produces a single number ("C") for processing, bypassing the need to handle the approximately 250,000 points in each video frame.

Comparisons to the slightly different method of the "single image and a plane of light" (Figure 1-15) are somewhat more involved. The digitization rate of this method is substantially improved by being able to process many points (all along the plane of light) from a single video image. The orientation of the plane of light, however, can only be changed between video frames, perhaps each 1/30 of a second; so the system is "committed" to an orientation for a large number of points. With the present random-access laser-beam system, this commitment is only for a single point; so the orientation of the beam can be changed "on the fly." This feature is especially important for those applications in which the digitization of the scene is context-dependent, that is, one in which each deflection position of the laser beam may be a function of the previously calculated 3-D positions.

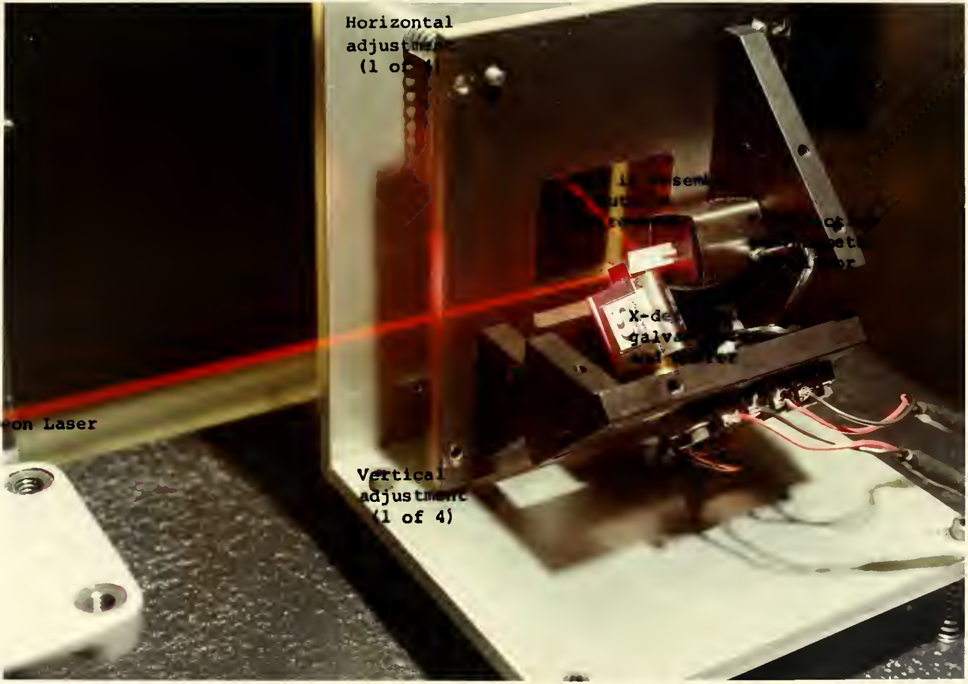


Fig. 3-4: The laser deflection system

Horizontal
adjustment
(1 of 4)

Hole in assembly
to outside
environment

Y-deflection
galvanometer
and mirror

X-deflection
galvanometer
and mirror

Helium-Neon Laser

Vertical
adjustment
(1 of 4)

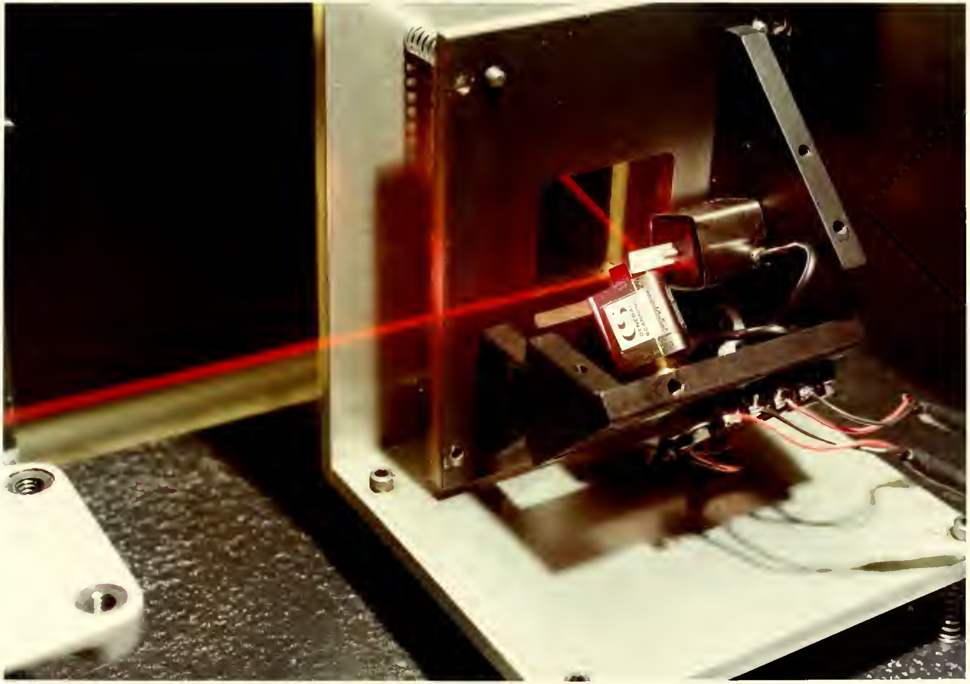


Fig. 3-4: The laser deflection system

CHAPTER 4

DATA ACQUISITION, ANALYSIS AND OBJECT RECONSTRUCTION

The implemented programs consist of an interlocking set of modules, from those which directly control the sensing system to those which analyze the data and generate the actual object descriptions.

4.1 Basic Data Acquisition Programs

Since the laser deflection system is completely under computer control, a set of programs specify control parameters to the hardware system. These parameters are calculated from higher-level specifications obtained from the human operator. The operator can specify the subregion in the laser deflection system's field of view which is to be scanned. He can also specify the number of positions in the horizontal and vertical directions for which measurements should be taken.

Due to the primitive nature of the present scanning system, a number of strategies have been implemented in hopes of improving the accuracy of the input measurements. A multiple number of measurements are usually taken at each deflection position of the laser. The resulting measurements are then used to calculate a single more reliable value. In addition, if the laser spot is not "seen" by enough of the sensors for a minimum number of these attempts, then no final value is recorded for that particular laser position. Another filter traps any value which falls outside of the specific "working volume" of the scene. (A more detailed explanation, along with actual Teletype listings of the programs in execution, is provided in Appendix C.)

4.2 Analysis and Object Reconstruction Methodology

While a number of different uses can be found for this unusual sensing system, it was decided that the task of object reconstruction would be an appropriate first application. The term object reconstruction is used here to mean the generation of complete, closed surface topologies, defined by a connected set of polygonal tiles, which approximates the surface data acquired from one or more views of the objects in the environment. Although this task has many similarities to the much-researched scene analysis problem in artificial intelligence, the present implementation has a somewhat different orientation in that it makes almost no assumptions about the specific kinds of objects it expects to find. This feature can be viewed as an advantage in favor of generality, but of course it also prevents the system from making many inferences from partial data -- concerning, for example, the likely location and characteristics of obscured surfaces.

Early in the project a similarity was noticed between this object reconstruction task and the task of visible surface algorithms in computer graphics. Basically, the task of a visible surface algorithm is to construct a particular view of a scene from a description of the objects in the scene and the specifications of the particular view. The task of the present effort is, in a way the reverse of this, to construct object descriptions from one or more views of the objects. Central to both these tasks is the effective manipulation of three-dimensional data. In a recent analysis of visible surface algorithms [18], Sutherland, Sproull and Schumacker make a number of observations which may also be applicable to the present effort . They note two important elements common to most visible surface algorithms: sorting and coherence.

4.2.1 Sorting

To bring order to the three-dimensional data with which all these algorithms must deal, they must all sort the data in an effective manner.The order of the dimensions of the sort and the type of sort used strongly influences the flavor of the final algorithm. Some sort along the Z axis first, then along the Y, then X; others sort Y first, then Z then X. The authors found implementations of almost all the combinations and even speculated on the characteristics of the algorithms which would evolve from the combinations not yet investigated.

4.2.2 Coherence

The idea of coherence was judged to be significant in reducing the enormous amount of processing involved in tasks of this kind. The basic notion is that there is a significant amount of coherence -- similarity, connectedness -- between adjacent parts of most pictures. In a scan-line oriented algorithm, for instance, every scan line does not have to be independently generated. Rather, it can simply be thought of as a modified version of the previous scan line. Making this modification is almost always cheaper than generating the line "from scratch."

4.2.3 Applicability to the Present Situation

Both these observations seem applicable, in a somewhat altered fashion to be sure, to the object reconstruction problem. Certainly some reasonable sorting mechanism must be developed to control the otherwise unwieldy interaction between all the elements of the input data. If all the input data was sorted along one of the three axial components, say the Z-axis, then the elements around one particular region (Z = 30

inches, for example) could easily be extracted and analyzed. This analysis process could presumably answer the quesion, "what do the objects 'look like' at this level" -- i.e. along the Z=30 plane. The process' answer could be a set of closed contours which would hopefully approximate the cross-sections of each object at this (Z=30) level.

The coherence idea suggests that this description may not greatly differ from the results of the analysis executed at a nearby plane -- at Z=29-inches, for example. If it is found that the objects have indeed not changed significantly, then there would be no further need for analysis anywhere between these two planes; the object descriptions throughout this region would be an interpolation between the already-detemined adjoining descriptions.

Basically then, the algorithm reconstructs the scene at a sequence of these parallel planes. The cross-sectional contours found in adjacent planes are connected and a surface of triangular polygons is defined between each pair of connected countours. The final description for each object is a collection of connected contours and a polygonal surface defined over them. (In the following section this entire process in described in greater detail.)

This basic approach was chosen because it very neatly reduces the dimensionality -- initially from three to two dimensions, and as will be seen later, eventually from two dimensions to one. Also, since no assumptions are made about the input data coming from a single view of the scene, the actual data can as easily come from one as from ten different views. Neither are assumptions made about the distances between the adjacent "cutting" planes on which the actual analysis takes place; so the distances can be varied, being made larger in those regions in which the analysis proces reveals little

change in the scene, being made smaller when the analyses from initially-adjacent planes are sufficient dissimilar. Also, as may become more apparent later, this approach allows incremental, modular improvements at virtually any place along the entire data-acquisition - analysis - reconstruction process.

4.3 Description of Analysis and Reconstruction Algorithm

Initially the basic surface points measured by the sensing system are converted into a surface representation. It is here that the single assumption about the scene is made. It is assumed that between adjacent scan-point positions the surface of the objects in the scene can be approximated linearly. (Of course, in an improved implementation, the distance between individual scan-point positions could be varied dynamically, perhaps according to the variation in the local surface contour, making the above assumption even less risky.) This single assumption allows the definition of a grid of small triangular tiles over the entire scanned region -- with each small triangle in the grid being defined by two consecutive laser positions on a laser scan line and a single laser position on one of the two adjacent scan lines. [Compare Figure 4-3 with Figures 4-1 and 4-2.] Of course since some of the points may be missing or may have been discarded as "unreliable", there may be holes in this surface structure.

These small triangular tiles are used as the data for all further processing. They are treated individually, so that tiles in the subsequent analysis and reconstruction programs can come from different scans, made most likely from different orientations to the scene.

Next, the entire group of these tiles -- whatever their origin -- are ordered according to their highest Z value (vertical distance from the floor). The series of

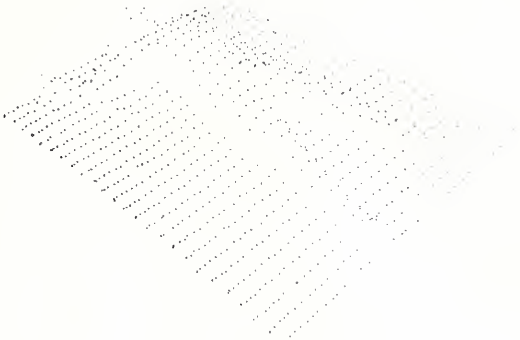


Fig. 4-1: Original 3-D data points

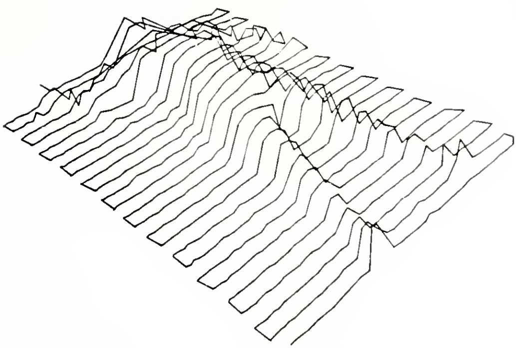


Fig. 4-2: Path of laser

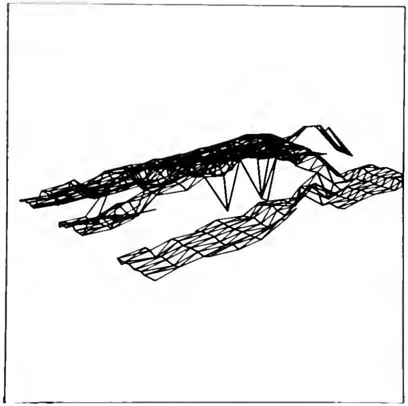


Fig. 4-3: Polygonal surface defined over data points

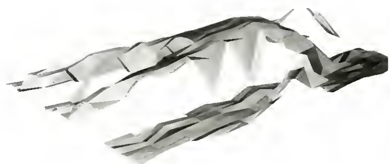


Fig. 4-4: Shaded version of polygonal surface

"cutting planes" parallel to the ground is now determined. These can be at arbitrary positions along the Z axis. (Actually there are already some provisions for varying the correspondence between the axes named in the input data and their assignment in the analysis system, e.g. the X Y Z input sequence can be treated as -X Z Y -- see Figure 4-5.)

