DATA CAPTURE FROM ENGINEERING DRAWINGS

by

CARY BRADFORD SKIDMORE II
B.S., Kansas State University, 1983

A THESIS

submitted in partial fulfillment of the

requirements for the degree

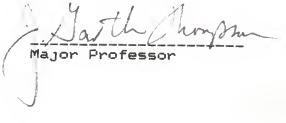
MASTER OF SCIENCE

Department of Mechanical Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1985

Approved by:



Major Professor

PREFACE

The thesis herein presented serves as a front end to a larger research project. The scope of this thesis is to introduce the overall problem and to report on the progress made in the data capture phase. The research project addresses the problem of converting two-dimensional engineering drawings on paper into a three-dimensional model in a computer aided design system. The data capture phase deals specifically with converting drawings on paper into a hierarchical point database in the computer. This type of database will later be used to to define a spline representation of the object.

Dr. J. Garth Thompson of the Department of Mechanical Engineering has been a very able supervisor for the project. A special thanks is due him for his expertise and the enthusiasm with which he served as my major professor on this project. Particular gratitude is felt from remembrance that he invited me to join him in graduate research work at a time when I was in need of direction in my professional life.

Dr. Richard Akins of the Chemical Engineering Department and Dr. C.L.D. Huang of the Mechanical Engineering Department have both served me well as instructors and committee members. Their influence on my professional development and attitudes is appreciated.

The graduate students, Terry Schmalzried and John Rasure, of the Department of Electrical and Computer Engineering have been especially helpful in learning the use of the Vax computer and Grinnell imaging systems. The Image Processing and Analysis Package (IPAP) written by John was a good