4.3.1 Analysis on Each Cutting Plane

The cross-sectional processing on each "cutting plane" is at the heart of the analysis system. Since the input surface tiles have been sorted by Z, determining which tiles intersect the plane is straightforward. The intersections between this plane and these appropriate tiles are now calculated. Reconstructing the object cross-sections ("sectionals") at this cutting plane is a matter of organizing the just-determined line-segments into a number of simple closed regions (Figure 4-6).

The major difficulty is that these line-segments don't usually connect directly into simple closed regions. There are invariably gaps and usually some conflicts. Conflicts occur when two or more line-segments intersect or lie so close to each other that they are thought to represent the same surface, just disagreeing about its exact location. (The accuracy of the original system was claimed by Burton to be around .7 cm. At present, the system -- at least when used with the laser deflection unit -- does not achieve the same level of accuracy.)

To facilitate the construction of these closed regions, sorting is again employed -- all the line-segments are ordered with respect to their larger Y coordinate. Closed regions are built up by sequentially processing the elements of this list, asking at each one, "does this line-segment connect to an already-begun regional contour ?" If it does,

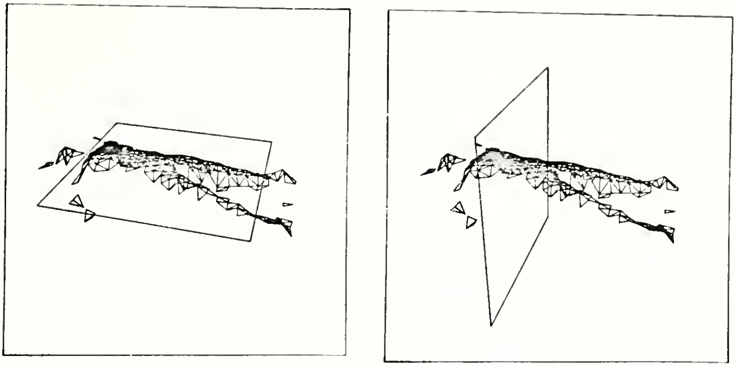


Fig. 4-5: Alternate orientations of cutting-analysis plane

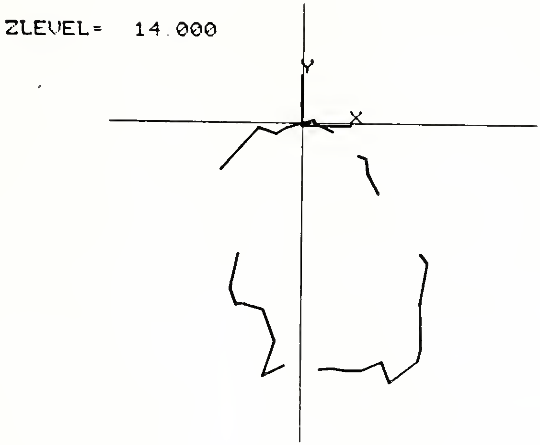


Fig. 4-6: The line-segment intersections between cutting plane and data polygons (segments not necessarily connected)

it is connected to that contour; otherwise a new contour is begun, with this line-segment as its only member (Figure 4-7).

It is at this point in the analysis that the processing is effectively reduced from two to one dimension; to determine the disposition of the current line-segment, the situation is analyzed only along the horizontal line touching the top edge of this line-segment.

It is also at this stage that conflicts in the input data due to overlapping line-segments are resolved. When such a conflict occurs, a pair of new segments is generated such that the contour's boundary is defined along the "inside" of the intersecting segments. [The "inside" of a line-segment is the side on which the object is presumed to lie. This information is derived from the associated input polygonal tile whose original definition -- from the order of its vertices in the scanning pattern -- enables distinction between the side of the tile toward the laser and the side toward the inside of the object on whose surface the tile lies.]

Of course, it is possible for a line-segment to connect two already-established contours, in which case the two are merged into a single, longer contour.

Since the list of input segments is ordered, each segment need be considered only once in this region-constructing process. After all the line-segments are processed, a series of (possibly, but most likely not closed) contours will have been formed.

These contours are combined into a set of closed contours ("sectionals") by adding some artificially-created line-segments. These added segments are marked "blank-unknown" so that a later process can guide the sensing mechanism specifically to these regions in order to make direct measurements of these uncharted areas.

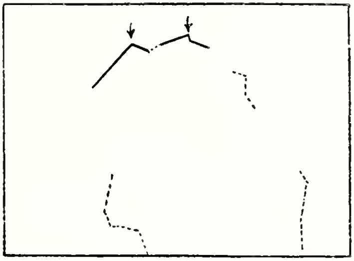


Fig. 4-7: Sectional building -- two started contours
(unprocessed line-segments are indicated by dotted lines)

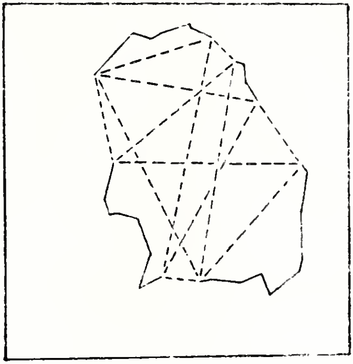


Fig. 4-8: Possible choices for sectional completion

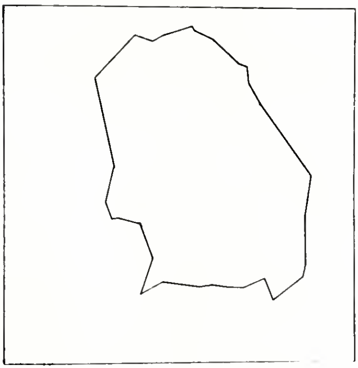


Fig. 4-9: Optimal completion path (minimum total length) chosen

Since there are many possible connection configurations, the one is chosed which contains the minimum total length of these added "blank-unknown" line-segments (see Figures 4-8 and 4-9).

4.3.2 Inter-Plane Reconstruction

After sectionals have been constructed on all cutting-analysis planes (Figure 4-10), the object-reconstruction program defines a "skin" of new polygons between sectionals on adjacent levels which are determined to be connected. The connectivity criterion used here is based on the overlap of sectional bodies in the XY plane -- i.e., whether or not there would be an overlap if the Z values were ignored. In general the connection pattern between sectionals can be more complex than just 1-to-1 (see Figure 4-11).

When a sectional does not connect to any sectional on an adjacent level, it is assumed that this part of the object being reconstructed has terminated here. A "cap" polygon is then generated which covers the entire sectional.

After this sequence of determining cutting planes and sectionals and "skins" is finished for all levels from the top of the scene to the floor, all the objects in the scene have been reconstructed -- as far as possible, that is, with the given input data.

It is important to distinguish these newly-defined polygons from the original polygonal tiles (as in Figure 4-3 and Figure 4-4). The original polygonal tiles coming from the basic laser scans are unordered, may come from several different scans, usually do not completely cover the surface of any one object (i.e., have "holes") and may have conflicts among themselves as to the exact location of some surface. These new polygons which are mapped over the closed cross-sectional contours completely define each object, in the sense that there are no gaps or "holes" in the surface and all

the conflicts have been resolved in the surface structure; so, for instance, the bottom of an object can be "filled in" (as in Figures 4-12, 4-13 and 4-14e) even though no original scan data tiles were actually defined there.

The various colors in the reconstructed objects (Figures 4-13 and 4-14e) indicate different kinds of surface regions: 1) blue indicates surfaces which were derived directly from scan data, 2) green indicates parts interpolated between known points, using the completing-sectionals criterion; 3) red indicates conflicting areas -- those for which the input data conflicted about the exact location of the surface. In a more sophisticated implementation, the location of these regions could help a higher-level controller determine the most advantageous orientation of the sensing system for subsequent scan attempts.

Figures 4-14a through 4-14e illustrate the sequence of steps involved in the data analysis and object-reconstruction processes. [The actual teletype listings of the programs' executions, with commentary, are in Appendix C.]

Figures 4-15 and 4-16 demonstrate the situation in which the sectionals of the same object at adjacent analysis levels are completely disjoint in the X-Y plane. In such a case the connection between these sectionals is not discovered; this has happened in Figure 4-16 with both arms and one of the legs. Presumably subsequent scan attemps of the same object would gather enough additional data about these troublesome regions to enable the appropriate connections to be made.

Figures 4-17 and 4-18 show, in two different forms, the unprocessed scan data from scenes with a small vacuum cleaner and a chair and a small box.

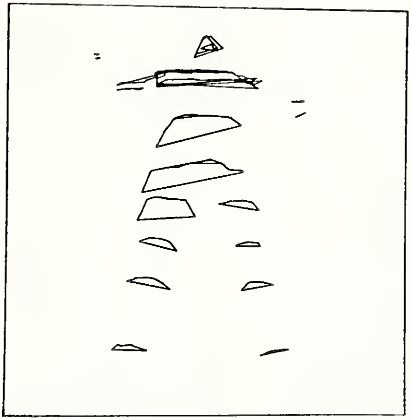


Fig. 4-10: Closed contours of object at various analysis levels

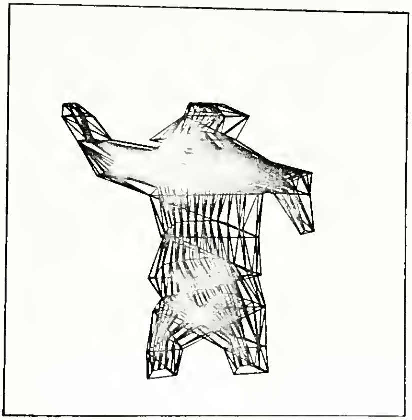


Fig. 4-11: Polygonal skin defined over the contours
(different object than Fig. 4-10)

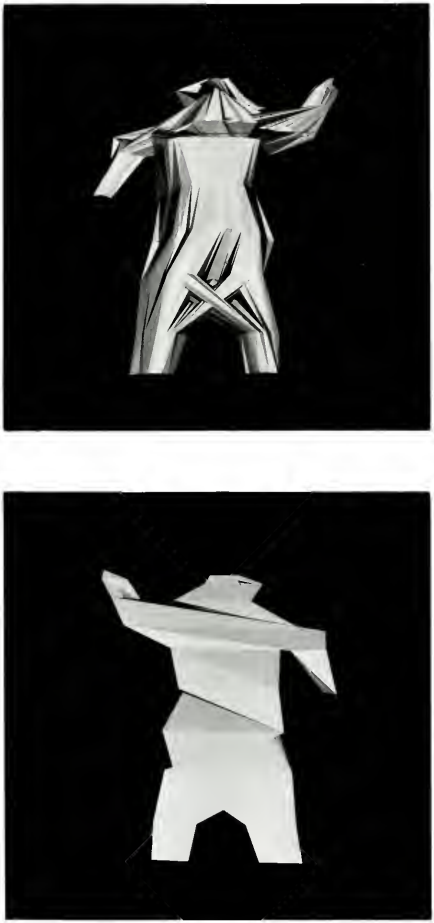


Fig. 4-12: Front and back views of reconstructed object (torso)

Fig. 4-13: Color-coded front and back views of torso
(Blue regions were extracted from the original laser-scan data.
Green regions were added by the analysis-reconstruction process.)

Fig. 4-14a: Perspective views of two separate laser scans of a simple cube (simulated data)

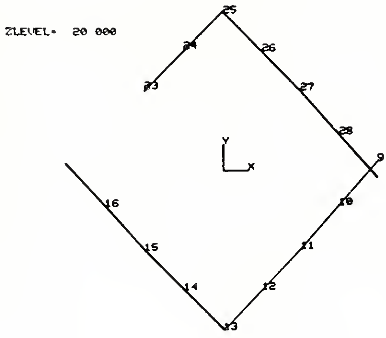


Fig. 4-14b: Line-segments at a Z-level (from the above scan data) illustrating instances of gap and overlap

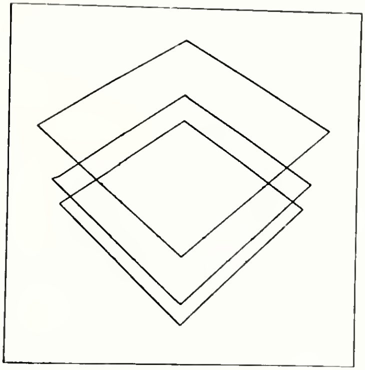


Fig. 4-14c: Completed cross-sections of cube, with gaps and overlaps resolved

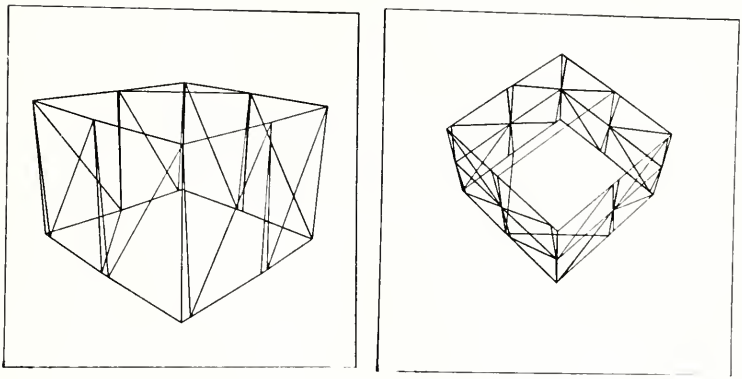


Fig. 4-14d: Perspective views of simulated cube, reconstructed at 2 and 3 levels

Fig. 4-14e: Color-coded views of reconstructed cube,
with and without top
(blue = original data; green = added regions;
red = conflict-resolved regions)

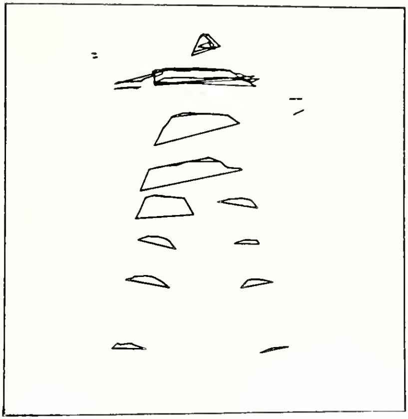


Fig. 4-15: Completed sectionals at 12 levels

Fig. 4-16: Incomplete reconstruction from the above sectionals

Fig. 4-17: Scanned surface of a vacuum and hose

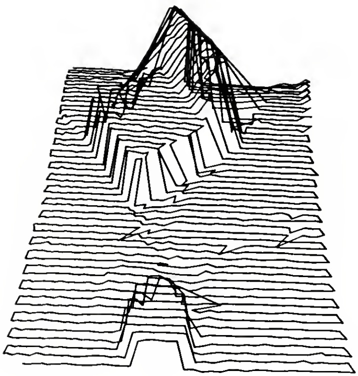


Fig. 4-18: Laser path of a scan of a chair and a box

CHAPTER 5

CONCLUSIONS AND POSSIBLE FUTURE DEVELOPMENTS

5.1 Conclusions

A combined hardware and software system has been described and demonstrated. It can measure surface contours of arbitrary three-dimensional objects and construct completed closed contour descriptions of all the objects seen from one or more views of the scene.

While many different methods for acquiring three-dimensional data have been developed, the present approach was chosen as one which would allow the most easily-controlled and most flexible interaction with the host computer. The simple, direct mode of sensor operation -- the digitization of a single arbitrary point in the system's field of view -- seems uniquely well-suited to the general point-by-point, context-dependent operation of the many pictorial pattern recognition techniques which seem extendable to the analysis of this kind of three-dimensional data.

Also, if the laser deflection unit and the spinning-disc detectors were mobile, the system could work in a wide range of environments; in order to acquire a description of a VW automobile for a computer animation system, for instance, one would no longer need to manually measure a model VW or the actual automobile; it would be reasonable to expect that one could simply take the system out to the parking lot (perhaps only at night,however) and have the system itself generate the description. If the system were mounted on a computer-controlled cart, it might even be able to move around the object,

digitizing only those parts which the analysis algorithm indicated still needed more data.

As a sensor for a robot, the system may enable more efficient processing of visual scenes. As previously mentioned, the robot's vision processing could become more context-dependent; if it just wanted to move through an area, it may only need to measure and analyze a relatively few points in its direct path. Only when it encountered an obstacle would it need to digitize the local region more densely, with the pattern of the points being digitized perhaps being guided by an object recognition system.

The uses for this kind of system are not limited to traditional computer science areas. In the field of medicine, the accurate measurement and analysis of the complex, irregular contours of the human body has the potential for making available entire new areas of observation to aid the physician in the diagnosis of human ailments. Changes in the shape, size and volume of various parts -- arms, breasts, legs -- may be too small to be noticed by the unaided eye, but may signal the start of significant physiological activity. Slowly-developing deformations in growing children may not be noticed until the disorder has progressed beyond the reach of certain therapies.

Essential to the success of this kind of application is not only a suitably accurate and practical input device, but also an effective analysis system which can transform the raw surface measurements into a form meaningful to the physician. The present software system may be applicable almost without modification to some of these tasks. In the human measurement studies discussed in [10], some of the forms of graphical output bear striking resemblance to the cross-sectional contours determined by the present analysis and object-reconstruction system.

The object descriptions produced by this kind of system can also be used to generate program specifications for numerically-controlled milling machines. A copy of an object can thus easily be produced without touching the original. This may be useful either when the original object is too delicate to disturb, or when it is finally to be made from a material which is unsuited for the design process. With this kind of system the designer can construct the object from the material of his choice -- clay, soft wood, plastic -- and still have the final object in the required medium, perhaps aluminum or steel.

5.2 Further Development

Before most of these applications can become a reality, many improvements need to made, both in the actual hardware sensing system and the analysis and reconstruction software.

5.2.1 Harware Improvements

The weakest parts of the present sensing system are the spinning-disc detectors mounted on the corners of the room. As mentioned before, these were developed as part of an earlier project [6]. While they are an interesting first attempt, they are next to inadequate by current standards. For starters, a room ringed by four 22-inch slotted metal discs, spinning at 3600 r.p.m., is not exactly an ideal working environment. The noise alone prevents all but emergency conversation. More seriously, the heat generated by the necessarily large electric motors is such that the system must be shut down for cooling after each 30 minutes of use. Moreover, the accuracy of the system is largely determined by parameters which are difficult to calibrate: the constantly-changing spinning rate, the slightly different slit positions, the different pulse

widths from the light-sensitive Photo-Multiplier tubes.

Another graduate student, Larry Evans, is presently developing alternative digitizing methods. A number of different approaches are being explored, all based on the idea of transforming the signals from several two-dimensional images of the scene into one-dimensional measurements. A key feature of the various methods, the complete absence of moving parts, is most encouraging. Interested readers are referred to Evans' research proposal and his forthcoming dissertation [7].

The other major part of the hardware system, the laser deflection mechanism, also has several serious limitations. Being a moving mechanical device, it encounters inevitable overshoot problems when attempting to move quickly from one position to another. The present electronics attempt to minimize this problem by gradually, rather than instantly, changing the galvanometer signals from an old to a new value. This solution, however, is a very rough one at best. A significant improvement would be a system whose galvanometers could provide continuous positional feedback, from which the electronics could determine a much more accurate control signal, enabling the system to respond much faster, and presumably with more accuracy. With the increased speed, more sophisticated sensing strategies -- such a real-time contour tracking -- could become practical.

Perhaps the major inherent problem with the present hardware model is the one of obstruction. Since it is a triangulating rangefinder, it needs an unobstructed line-of-sight from both the laser origin and at least one of the detectors in order to digitize a surface position. The more extreme the concavities on the object's surface, the more often this obstruction problem prevents the digitization of a particular position. One possible solution is to have a different kind of ranging system. One which seems a

reasonable candidate is a modulated laser time-of-flight rangefinder. While there is at least one such product on the market whose specifications are outstanding, with accuracy approaching one part in 10**6, its price is unfortunately prohibitively expensive [9].

Simpler, less expensive systems can, however, be constructed, but they are presently limited to an accuracy of about one inch [15]. While this is not accurate enough for most digitizing purposes, it may still be appropriate for certain other applications like robotics. These would certainly overcome many of the limitations of triangulating rangefinders. There would no longer be a need for at least two separate positions from which to triangulate. Thus the operating environment could become less restrictive. It is reasonable to expect that the range of useful distances could also be enlarged -- for instance, real autos could perhaps be analyzed instead of just toy models. All these advantages may outway the accuracy limitations for certain applications.

Of course, for some applications it may be advantages to have either the object or the sensing device on a movable platform to allow controlled changes of orientation between the sensing device and the object(s) under consideration.

5.2.2 Improved Analysis and Reconstruction Methods

The range of possible improvements in the software is even larger than in the hardware sensing system. Due to the modular nature of the software implementation, improvements in virtually any area can be made without modifying any other section.

The basic digitization could be made significantly more accurate. In the present Implementation, the basic three-dimensional coordinate determination from the sensor

input values does not take full advantage of the fact that the line of the laser light in the room is known. If the deflection system were to be calibrated with some precision, then the angular deflection parameters could be used in these calculations. A formulation of the problem optimized to the geometry of the system may also be helpful; for instance, a polar coordinate representation with origin at the laser deflection point may yield a simpler set of equations to be solved.

Since the laser deflection is directly under computer control, an obvious improvement would be to implement a dynamically-changing scan of the environment. This could be as simple as varying the distance between adjacent sample positions based on the local surface contour, or it could be as involved as placing the entire reconstruction and recognition process directly in control of the scan. In this way, the laser beam could only be moved to those parts of the scene which were of significant interest to the system.

Perhaps the most far-reaching modification would be to infuse the analysis and reconstruction process with some 'a priori' knowledge about the objects it is likely to encounter. Such knowledge could aid not only in guiding the scanning process, but also in helping to resolve difficulties in reconstructing parts of objects about which there is incomplete information.

APPENDIX A

MAJOR SOFTWARE MODULES

NWRSL--(NeW Real-time digitization with Scanning Laser) -- or RSL -- controls the laser deflection system, acquires raw points from the Burton Box hardware and software, generates a ---.DAT file of all acceptable 3-D measurements in a laser scan.

NPTDIS -- (New PoinT DISplay) -- displays (onto a Tektronix 4012 storage scope) the raw 3-D points from a (---.DAT) file from NWRSL . It can also filter out points which are grossly out of place, the filtering being based on a minimum distance criterion from adjacent sample points.

NGENPO--(New GENerate POlygonal surface) -- reads in a points file (---.DAT) from NWRSL and generates a surface of triangular polygons over these points. These polygon definitions are output onto a ---.POL file. For diagnostic purposes, a number of other data files can also be generated. The most frequently used ones are

---.PTB, a two-dimensional grid of characters showing whether or not the data point at each position in the scan has been succesfully digitized,

---.MDR, the original data points and the just-defined polygonal surface, in the standard (MOTION-DATARD) graphics system format. This enables easy display of the original points and polygonal covering.

SCANA -- (SCan ANAlyzer) -- accepts one or more pairs of point (---.DAT) and polygon (---.POL) files from NWRSL and NGENPO; and generates a file of line-segments (---.SGM). These line-segments are the intersections of the polygons with chosen cutting/analysis planes.

MAKSEC -- (MAKe Sectionals) -- generates a (---.SEC) file of sectionals (closed cross-sectional regions) from a line-segment file from SCANA.

OBREC -- (OBject REConstruction) -- generates complete object descriptions from a cross-sectional file (---.SEC) of MAKSEC. These descriptions are in the format of the current general-purpose graphics software at U. of Utah (MOTION-DATARD).

APPENDIX B

DATA FILE FORMATS

Original Point File Format

<---.DAT file> ::= <scan descriptor> <a data point>* [* = one or more instances]

<scan descriptor> ::= <number of positions on a scan line> <number of scan lines> <scan i.d. number>

<a data point> ::= <point i.d. number> <X-value> <Y-value> <Z-value>

A Sample ---.DAT File

```
5 3 1
1001001 +.1   0.01 +20
1001002 -10   0.02 +10
1001003 -20   0.03 0
1001004 -10   0.02 -10
1001005 +.1   0.01 -20
1002005 +.1  10.01 -20
1002004 -10  10.02 -10
1002003 -20  10.03 0
1002002 -10  10.02 +10
1002001 +.1  10.01 +20
1003001 +.1  20.01 +20
1003002 -10  20.02 +10
1003003 -20  20.03 0
1003004 -10  20.02 -10
1003005 +.1  20.01 -20
```

Polygon File Format

<---.POL file> ::= <one text line> <a polygon definition>*

<a polygon definition> ::= <point i.d. number>*** <carriage-return and line-feed>
[*** = 3 or more instances]

A sample ---.POL file

```
NXSTEPS=5 NYSTEPS=3 IDSCAN=1000000
1001001 1002001 1001002
1001002 1002001 1002002
1001002 1002002 1001003
1001003 1002002 1002003
1001003 1002003 1001004
1001004 1002003 1002004
1001004 1002004 1001005
1001005 1002004 1002005
1002001 1003001 1002002
1002002 1003001 1003002
1002002 1003002 1002003
1002003 1003002 1003003
1002003 1003003 1002004
1002004 1003003 1003004
1002004 1003004 1002005
1002005 1003004 1003005
```

Point-Table Format

<---.PTB file> ::= <single text line>
<one scan line descriptor>*

<one scan line descriptor> ::= <scan line #> = <one scan position descriptor>*

<one scan position descriptor> ::= <position has good data>
::= <position does NOT have good data>

<position has good data> ::= X
::= -

<position does NOT have good data> ::= ,

A Sample ---.PTB File

```
NXSTEPS=40 NYSTEPS=30 IDSCAN=2000000
30=,,,,,,,,,, ,,,,,,,,,X X,XXX,X,,, XXXX,,,,,,
29=,,,,,,,,,, ,,,,X,,,,X XX,,,,XX,, XXXXXX,,,,
28=,,,,,,,,X, ,,,,X,,X,X X,,XX,,,XX XXXX,,X,X,
27=,,,,,,,,,, ,,,,,,,,,X XXX,,,,,,, ,XXX,XX,,X
26=,,,,,,,,,, ,,,,,,,,,X XXXX,,,,,, ,XX,,X,XXX
25=,,,,,,,,,, X,,,,,,,,, XXXX,,,,,, ,XXX,,,,,X
24=,,,,,,,,,, ,,,,,,,,,X XXXX,,,,,X XXX,,,,,,,
23=,,,,,,,,,, ,,,,,,,,,, XXXXXX,,,, XXX,,,,,,,
22=,,,,,,,,,, ,,,,,,,,,, XXXXXX,,,X XX,X,,,,,,
21=,,,,,,,,,, ,,,,,,,,,X XXXXX,X,,X XXXX,X,,,,
20=,,,,,,,,,, ,,,,,,,,,X XXXXXX,,XX --,,,,,,,,
19=,,,,,,,,,, ,,,,,,,,,, ---------- ---,,,,,,,
18=,,,,,,,,,, ,,,,,,,,,- ------,--- --,,,,,,,,
17=,,,,,,,,,, ,,,,,,,,,, ---------- --,,,,,,,,
16=,,,,,,,,,, ,,,,,,,,,, ---------- -,,,,,,,,,
15=,,,,,,,,,, ,,,,,,,,,- ,--------- -,,,,,,,,,
14=,,,,,,,,,, ,,,,,,,,,, ---------- ,,,,,,,,,,
13=,,,,,,,,,, ,,,,,,,,,, ,--------- -,,,,,,,--
12=,,,,,,,,,, ,,,,,,,,,, XXXXXXXXXX ,,,,,,,XX,
11=,,,,,,,,,, ,,,,,X,,,, XXXXXXXXXX X,,,,,,XX,
10=,,,,,,,,,, ,,,,,,,,,, XXXXXXXXXX XX,,XXX,,,
9=,,,,,,,,,, ,,,,,XXXXX XXXXXXXXXX XX,XXXX,,,
8=,,,,,,,,X, ,,,XXXXXXX XXXXXXXXXX XXXX,,,,,,
7=,,,,,,,,,, ,,XXX,,,X, ,,,,,,X,,, ,,,,,,,,,,
6=,,,,,,,,,, XX,,,,,,,, XX,,XXXX,, ,,,,,,,,,,
5=,,,,,,,,,X XX,,,,,,,, ,,,,XXXX,, ,,,,,,,,,,
4=,,,,,,,,X, X,,,,,,,,, ,,X,XXXX,, ,,,,,,,,,,
3=,,,,,,,,,, ,,,,,,,,,, ,,,XX,,,,, ,,,,,,,,,,
2=,,,,,,,,,, ,,,,,,,,,, ,,XXXX,,,, ,,,,,,,,,,
1=,,,,,,,,,, ,,,,,,,,,, ,,,,,,,,,, ,,,,,,,,,,
```

Segments File Format

<---.SGM file > ::= <one level's segments>* END

<one level's segments> ::= LEVEL <Z-value> <a segment description>*

<a segment description> ::= <X1 value> <Y1 value> <X2 value> <Y2 value>
<orientation> <a single polygon description>

A Sample ---.SGM File

```
LEVEL   20.000
    9.980 -10.020    .000 -20.000        1 1002003 1003002 1003003
     .000 -20.000   -.030 -19.970        1 1002003 1003003 1002004
    -.030 -19.970 -10.000 -10.000        1 1002004 1003003 1003004
   -9.980  10.020    .000  20.000        1 2002003 2003002 2003003
     .000  20.000    .030  19.970        1 2002003 2003003 2002004
     .030  19.970  10.000  10.000        1 2002004 2003003 2003004
   19.990    .090  10.000 -10.000        1 1002002 1003001 1003002
   10.000 -10.000   9.980 -10.020        1 1002002 1003002 1002003
  -10.000 -10.000 -10.020  -9.980        1 1002004 1003004 1002005
  -10.020  -9.980 -20.000    .100        1 1002005 1003004 1003005
  -19.990   -.489 -10.000  10.000        1 2002002 2003001 2003002
  -10.000  10.000  -9.980  10.020        1 2002002 2003002 2002003
   10.000  10.000  10.020   9.979        1 2002004 2003004 2002005
   10.020   9.979  20.000   -.500        1 2002005 2003004 2003005
   20.000    .100  19.990    .090        1 1002001 1003001 1002002
  -20.000   -.500 -19.990   -.489        1 2002001 2003001 2002002
LEVEL     9.000
    8.979 -11.021    .000 -20.000        1 1001003 1002002 1002003
     .000 -20.000  -1.029 -18.971        1 1001003 1002003 1001004
   -1.029 -18.971 -10.000 -10.000        1 1001004 1002003 1002004
   -8.979  11.021    .000  20.000        1 2001003 2002002 2002003
     .000  20.000   1.029  18.971        1 2001003 2002003 2001004
    1.029  18.971  10.000  10.000        1 2001004 2002003 2002004
   18.989   -.921  10.000 -10.000        1 1001002 1002001 1002002
   10.000 -10.000   8.979 -11.021        1 1001002 1002002 1001003
  -10.000 -10.000 -11.019  -8.971        1 1001004 1002004 1001005
  -11.019  -8.971 -20.000    .100        1 1001005 1002004 1002005
  -18.989    .562 -10.000  10.000        1 2001002 2002001 2002002
  -10.000  10.000  -8.979  11.021        1 2001002 2002002 2001003
```

```
   10.000  10.000  11.019   8.930        1 2001004 2002004 2001005
   11.019   8.930  20.000   -.500        1 2001005 2002004 2002005
   20.000    .100  18.989   -.921        1 1001001 1002001 1001002
LEVEL     3.000
    2.973 -17.027    .000 -20.000        1 1001003 1002002 1002003
     .000 -20.000  -7.023 -12.977        1 1001003 1002003 1001004
   -7.023 -12.977 -10.000 -10.000        1 1001004 1002003 1002004
   -2.973  17.027    .000  20.000        1 2001003 2002002 2002003
     .000  20.000   7.023  12.977        1 2001003 2002003 2001004
    7.023  12.977  10.000  10.000        1 2001004 2002003 2002004
   12.983  -6.987  10.000 -10.000        1 1001002 1002001 1002002
   10.000 -10.000   2.973 -17.027        1 1001002 1002002 1001003
  -10.000 -10.000 -17.013  -2.917        1 1001004 1002004 1001005
  -17.013  -2.917 -20.000    .100        1 1001005 1002004 1002005
  -12.983   6.868 -10.000  10.000        1 2001002 2002001 2002002
  -10.000  10.000  -2.973  17.027        1 2001002 2002002 2001003
   10.000  10.000  17.013   2.636        1 2001004 2002004 2001005
   17.013   2.636  20.000   -.500        1 2001005 2002004 2002005
   20.000    .100  12.983  -6.987        1 1001001 1002001 1001002
END
```

Sectionals File Format

< ---.SEC file>	::= <one level's sectionals>* END
<one level's sectionals>	::= LEVEL <Z-value> <a sectional description>*
<a sectional description>	::= SECTIONAL <line-segment description>***
<line-segment description>	::= <X-value> <Y-value> <type of line-segment (from this point to the next) >
<type of line-segment>	::= <standard line-segment> ::= CONFLICT ::= BLANK
<standard line-segment>	::= POLYGON <a single polygon description>
<a single polygon description>	::= <point i.d. number>***

A Sample ---.SEC file

```
LEVEL   20.000
SECTIONAL
    .000 -20.000 POLYGON  1002003 1003003 1002004
   -.030 -19.970 POLYGON  1002004 1003003 1003004
 -10.000 -10.000 POLYGON  1002004 1003004 1002005
 -10.020  -9.980 CONFLICT
 -19.709   -.194 CONFLICT
 -10.000  10.000 POLYGON  2002002 2003002 2002003
  -9.980  10.020 POLYGON  2002003 2003002 2003003
    .000  20.000 POLYGON  2002003 2003003 2002004
    .030  19.970 POLYGON  2002004 2003003 2003004
  10.000  10.000 POLYGON  2002004 2003004 2002005
  10.020   9.979 CONFLICT
  19.709   -.194 CONFLICT
  10.000 -10.000 POLYGON  1002002 1003002 1002003
   9.980 -10.020 POLYGON  1002003 1003002 1003003
LEVEL    9.000
SECTIONAL
 -18.989    .562 POLYGON  2001002 2002001 2002002
 -10.000  10.000 POLYGON  2001002 2002002 2001003
```

```
  -8.979  11.021 POLYGON  2001003 2002002 2002003
    .000  20.000 POLYGON  2001003 2002003 2001004
   1.029  18.971 POLYGON  2001004 2002003 2002004
  10.000  10.000 POLYGON  2001004 2002004 2001005
  11.019   8.930 CONFLICT
  19.709   -.194 CONFLICT
  18.989   -.921 POLYGON  1001002 1002001 1002002
  10.000 -10.000 POLYGON  1001002 1002002 1001003
   8.979 -11.021 POLYGON  1001003 1002002 1002003
    .000 -20.000 POLYGON  1001003 1002003 1001004
  -1.029 -18.971 POLYGON  1001004 1002003 1002004
 -10.000 -10.000 POLYGON  1001004 1002004 1001005
 -11.019  -8.971 POLYGON  1001005 1002004 1002005
 -20.000    .100 BLANK
LEVEL    3.000
SECTIONAL
 -12.983   6.868 POLYGON  2001002 2002001 2002002
 -10.000  10.000 POLYGON  2001002 2002002 2001003
  -2.973  17.027 POLYGON  2001003 2002002 2002003
    .000  20.000 POLYGON  2001003 2002003 2001004
   7.023  12.977 POLYGON  2001004 2002003 2002004
  10.003  10.000 POLYGON  2001004 2002004 2001005
  17.013   2.636 CONFLICT
  19.709   -.194 CONFLICT
  12.983  -6.987 POLYGON  1001002 1002001 1002002
  10.000 -10.000 POLYGON  1001002 1002002 1001003
   2.973 -17.027 POLYGON  1001003 1002002 1002003
    .000 -20.000 POLYGON  1001003 1002003 1001004
  -7.023 -12.977 POLYGON  1001004 1002003 1002004
 -10.000 -10.000 POLYGON  1001004 1002004 1001005
 -17.013  -2.917 POLYGON  1001005 1002004 1002005
 -20.000    .100 BLANK
END
```

A Sample of a Reconstructed Object File

Since this file is in the format of the currently popular Utah graphics software (MOTION-DATARD), interested readers should refer to the internal Computer Science Dept. memos on the subject. Basically the file consists of 3-D point positions and polygons defined over the points. The specific structure of the file here is a sequence of blocks surrounded by "NAME=" and "END=" statements. The names used with these statements are L1, L2, L3, etc., indicating the Z-levels at which the reconstruction process was applied. Defined at each level in this file are the points that lie in that plane (notice that their Z values are identical) and the polygons that lie entirely in this plane (the top and bottom "cap" polygons) and the polygons which form the surface between connected sectionals on this level and sectionals on the preceeding level. ("↑" in the polygon definition sections refer to points at the preceeding level.) "COLTAB" commands indicate changes in the coloring of the polygons being defined. Colors of the polygons indicate the nature of the local surface region -- being either a 1) standard-measured surface, or 2) one which was initially blank but was later "filled in", or 3) one whose exact position was in conflict.

```
SMOOTH=NO
NAME=L1
BODY=SCENE
POINTS
1 1 0 0
2 0 1 0
3 0 0 1
4 0 0 0
5  .000 -20.000 -20.000
```

```
6 -.030 -19.970 -20.000
7 -10.000 -10.000 -20.000
8 -10.020 -9.980 -20.000
9 -19.709 -.194 -20.000
10 -10.000  10.000 -20.000
11 -9.980  10.020 -20.000
12  .000  20.000 -20.000
13  .030  19.970 -20.000
14  10.000  10.000 -20.000
15  10.020  9.979 -20.000
16  19.709 -.194 -20.000
17  10.000 -10.000 -20.000
18  9.980 -10.020 -20.000
POLYGONS
1  18  5  6  7
COLTAB 9
2  7  8  9  10  11  12  13  14  15  16  17  18
AXIS 1  2 4 3
AXIS 2  3 4 1
AXIS 3  1 4 2
END=L1
NAME=L2
POP=L1
BODY=SCENE
POINTS
1 -18.989  .562 -9.000
2 -10.000  10.000 -9.000
3 -8.979  11.021 -9.000
4  .000  20.000 -9.000
5  1.029  18.971 -9.000
6  10.000  10.000 -9.000
7  11.019  8.930 -9.000
8  19.709 -.194 -9.000
9  18.989 -.921 -9.000
10  10.000 -10.000 -9.000
11  8.979 -11.021 -9.000
12  .000 -20.000 -9.000
13 -1.029 -18.971 -9.000
14 -10.000 -10.000 -9.000
15 -11.019 -8.971 -9.000
16 -20.000  .100 -9.000
POLYGONS
1 ↑5  8 ↑6
2 ↑6  8 ↑7
3 ↑7  8  9
4 ↑7  9 ↑8
5 ↑8  9 ↑9
6 ↑9  9  10
7 ↑9  10 ↑10
COLTAB 6
8 ↑10  10  11
```

```
9 ↑10  11  12
10 ↑10  12 ↑11
11 ↑11  12  13
12 ↑11  13 ↑12
13 ↑12  13  14
14 ↑12  14  15
COLTAB 7
15 ↑12  15  16
16 ↑12  16  1
COLTAB 6
17 ↑12  1  2
18 ↑12  2  3
19 ↑12  3 ↑13
20 ↑13  3  4
21 ↑13  4 ↑14
22 ↑14  4  5
23 ↑14  5  6
COLTAB 9
24 ↑14  6 ↑15
25 ↑15  6  7
26 ↑15  7 ↑16
27 ↑16  7 ↑17
28 ↑17  7  8
29 ↑17  8 ↑18
30 ↑18  8 ↑5
END=L2
NAME=L3
POP=L2
BODY=SCENE
POINTS
1 -12.983  6.868 -3.000
2 -10.000  10.000 -3.000
3 -2.973  17.027 -3.000
4  .000  20.000 -3.000
5  7.023  12.977 -3.000
6  10.000  10.000 -3.000
7  17.013  2.636 -3.000
8  19.709 -.194 -3.000
9  12.983 -6.987 -3.000
10  10.000 -10.000 -3.000
11  2.973 -17.027 -3.000
12  .000 -20.000 -3.000
13 -7.023 -12.977 -3.000
14 -10.000 -10.000 -3.000
15 -17.013 -2.917 -3.000
16 -20.000  .100 -3.000
POLYGONS
COLTAB 7
1 ↑1  16  1
COLTAB 6
2 ↑1  1  2
```

```
3 ↑1  2 ↑2
4 ↑2  2  3
5 ↑2  3 ↑3
6 ↑3  3  4
7 ↑3  4 ↑4
8 ↑4  4  5
9 ↑4  5 ↑5
10 ↑5  5  6
11 ↑5  6 ↑6
COLTAB 9
12 ↑6  6  7
13 ↑6  7 ↑7
14 ↑7  7  8
15 ↑7  8 ↑8
16 ↑8  8  9
17 ↑8  9 ↑9
18 ↑9  9  10
COLTAB 6
19 ↑9  10 ↑10
20 ↑10  10  11
21 ↑10  11 ↑11
22 ↑11  11  12
23 ↑11  12 ↑12
24 ↑12  12  13
25 ↑12  13 ↑13
26 ↑13  13  14
27 ↑13  14 ↑14
COLTAB 9
28 ↑14  14  15
29 ↑14  15 ↑15
30 ↑15  15  16
COLTAB 7
31 ↑15  16 ↑16
32 ↑16  16 ↑1
COLTAB 6
33  14  13  12  11  10
COLTAB 9
34  14  10  9  8  7  6  5  4  3  2
35  2  1  16  15  14
END=L3
END= DATA
```

APPENDIX C

SAMPLE PROGRAM EXECUTIONS

Here is an actual Teletype listing of the software system in execution. This informal commentary is meant to serve as a substitute for those interested readers unable to see a live demonstration. It is assumed that the rest of the dissertation and the previous appendices have been read.

(The underlined parts are those typed in by the user.)

First, the raw 3-D data are acquired from a laser-scan of a scene of object(s) by executing NWRSL (or RSL) on the "Single-User" PDP-10, the computer which controls the laser deflection system and the spinning-disc detectors.

```
.RUN DTA1 RSL

WANT LASER SCAN AND DTA OUTPUT? Y
ARE YOU ON SINGLE USER ?(Y OR N) Y  GP10 DACS INITIALIZED.

DTA# & FILE NAME: (DTA#<-8 => NO-OUTPUT TESTING) 1 SMBOX

MIN. ELEVATION OF GOOD PTS.=?-1

DEFAULT SCAN (D=100,X=+-28,Y=+-30) ?
                                          OK ?(Y/N) N

LASER-OBJECT DIST.=?100

XMIN =? -25

OK ?(Y/N) Y

XMAX =? 10

OK ?(Y/N) N

XMAX =? 6

OK ?(Y/N) N

XMAX =? 2

OK ?(Y/N) Y
```

Since this particular program is an extension of Burton's Twinklebox system (as described in Chapter 3) the option still exists to run it with L.E.D,'s instead of with a laser; for now, we choose the laser.

Since program can also be tested on the Utah TENEX time-sharing system, the host system needs to be identified.

Specified next are the output device and file name for the coming data.

3-D points with vertical components less than the minimal elevation are ignored. (Setting this value to -1 -- 1 inch below the floor -- effectively suppresses this filter.)

The default scan area covers a region of approximately +/- 28 units by +/- 30 units at a distance of 100 units from the laser.

Declining this setting, we are allowed to specify each of the limits of the scan, to visually check them, for possible adjustment, before specifying the next limit value.

```
YMIN =? -10
OK ?(Y/N) N
YMIN =? -20
OK ?(Y/N) N
YMIN =? -26
OK ?(Y/N) Y
YMAX =? 0
OK ?(Y/N) Y
NXSTEPS, NYSTEPS=? 15 15
NSAMP>=MINSAM>=NCUTOF=? 10 7 6
SCAN ID. NM. =? 1
NM. OF TOTAL SCANS=? 1
WANT OUTPUT OF RAW MEASUREMENTS ?
                                        OK ?(Y/N) N
TRAPSS CALLED. IT IS NOT HERE AND NO LONGER NECESSARY
NUMBER OF L.E.D.S:    2
```

At each position of the laser the 3-D position of the surface spot is measured by the sensors and calculated a multiple number of times (for increased reliability and accuracy). [In this case, this number -- NSAMP -- is set to 10.] If the measurement attempt is successful for less than a prescribed number (in this case 7) of these attempts (it can fail if less than 3 of the sensors "see" the spot or the calculated position varies too far from the previously calculated position) then no 3-D position value is recorded for that position of the laser beam. To further minimize the effect of position values which stray too far from mean position, the final value is calculated from a subset of the values which lie closest to the initial mean value. (In this case, the 6 closest values are used in the final calculations, i.e. the 1 to 4 farthest ones are ignored.)

An identification number is used for each scan so that data from multiple scan of the same scene can be uniquely identified and thus used in the same reconstruction process.

The raw measurements (as discussed above) can be output for diagnostic purposes.

Setting the number of L.E.D.'s is a vestige of Burton's program. The only purpose it serves with the laser option is to automatically eliminate calculated positions which are more than a threshold (16 inches) away from the previous position calculated.

In addition to going onto the DECtape, the data can also be directed to another device; in this application, output to the Tektronic 611 storage scope is the most convenient, enabling easy monitoring of the scan's progress.

```
DATA TO TELETYPE, SCOPE, OR DISK (T,S,D): S

SCALE: 15

SINGLE OR QUADRANT: Q

GO? Y

ENDFILE AND CALL RELEAS ON  10
NM. OF BAD POSITIONS:    37

DTA# & FILE NAME: (DTA#<-8 => NO-OUTPUT TESTING) ↑C
.↑C
```

(Scale factor for putting data points onto the scope.)

Quadrant option puts the data onto the scope in three different axes -- XY, YZ, and ZX.

(The scan now takes place -- approx. 5-10 minutes.) The End-of-File written onto theDECtape file signals the completion of the scan.

The number of points rejected (as described above) out of the (40 x 40 =) 1600 total laser positions.

Another, new scan can now be specified.

Listed now are the original data files, as they would come from the digitizing software (NWRSL). As described more fully in Appendix B, these files start with a descriptor line, telling the number of samples on a scan line (5), the number of scan lines (3), and the scan identification number (1 or 2 here). Of course most actual scans contain several hundred points or more.

It should be noted that the data used here is simulated, generated to illustrate, with minimum distractions, much of the system's capabilities.

```
@COP TBOX1.DAT:4 (TO) TTY: [OK]
5 3 1
1001001 +.1   0.01 +20
1001002 -10   0.02 +10
1001003 -20   0.03 0
1001004 -10   0.02 -10
1001005 +.1   0.01 -20
1002005 +.1  10.01 -20
1002004 -10  10.02 -10
1002003 -20  10.03 0
1002002 -10  10.02 +10
1002001 +.1  10.01 +20
1003001 +.1  20.01 +20
1003002 -10  20.02 +10
1003003 -20  20.03 0
1003004 -10  20.02 -10
1003005 +.1  20.01 -20
@COP TBOX2.DAT:5 (TO) TTY: [OK]
5 3 2
2001001 -.5   0.01 -20
2001002 +10   0.02 -10
2001003 +20   0.03 0
2001004 +10   0.02 +10
2001005 -.5   0.01 +20
2002005 -.5  10.01 +20
2002004 +10  10.02 +10
2002003 +20  10.03 0
2002002 +10  10.02 -10
2002001 -.5  10.01 -20
2003001 -.5  20.01 -20
2003002 +10  20.02 -10
2003003 +20  20.03 0
2003004 +10  20.02 +10
2003005 -.5  20.01 +20
@NGENPO

INPUT FILE:TBOX1.DAT

WANT POLYGONS-ONLY FILE?Y
NEW POLYGON FILE NAME=?TBOX1.POL

WANT NO-GOOD POLYGONS' FILE?N

WANT POINT-TABLE FILE?
1866 DSK PAGES LEFT!!
FUCHS IS 315 PAGES OVER ALLOCATION!!
N

WANT MOTION-DATARD OUTPUT?Y
MOTION-DATARD FILE=?BN?TBOX1.MDR

WANT FILL-BLANK-POINTS ?N

NXSTEPS=5 NYSTEPS=3 IDSCAN=1000000
ENTER LIMITS OF WORKING VOLUME: XMIN,XMAX, YMIN,YMAX, ZMIN,ZMAX ?=
-100 100 -100 100 -100 100

ENTER 'BOTTOM OF OBJECT' HEIGHT (FOR COLORING USE) ?=0
SMOOTH=NO
NAME=SCAN
```

We first generate a skin of polygons (triangles, actually) over all adjacent points which are in the input file.

N -- We don't care to save (in some disk file) all those polygons which are no good -- those for which one or more defining points are missing. We know that all current points are good.

POINT-TABLE -- is a scan grid showing present/missing status of each point.

(Like some real-world creditors, this computer system mercilessly pesters over-extended clients.)

(Dashed lines indicate typing errors.)

TBOX1.MDR --This will be a point and polygon file of the just-defined surface in the format of the currently standard graphics software at Utah (MOTION-DATARD).

All points NOT within this defined volume are assumed to be incorrect and are discarded.

```
YNOW INITIALIZED TO 1 PTID,NEXT=1001001 PTID,NEXT=1001002 PTID,N+0

EXIT.
↑C
@COP TROXI.POL;2 (TO) TTY: [OK]
NXSTEPS=5 NYSTEPS=3 IDSCAN=1000000
1001001 1002001 1001002
1001002 1002001 1002002
1001002 1002002 1001003
1001003 1002002 1002003
1001003 1002003 1001004
1001004 1002003 1002004
1001004 1002004 1001005
1001005 1002004 1002005
1002001 1003001 1002002
1002002 1003001 1003002
1002002 1003002 1002003
1002003 1003002 1003003
1002003 1003003 1002004
1002004 1003003 1003004
1002004 1003004 1002005
1002005 1003004 1003005
@COP TROXI.MOR;2 (TO) TTY: [OK]
SMOOTH=NO
NAME=SCAN1
BODY=SCENE
POINTS
1   1 0 0
2   0 1 0
3   0 0 1
4   0 0 0
5   -30  -5  25
6   45  -5  25
7   45  -5  -30
8   -30  -5  -30
9   -30  5 -30
POLYGONS
COLTAB 5
1   5 6 7 8
2   7 8 9
3   9 8 7
AXIS 1   2 4 3
AXIS 2   3 4 1
AXIS 3   1 4 2
END=SCAN1
NAME=SON1
BODY=SCENE
POP=SCAN1
POINTS
1    .100    .010   20.000
2   -10.000   .020   10.000
3   -20.000   .030   .000
4   -10.000   .020  -10.000
5    .100    .010  -20.000
6    .100   10.010  -20.000
7   -10.000   10.020  -10.000
8   -20.000   10.030   .000
9   -10.000   10.020   10.000
10    .100   10.010   20.000
11    .100   20.010   20.000
12  -10.000   20.020   10.000
13  -20.000   20.030   .000
14  -10.000   20.020  -10.000
15    .100   20.010  -20.000
POLYGONS
1  1  10  2
```

The rest of this program is diagnostic messages; a ↑O ("CONTROL-O") disables this kind of output.

We now look at the polygon file just generated; it looks good -- just a list of input data points which define each polygon.

The large point i.d. numbers make it particularly easy to integrate data from many different scans into the same analysis structure and still maintain unique point i.d.'s.

We now look at the standard graphics format file just generated. It contains, in addition to the data in the previous point and polygon files, a definition of orthonormal vectors for easy rotation manipulations when later viewing this object. This file also defines two additional polygons -- a "floor" and a "tab" on this floor -- to aid visual orientation when displaying the object(s).

The point values here match those of the original input files, except this graphic system requires consecutive point i.d.'s, starting with 1.

```
2  2  10  9
3  2  9  3
4  3  9  8
5  3  8  4
6  4  8  7
7  4  7  5
8  5  7  6
9  10  11  9
10  9  11  12
11  9  12  8
12  8  12  13
13  8  13  7
14  7  13  14
15  7  14  6
16  6  14  15
END=SONI
END=DATA
@NGENPO
```

We generate the polygons file for the second original point file. (Exactly the same as for the first file.)

```
INPUT FILE:TBOX2.DAT

WANT POLYGONS-ONLY FILE?
1837 DSK PAGES LEFT!!
FUCHS IS 317 PAGES OVER ALLOCATION!!
Y
NEW POLYGON FILE NAME=?TBOX2.POL

WANT NO-GOOD POLYGONS' FILE?N

WANT POINT-TABLE FILE?N

WANT MOTION-DATARD OUTPUT?Y
MOTION-DATARD FILE=?TBOX2.MDR

WANT FILL-BLANK-POINTS ?N

NXSTEPS=5 NYSTEPS=3 IDSCAN=2000000
ENTER LIMITS OF WORKING VOLUME: XMIN,XMAX, YMIN,YMAX, ZMIN,ZMAX ?=
-100 100 -100 100 -100 100

ENTER 'BOTTOM OF OBJECT' HEIGHT (FOR COLORING USE) ?=0
SMOOTH=NO
NAME=SCAN
 YNOW INITIALIZED TO 1 PTID,NEXT0
```

For experimentation, let's alter the value of one of the original points -- from -.5 to -87.5.

```
EXIT.
↑C
@POET
 POET   4-DEC-74 DSP
#READ TBOX2.DAT (OLD VERSION)
#2EDIT
2001001  -.5     0.01  -20
2001001  -<87.5    0.01  -20
#/
2001001  -87.5    0.01  -20
#1,5/
5 3 2
2001001  -87.5    0.01  -20
2001002  +10     0.02  -10
2001003  +20     0.03  0
2001004  +10     0.02  +10
#OVERWRITE TBOX2.DAT.7 (NEW VERSION)
#FINI
@NGENPO
```

We'll again execute NGENPO to get a polygon mapping for this altered data file.

```
INPUT FILE:TBOX2.DAT
```

```
[illegible]

WANT POLYGONS-ONLY FILE?
1799 DSK PAGES LEFT!!
FUCHS IS 320 PAGES OVER ALLOCATION!!
Y
NEW POLYGON FILE NAME=?TV\VBOX2.POL
```

We'll now look at a "no good" polygons file to check if the polygon depending on this altered (and to be discarded) data point is also discarded.

```
WANT NO-GOOD POLYGONS' FILE?Y
NEW 'FAILED' POLYGONS' FILE NAME=?TBOX2.NGP

WANT POINT-TABLE FILE?Y
POINT-TABLE FILE=?TBOX2.PTB
```

We'll also look at ---.PTB file and check for the discarded position.

```
WANT MOTION-DATARD OUTPUT?Y
MOTION-DATARD FILE=?TBOX2.MDR

WANT FILL-BLANK-POINTS ?N
```

Hopefully by defining the acceptable volume, the previously altered point will be judged unacceptable.

```
NXSTEPS=5 NYSTEPS=3 IDSCAN=2000000
ENTER LIMITS OF WORKING VOLUME: XMIN,XMAX, YMIN,YMAX, ZMIN,ZMAX ?=
-25 25 -25 25 -25 25

ENTER 'BOTTOM OF OBJECT' HEIGHT (FOR COLORING USE) ?=0
SMOOTH=NO
NAME=SCAN
 Y NOW INITIALIZED TO 1 PTID+0

EXIT.
↑C
@COP TBOX2.NGP:1 (TO) TTY: [OK]

2001001 2002001 2001002
@COP TBOX2.PTB:2 (TO) TTY: [OK]
NXSTEPS=5 NYSTEPS=3 IDSCAN=2000000
3=XXXXX
2=XXXXX
1=,XXXX
@SCANA
```

As expected the discarded polygon definition appears in this "discarded" list (it's the only entry).

Looking at the point-table file we see (by the X's) that most of the positons have valid entries. The only missing position (markded by a ",") is the first sample in the first row of the scan -- the very point we severely altered.

```
MUMMY MODE? (Y/N)N

POINT FILE          1=TBOX1.DAT

DESCRIPTION:5 3 1
TOTAL NM. PTS.=        15
WANT SORTED POINTS? (Y/N)N

POLYGON FILE         1=TBOX1.POL
```

SCANA - This will accept an arbitrary number of point and polygon files and generate the line-segment intersections between positions of the cut-plane (analysis plane) and the data polygons.

```
DESCRIPTION:NXSTEPS=5 NYSTEPS=3 IDSCAN=1000000
TYPE DISCARDED POLYGONS? (Y/N)N

MAX.GROUND-LEVEL?
1745 DSK PAGES LEFT!!
FUCHS IS 324 PAGES OVER ALLOCATION!!
0

TOTAL NM. POLYGONS=        16
TOTAL NM. POLYGONS CONSIDERED=        16
POINT FILE         2=TBOX2.DAT

DESCRIPTION:5 3 2
TOTAL NM. PTS.=       3?
WANT SORTED POINTS? (Y/N)Y

(        1)= 1001001  2?.???    .1??     .21?
```

One diagnostic output is listed here -- the original points (from all input files) sorted by their I.d. numbers.

```
(        2)= 1001002  10.000 -10.000      .020
(        3)= 1001003    .000 -20.000      .030
(        4)= 1001004 -10.000 -10.000      .020
(        5)= 1001005 -20.000    .100      .010
(        6)= 1002001  20.000    .100    10.010
(        7)= 1002002  10.000 -10.000    10.020
(        8)= 1002003    .000 -20.000    10.030
(        9)= 1002004 -10.000 -10.000    10.020
(       10)= 1002005 -20.000    .100    10.010
(       11)= 1003001  20.000    .100    20.010
(       12)= 1003002  10.000 -10.000    20.020
(       13)= 1003003    .000 -20.000    20.030
(       14)= 1003004 -10.000 -10.000    20.020
(       15)= 1003005 -20.000    .100    20.010
(       16)= 2001001 -20.000 -07.500      .010
(       17)= 2001002 -10.000  10.000      .020
(       18)= 2001003    .000  20.000      .030
(       19)= 2001004  10.000  10.000      .020
(       20)= 2001005  20.000   -.500      .010
(       21)= 2002001 -20.000   -.500    10.010
(       22)= 2002002 -10.000  10.000    10.020
(       23)= 2002003    .000  20.000    10.030
(       24)= 2002004  10.000  10.000    10.020
(       25)= 2002005  20.000   -.500    10.010
(       26)= 2003001 -20.000   -.500    20.010
(       27)= 2003002 -10.000  10.000    20.020
(       28)= 2003003    .000  20.000    20.030
(       29)= 2003004  10.000  10.000    20.020
(       30)= 2003005  20.000   -.500    20.010
POLYGON FILE        2=TBOX2.POL

DESCRIPTION: NXSTEPS=5 NYSTEPS=3 IDSCAN=2000000
TYPE DISCARDED POLYGONS? (Y/N)
1723 DSK PAGES LEFT!!
FUCHS IS 324 PAGES OVER ALLOCATION!!
Y

MAX.GROUND-LEVEL?.015

TOTAL NUM. POLYGONS=        31
TOTAL NUM. POLYGONS CONSIDERED=        31
POINT FILE        3=

TYPEOUT POINTS  ?  (Y/N)N

TYPEOUT POLYGONS  ?  (Y/N)N

TYPEOUT HIGH & LOW Z'S  ?  (Y/N)N

TYPEOUT SORTED Z'S  ?  (Y/N)Y

         1  20.030  10.030       12
         2  20.030  10.020       13
         3  20.030  10.020       14
         4  20.030  10.030       27
         5  20.030  10.020       28
         6  20.030  10.020       29
         7  20.020  10.020       10
         8  20.020  10.020       11
         9  20.020  10.010       15
        10  20.020  10.010       16
        11  20.020  10.020       25
        12  20.020  10.020       26
        13  20.020  10.010       30
        14  20.020  10.010       31
        15  20.010  10.010        9
```

Another example of diagnostic output -- the high and low Z values in each polygon -- sorted by the high Z value. (This is useful for determining which polygons intersect with a cutting-analysis plane at a given Z level.

```
        16  20.010  10.
1722 DSK PAGES LEFT!!
FUCHS IS 324 PAGES OVER ALLOCATION!!
010        24
        17  10.030     .030      4
        18  10.030     .020      5
        19  10.030     .020      6
        20  10.030     .030     19
        21  10.030     .020     20
        22  10.030     .020     21
        23  10.020     .020      2
        24  10.020     .020      3
        25  10.020     .010      7
        26  10.020     .010      8
        27  10.020     .020     17
        28  10.020     .020     18
        29  10.020     .010     22
        30  10.020     .010     23
        31  10.010     .010      1
SCOPEFACTORS:
XLOW= -20.000
XHIGH=  20.000
YLOW= -57.500
YHIGH=  20.000
XORIGIN= 512.000
YORIGIN= 630.030
XC=    7.153
YC=    7.153
ZHIGH=  20.030  ZLOW=     .010 ZLEVEL=?20

TYPE EDGES AT ZLEVEL=  20.000 ? (Y/N)
TYPE EDGES AT ZLEVEL=  20.000 ? (Y/N)Y

   9.980 -10.020     .000 -20.000          1      12
    .000 -20.000    -.030 -19.970          1      13
   -.030 -19.970  -10.000 -10.000          1      14
  -9.980  10.020     .000  20.000          1      27
    .000  20.000     .030  19.970          1      28
    .030  19.970   10.000  10.000          1      29
  19.990     .090  10.000 -10.000          1      10
  10.000 -10.000    9.980 -10.020          1      11
 -10.000 -10.000  -10.020  -9.980          1      15
 -10.020  -9.980  -20.000    .100          1      16
 -19.990    -.490 -10.000  10.000          1      25
 -10.000  10.000   -9.980  10.020          1      26
  10.000  10.000   10.020   9.970          1      30
  10.020   9.970   20.000   -.500          1      31
  20.000     .100  19.990    .090          1       9
 -20.000    -.500 -19.990   -.490          1      24
DISPLAY EDGES ? (NOTE: 'E' TO EXIT FROM DISPLAY).(Y/N)N

WANT SGMFILE? (Y/N)Y

WANT NEW SGM FILE?Y

WANT TO CLOSE EXISTING FILE?N
 SGM FILE NAME=?TROY12.SGM
 NM.SEGMENTS=          16
FINISHED OUTPUT THIS LEVEL. WANT TO CLOSE FILE NOW?N

MULTI-LEVEL DISPLAY? (Y/N)N

ZHIGH=  20.030  ZLOW=     .010 ZLEVEL=?9

TYPE EDGES AT ZLEVEL=   9.000 ? (Y/N)
TYPE EDGES AT ZLEVEL=   9.000 ? (Y/N)N
```

The program now requests a trial Z value for a cutting plane. This program is usually run on a Tektronix 4012 graphic terminal; so the intersection line-segments at each Z level can be observed on the screen.

Here we just list the line-segments, in format X1, Y1 X2, Y2, DIRECTION (not important here), and POLYGON I.D.

Here we put these line-segments into the newly-created segments file ---.SGM.

```
DISPLAY EDGES ? (NOTE: 'E' TO EXIT FROM DISPLAY) (Y/N)N

WANT SGMFILE? (Y/N)Y

WANT NEW SGM FILE?N
 NM.SEGMENTS=      15
FINISHED OUTPUT THIS LEVEL. WANT TO CLOSE FILE NOW?N

MULTI-LEVEL DISPLAY? (Y/N)N

ZHIGH=  20.230  ZLOW=     .010 ZLEVEL=?3

TYPE EDGES AT ZLEVEL=   3.000 ? (Y/N)
TYPE EDGES AT ZLEVEL=   3.000 ? (Y/N)N

DISPLAY EDGES ? (NOTE: 'E' TO EXIT FROM DISPLAY) (Y/N)N
1 523 DSK PAGES LEFT!!
FUCHS IS 324 PAGES OVER ALLOCATION!!
```

After selecting several other levels for the cutting-analysis plane, we close the ---.SGM file.

```
WANT SGMFILE? (Y/N)Y

WANT NEW SGM FILE?N
 NM.SEGMENTS=      15
FINISHED OUTPUT THIS LEVEL. WANT TO CLOSE FILE NOW?Y

MULTI-LEVEL DISPLAY? (Y/N)↑C
.COP TROYIE.SGM:1 (TO) TTY: [OK]
```

We next list the ---.SGM file we've just created. It not only gives the definition of each line-segment at each level, but it also preserves the definition of the polygon which generated this line-segment. Although this information is not used in the present implementation, a more sophisticated one could take advantage of this (line-segment to original polygon) mapping to control the explosive growth of the number of new polygons generated in the later object-reconstruction section.

```
LEVEL   20.000
   9.987 -10.020      .000 -20.000        1 1002003 1
    .000 -20.000     -.030 -19.970        1 1002003 1
   -.030 -19.970  -10.000 -10.000         1 1002004 1
  -9.987  10.020      .000  20.000        1 2002003 2
    .000  20.000      .030  19.970        1 2002003 2
    .030  19.970   10.000  10.000         1 2002004 2
  19.997    .003   10.000 -10.000         1 1002002 1
  10.000 -10.000    9.987 -10.020         1 1002002 1
 -10.000 -10.000  -10.020  -9.987         1 1002004 1
 -10.020  -9.987  -20.000    .100         1 1002005 1
 -19.997   -.499  -10.000  10.000         1 2002002 2
 -10.000  10.000   -9.987  10.020         1 2002002 2
  10.000  10.000   10.020   9.979         1 2002004 2
  10.020   9.979   20.000   -.500         1 2002005 2
  20.000    .100   19.997    .003         1 1002001 1
 -20.000   -.500  -19.997   -.499         1 2002001 2
LEVEL    9.000
   8.979 -11.021      .000 -20.000        1 1001003 1
    .000 -20.000   -1.029 -18.971         1 1001003 1002003 1001004
  -1.029 -18.971  -10.000 -10.000         1 1001004 1002003 1002004
  -8.979  11.021      .000  20.000        1 2001003 2002002 2002003
    .000  20.000    1.029  18.971         1 2001003 2002003 2001004
   1.029  18.971   10.000  10.000         1 2001004 2002003 2002004
  18.989   -.921   10.000 -10.000         1 1001002 1002001 1002002
  10.000 -10.000    8.979 -11.021         1 1001002 1002002 1001003
 -10.000 -10.000  -11.019  -8.971         1 1001004 1002004 1001005
 -11.019  -8.971  -20.000    .100         1 1001005 1002004 1002005
 -18.989    .552  -10.000  10.000         1 2001002 2002001 2002002
 -10.000  10.000   -8.979  11.021         1 2001002 2002002 2001003
  10.000  10.000   11.019   8.930         1 2001004 2002004 2001005
  11.019   8.930   20.000   -.500         1 2001005 2002004 2002005
  20.000    .100   18.989   -.921         1 1001001 1002001 1001002
LEVEL    3.000
   2.973 -17.027      .000 -20.000        1 1001003 1002002 1002003
    .000 -20.000   -7.023 -12.977         1 1001003 1002003 1001004
```

```
  -7.023 -12.977 -10.000 -10.000        1 1001004 1002003 1002004
  -2.973  17.027    .000  20.000        1 2001003 2002002 2002003
    .000  20.000   7.023  12.977        1 2001003 2002003 2001004
   7.023  12.977  10.000  10.000        1 2001004 2002003 2002004
  12.983  -5.987  10.000 -10.000        1 1001002 1002001 1002002
  10.000 -10.000   2.973 -17.027        1 1001002 1002002 1001003
 -10.000 -10.000 -17.013  -2.917        1 1001004 1002004 1001005
 -17.013  -2.917 -20.000    .100
1625 DSK PAGES LEFT!!
     1 1001005 1002005FUCHS IS 325 PAGES OVER ALLOCATION!!
4 1002005
 -12.983   5.850 -10.000  10.000        1 2001002 2002001 2002002
 -10.000  10.000  -2.973  17.027        1 2001002 2002002 2001003
  10.000  10.000  17.013   2.635        1 2001004 2002004 2001005
  17.013   2.635  20.000   -.500        1 2001005 2002004 2002005
  20.000    .100  12.983  -6.987        1 1001001 1002001 1001002
END
?MAKSEC

SEGMENTS IN-FILE NAME?TBOX12.SGM

SECTIONALS OUT-FILE NAME?TBOX12.SEC

WANT TO ALWAYS CALL CONNECT?Y
```

We now call the program which organizes the simple line-segments into closed contours (sectionals) at each Z level. (Much diagnostics follow.)

```
ZLEV=     .000 DIAG AT: BEFORE READALEVEL        1 [Y,N,#]10
 <LEVEL>  <9.980>  <.000>  <-.030>  <-9.930>  <.000>  <.030>  <10.000>
**<10.000>  <-10.000>  <-10.020>  <-10.000>  <-10.000>  <10.000>  <10.02
**0>  <20.000>  <-20.000>  <LEVEL>
 ISEG=       1
 MDIF=  -2.000
ZLEV=  20.000 DIAG AT: ENTER TRYINTERSECTION: INTSEG=        1 [Y,N,#]100

 MDIF=  -2.000
 MDIF=   2.000
 MDIF=   -.043
 MDIF=   1.952
 MDIF=  -1.905
 MDIF=    .043
 MDIF=  -1.935
 MDIF=   -.300
 MDIF=  -1.943
 NEWAI2=      17
 NEWBI2=     -18
 BISEG=       5
 LOOKI2=       7
 AISEG=       8
 LOOKI2=      10
 MDIF=   1.990
 MDIF=   1.952
 ISEG=        9
 MDIF=   1.980
 MDIF=   1.942
 NEWAI2=     -19
 NEWBI2=      20
 AISEG=       7
 LOOKI2=       8
 BISEG=      10
 LOOKI2=      11
 MDIF=   1.899
 MDIF=   -.081
 MDIF=   -.091
 ISEG=       11
 MDIF=   -.910
 MDIF=  -1.990
 MDIF=  -2.000
```

More diagnostics (solely for the system designer's benefit)

```
 MDIF=  -1.999
 MDIF=  -1.990
 MDIF=  -2.000
 MDIF=  -1.999
 MDIF=   2.000
 MDIF=    .010
 MDIF=    .091
 MDIF=   2.000
 MDIF=    .091
 MDIF=  -2.000
 MDIF=  -2.000
 MDIF=  -1.999
ZLEV=  20.000 DIAG AT: SWAPPED ISEG       15 (Y,N,#)1100

BEFORE CONNECTLENGTH: TIP TAIL DIST
        1        1      .014
        1        2   39.994
        2        1   39.994
        2        2      .015
WANT DIAG AT ENTER-CONNECTLENGTH?          1
1573 DSK PAGES LEFT!!
FUCHS IS 325 PAGES OVER ALLOCATION!!
0

FINAL-MINTOTAL=      .029
        1        1      .014
        2        2      .015
 IATIP=          1
 IATAIL=          1
 IATIP=         2
 IATAIL=         2 <8.979>  <.000>  <-1.029>  <-8.979>  <.000>  <1.029>  <
**10.989>  <10.000>  <-10.000>  <-11.019>  <-10.989>  <-10.000>  <10.000
**>  <11.019>  <20.000>  <LEVEL>
 ISEG=        1
 MDIF=  -2.000
 MDIF=  -2.000
 MDIF=   2.000
 MDIF=   -.048
 MDIF=   1.952
 MDIF=  -1.935
 MDIF=    .048
 MDIF=  -1.935
 MDIF=   -.000
 MDIF=  -1.943
 MDIF=   -.038
 ISEG=        8
 MDIF=    .038
 MDIF=   1.943
 NEWAI2=      -16
 NEWBI2=       17
 AISEG=        7
 LOOKI2=        7
 BISEG=        9
 LOOKI2=        8
 MDIF=    .038
 MDIF=   -.000
 MDIF=   1.980
 MDIF=  -1.943
 MDIF=  -1.980
 MDIF=   -.000
 MDIF=  -1.952
 MDIF=  -1.990
 MDIF=   -.010
 MDIF=   -.010
 MDIF=    .048
 MDIF=    .010
```

The characters between "<" and ">" are the first tokens encountered in each input line -- printed out as the program first reads them in. (It's an easy way to follow the program's progress.) We can see that the program reads segment descriptors until it reaches a new "LEVEL" indicator.

```
 MDIF=   2.000
 MDIF=   1.990
 MDIF=    .048
ZLEV=   0.000 DIAG AT: ENTER TRYINTERSECTION: INTSEG=      12 [Y,N,#]100

 MDIF=   2.000
 MDIF=   1.990
 MDIF=  -1.952
 MDIF=  -2.000
 MDIF=   -.010
BEFORE CONNECTLENGTH: TIP TAIL DIST
      1       1   1.112
WANT DIAG AT ENTER-CONNECTLENGTH?        IN

FINAL-MINTOTAL=    1.112
      1       1    1.112
 IATIP=        1
 IATAIL=          1 <2.973>  <.000>  <-7.023>  <-2.973>  <.000>  <7.023>  <
**12.083>  <10.000>  <-10.000>  <-17.013>  <-12.083>  <-10.000>  <10.000
**>  <17.013>  <20.000>  <END>
 ISEG=        1
 MDIF=  -2.000
 MDIF=   2.000
 MDIF=  -2.000
 MDIF=   -.048
 MDIF=   1.952
 MDIF=  -1.905
 MDIF=    .048
 MDIF=  -1.905
 MDIF=   -.000
 MDIF=  -1.942
 MDIF=   -.038
 ISEG=        8
 MDIF=    .038
 MDIF=   1.943
 NEWAI2=      -15
 NEWBI2=       17
 AISEG=        7
 LOOKI2=        7
 BISEG=        9
 LOOKI2=        9
 MDIF=  -1.943
 MDIF=  -1.980
 MDIF=   -.000
 MDIF=    .038
 MDIF=   -.000
 MDIF=   1.980
 MDIF=   1.980
 MDIF=  -1.952
 MDIF=  -1.990
 MDIF=   -.010
 MDIF=   -.010
 MDIF=    .048
ZLEV=   3.000 DIAG AT: ENTER TRYINTERSECTION: INTSEG=      11 [Y,N,#]Y

DIAGNOSTIC FILE TO DSK OR TTY ?TTY
ZLEV=   3.000 DIAG AT: ENTER TRYINTERSECTION: INTSEG=      11
NUMBODIES=        2

      I   BODYTAIL   BODYTIP BODYACTIVES
      1       0         0         1
      2       0         0         2
NMACTIVES=        2

I XTAIL  GOODTAIL  XTIP  GOODTIP  TAILACTIVE TIPACTIVE  NEXTACTIVE  LAST
**ACTIVE
```

Here the program has encountered the END of the input file.

For illustr ative purposes, we allow diagnostic output this time.

```
        1         0         0         0         0        7        11        0        0
        2         0         0         0         0       12         8        0        0
NSEG=        15
MAXSEGMENTS=      1000

        I        X1        Y1        X2        Y2 DIRECTION   TYPE    YPT2 ISSEPT
**2
        1   -2.973   17.027      .000   20.000        1        4  17.027       1
        2     .000   20.000     7.023   12.977        1        5  12.977       2
        3  -10.000   10.000   -
1503 DSK PAGES LEFT!!
2.973  17.027        1       1FUCHS IS 326 PAGES OVER ALLOCATION!!
2  10.000          3
        4    7.023   12.977   10.000   10.000        1        6  10.000       4
        5  -12.983    6.868  -10.000   10.000        1       11   6.868       5
        6   10.000   10.000   17.013    2.636        1       13   2.636       6
        7   17.013    2.635   20.000    -.500        1       14   -.500     -16
        8  -17.013   -2.917  -20.000     .100        1       10  -2.917       8
        9   20.000     .100   12.983   -6.987        1       15  -6.987      17
       10  -10.000  -10.000  -17.013   -2.917        1        9 -10.000      10
       11   12.983   -6.987   10.000  -10.000        1        7 -10.000      11
       12   -7.023  -12.977  -10.000  -10.000        1        3 -12.977      12
       13   10.000  -10.000    2.973  -17.027        1        8 -17.027      13
       14   -7.023  -12.977     .000  -20.000       -1        2 -20.000      14
       15    2.973  -17.027     .000  -20.000        1        1 -20.000      15
       16   17.013    2.536   19.779    -.194        1       -2 -20.000      15
       17   19.779    -.194   12.983   -6.987        1       -2    .000       0

        I        C         M   NEXTSEG LASTSEG
        1  -20.000     1.000        2       3
        2   20.000    -1.000        4       1
        3  -20.000     1.000        1       5
        4   20.000    -1.000        6       2
        5  -19.524      .952        3      -1
        6   19.523     -.952       16       4
        7   19.524     -.952       -1       6
        8  -19.901     -.990       -2      10
        9   19.901      .990        9       9
       10  -19.901     -.990        8      12
       11   19.900      .990       -1      17
       12  -20.000    -1.000       10      -2
       13   20.000     1.000       14      10
       14  -20.000    -1.000       15      13
       15   20.000     1.000       12      14
       16   19.524     -.952       :7       6
       17   19.901      .990       11      16
 MDIF=      .010
ZLEV=    3.000 DIAG AT: ENTER TRYINTERSECTION: INTSEG=       12 (Y,N,#)100

 MDIF=    2.000
 MDIF=    1.990
 MDIF=   -1.952
 MDIF=   -2.000
 MDIF=    -.010
 MDIF=     .048
 MDIF=    2.000
 MDIF=    1.990
BEFORE CONNECTLENGTH: TIP TAIL DIST
        1         1   9.749
WANT DIAG AT ENTER-CONNECTLENGTH?         1N

FINAL-MINTOTAL=    9.749
        1         1   9.749
 IATIP=        1
 IATAIL=        1
EXIT.
```

The sectional-making program (MAKSEC) exits.

```
^C
@COP T30X12SECNCNFNS.SEC:1 (T0) TTY: [OK]

LEVEL   20.000
SECTIONAL
    .000 -20.000 POLYGON  1002003 1003003 1
   -.030 -19.970 POLYGON  1002004 1003003 1
 -10.000 -10.000 POLYGON  1002004 1003004 1
 -10.020  -9.980 CONFLICT
 -19.709   -.194 CONFLICT
 -10.000  10.000 POLYGON  2002000 2003000 2
  -9.980  10.020 POLYGON  2002003 2003002 2
    .000  20.000 POLYGON  2002003 2003003 2
    .030  19.970 POLYGON  2002004 2003003 2
  10.000  10.000 POLYGON  2002004 2003004 2
  10.020   9.970 CONFLICT
  19.709   -.194 CONFLICT
  10.000 -10.000 POLYGON  1002002 1003002 1
   9.980 -10.020 POLYGON
1002003 1003002 1003003
LEVEL    9.000
SECTIONAL
 -10.980    .552 POLYGON  2001002 2002001 2
 -10.000  10.000 POLYGON  2001002 2002002 2
  -8.970  11.021 POLYGON  2001003 2002002 2
    .000  20.000 POLYGON  2001003 2002003 2
  1.029888 DSK PAGES LEFT!!
  10.971E1 POLYGON  2001004 2002003 2002004
  10.030  10.000 POLYGON  2001004 2002004 2
  11.019   8.937 CONFLICT
  19.709   -.194 CONFLICT
  10.030   -.921 POLYGON  1001002 1002001 1
  10.000 -10.000 POLYGON  1001002 1002002 1
   8.970 -11.021 POLYGON  1001003 1002002 1
   .0000 -20.000 POLYGON  1001003 1002003 1
  -1.029 -10.971 POLYGON  1001004 1002003 1
 -10.000 -10.000 POLYGON  1001004 1002004 1
 -11.019  -8.971 POLYGON  1001005 1002004 1
 HS IS 327 PAGES OVER ALLOCATION!!
-20.000    .100 BLANK
LEVEL    3.000
SECTIONAL
 -12.983   5.858 POLYGON  2001002 2002001 2002002
 -10.000  10.000 POLYGON  2001002 2002002 2001003
  -2.973  17.027 POLYGON  2001003 2002002 2002003
    .000  20.000 POLYGON  2001003 2002003 2001004
   7.023  12.977 POLYGON  2001004 2002003 2002004
  10.000  10.000 POLYGON  2001004 2002004 2001005
  17.013   2.535 CONFLICT
  19.709   -.194 CONFLICT
  12.983  -5.987 POLYGON  1001002 1002001 1002002
  10.000 -10.000 POLYGON  1001002 1002002 1001003
   2.973 -17.027 POLYGON  1001003 1002002 1002003
    .000 -20.000 POLYGON  1001003 1002003 1001004
  -7.023 -12.977 POLYGON  1001004 1002003 1002004
 -10.000 -10.000 POLYGON  1001004 1002004 1001005
 -17.013  -2.917 POLYGON  1001005 1002004 1002005
 -20.000    .100 BLANK
END
@OBREC

INPUT (SEC) FILE:T30X12.SEC

OUTPUT (DATARD) FILE NAME:T30X12.MDR
-AT-ENTER-DSK OUT,BADENTER=0
WANT 'PLASTER' ?Y
```

We now see the results of the sectional-building programs. The sectionals produced carry with them the information about their history. They are either marked 1) with definions of the polygons which generated them, 2) as "filler" (figure) polygons -- generated to fill in unconnected boundary regions, or 3) as conflicts -- part of a boundary region which had at least two different line-segments trying to define it.

Each line contains an X,Y value pair and the description of the line segment between it and the following point.

This file represents a contour description of all the (1 here) object(s) found in the scene.

We now run the object reconstruction program to fit a polygonal "skin" over this contour description.

```
<LEVEL> ... <SECTIONAL> ...
** .000BEFOREBTAG,YTEMP=-20.0000BTAG,IPT=5 <P↑0

EXIT.
↑C
PCOP TRDX12.MDR:1 (TO) TTY: [OK]
SMOOTH=NO
NAME=L1
BODY=SCENE
POINTS
1 1 0 0
2 0 1 0
3 0 0 1
4 0 0 0
5  .000 -20.000 -20.000
6 -.030 -19.970 -20.000
7 -10.013 -10.000 -20.000
8 -13.020 -9.987 -20.000
9 -19.799 -.194 -20.000
10 -10.000  10.000 -20.000
11  -9.980  10.020 -20.000
12   .000  20.000 -20.000
13   .030  19.970 -20.000
14  10.000  10.000 -20.000
15  13.020  9.970 -20.000
16  19.799 -.194 -20.000
17  10.000 -10.000 -20.000
18  9.980 -10.020 -20.000
POLYGONS
1  18  5  6  7
COLTAB 9
2  7  8  9  10  11  12  13  14  15  16  17  18
AXIS 1  2 4 3
AXIS 2  3 4 1
AXIS 3  1 4 2
END=L1
NAME=L2
POP=L1
BODY=SCENE
POINTS
1  -19.989   .592 -9.000
2  -10.000  10.000 -9.000
3  -8.979  11.021 -9.000
4   .000  20.000 -9.000
5   1.029  18.971 -9.000
6   10.000  10.000 -9.000
7   11.019  8.931 -9.000
8   19.799 -.194 -9.000
9   18.989 -.921 -9.000
10  10.000 -10.000 -9.000
11   8.979 -11.021 -9.000
12   .000 -20.000 -9.000
13  -1.029 -18.971 -9.000
14  -10.000 -10.000 -9.000
15  -11.019 -8.971 -9.000
16  -20.000   .100 -9.000
POLYGONS
1  ↑5  8  ↑6
2  ↑6  8  ↑7
3  ↑7  8  9
4  ↑7  9  ↑8
5  ↑8  9  ↑9
6  ↑9  9  10
7  ↑9  10  ↑10
COLTAB 6
8  ↑10  10  11
```

We now lock at the completed reconstruction file. It's in the standard (MOTION-DATARD) graphics format (see Figures 4-14d and 4-14e).

```
9 ↑10 11 12
10 ↑10 12 ↑11
11 ↑11 12 13
12 ↑11 13 ↑12
13 ↑12 13 14
14 ↑12 14 15
COLTAB 7
15 ↑12 15 16
16 ↑12 16 1
COLTAB 6
17 ↑12 1 2
18 ↑12 2 3
19 ↑12 3 ↑13
20 ↑13 3 4
21 ↑13 4 ↑14
22 ↑14 4 5
23 ↑14 5 6
COLTAB 9
24 ↑14 6 ↑15
25 ↑15 6 7
26 ↑15 7 ↑16
27 ↑16 7 ↑17
28 ↑17 7 8
29 ↑17 8 ↑18
30 ↑18 8 ↑5
EN
1826 DSK PAGES LEFT!!
FUCHS IS 329 PAGES OVER ALLOCATION!!
D=L2
NAME=L3
POP=L2
BODY=SCENE
POINTS
1 -12.983 5.868 -3.000
2 -10.000 10.000 -3.000
3 -2.973 17.027 -3.000
4 .000 20.000 -3.000
5 7.023 12.977 -3.000
6 10.000 10.000 -3.000
7 17.013 2.635 -3.000
8 19.709 -.194 -3.000
9 12.983 -5.937 -3.000
10 10.000 -10.000 -3.000
11 2.973 -17.027 -3.000
12 .000 -20.000 -3.000
13 -7.023 -12.977 -3.000
14 -10.000 -10.000 -3.000
15 -17.013 -2.917 -3.000
16 -20.000 .100 -3.000
POLYGONS
COLTAB 7
1 ↑1 16 1
COLTAB 6
2 ↑1 1 2
3 ↑1 2 ↑2
4 ↑2 2 3
5 ↑2 3 ↑3
6 ↑3 3 4
7 ↑3 4 ↑4
8 ↑4 4 5
9 ↑4 5 ↑5
10 ↑5 5 6
11 ↑5 6 ↑6
COLTAB 9
12 ↑6 6 7
13 ↑6 7 ↑7
```

```
14 ↑7  7  8
15 ↑7  8 ↑8
16 ↑8  8  9
17 ↑8  9 ↑9
18 ↑9  9  10
COLTAB 6
19 ↑9  10 ↑10
20 ↑10  10  11
21 ↑10  11 ↑11
22 ↑11  11  12
23 ↑11  12 ↑12
24 ↑12  12  13
25 ↑12  13 ↑13
26 ↑13  13  14
27 ↑13  14 ↑14
COLTAB 9
28 ↑14  14  15
29 ↑14  15 ↑15
30 ↑15  15  16
COLTAB 7
31 ↑15  16 ↑16
32 ↑15  16 ↑1
COLTAB 6
33  14  13  12  11  10
COLTAB 9
34  14  10  9  8  7  6  5  4  3  2
35  2  1  16  15  14
END=L3
END= DATA
@
```



```
 DAY
 WEDNESDAY, MAY 21, 1975 05:20:11-MDT
@
```

REFERENCES

[1] G. J. Agin, "Representation and description of curved objects," Stanford Artificial Intelligence Lab., Memo AIM-173, Oct. 1972.

[2] G. J. Agin and T. O. Binford, "Computer description of curved objects," in *Proc. of the Third International Joint Conference on Artificial Intelligence*, Stanford, Calif., Aug. 1973, pp. 628-640.

[3] B. G. Baumgart, "Geometric modeling for computer vision," Stanford Artificial Intelligence Lab., Memo AIM-249, Oct. 1974.

[4] B. G. Baumgart, "A polyhedral representation for computer vision," in *Proc. of 1975 National Computer Conf.*, Anaheim, Calif., May 1975, pp. 589-596.

[5] T. O. Binford, "Visual perception by computer," *Proc. of the IEEE Conf. on Systems and Control*, Miami, Dec. 1971.

[6] R. P. Burton, "Real-time measurement of multiple three-dimensional positions," U. of Utah Computer Science Dept., Tech. Report UTEC-CSc-73-122, June 1973.

[7] L. Evans, Ph.D. Research Proposal, U. of Utah Computer Science Dept., 1974.

[8] A. D. Gara, R. F. Majkowski and T. T. Stapleton, "A holographic system for automatic surface mapping," General Motors Research Labs., Research Publication GMR-1342, Warren, Michigan, March 1973.

[9] Geodolite Laser Distance Measuring Instrument, Data Sheet, Spectra-Physics, Mountain View, Calif., Feb. 1969.

[10] R. E. Herron, "Biostereometric measurement of body form," *Yearbook of Physical Anthropology*, vol. 16, pp. 80-121, 1972.

[11] B. Julesz, "Toward the automation of binocular depth perception," *Proc. of IFIPS Congress, 1962*, North Holland, Amsterdam, pp. 439-443, 1963.

[12] M. D. Levine, D. A. O'Handley and G. M. Yagi, "Computer determination of depth maps," *Computer Graphics and Image Processing*, vol. 2, no. 2, pp. 131-151, Oct. 1973.

[13] R. Navatja and T. O. Binford, "Structured description of complex objects," in *Proc. of the Third International Joint Conf. on Artificial Intelligence*, Stanford, Calif., August 1973, pp. 641-647.

[14] F. I. Parke, "Computer generated animation of faces," U. of Utah Computer Science Dept., Tech. Report UTEC-CSc-72-120, June 1972.

[15] "Picosecond timing sharpens laser rangefinder resolution," *Electronic Design*, vol. 15, pp. 48-50, July 19, 1974.

[16] Y. Shirai and M. Suwa, "Recognition of polyhedrons with a rangefinder," in *Proc. of the Second International Joint Conf. on Artificial Intelligence*, London, September 1971, pp. 80-85.

[17] B. S. Spreight, C. A. Miles and K. Moledina, "Recording carcass shapes by a Moire method," *Medical and Biological Engineering*, pp. 221-226, March 1974.

[18] I. E. Sutherland, R. F. Sproull and R. A. Schumacker, "A characterization of ten hidden-surface algorithms," *ACM Computing Surveys*, vol. 6, no. 1, pp. 1-55, March 1974.

[19] H. Takasaki, "Moire topography," *Applied Optics*, vol. 9, pp. 1467-1472, 1970.

[20] D. Vickers, "The sorcerer's apprentice," U. of Utah Computer Science Dept., Tech. Report UTEC-CSc-74-078, July 1974.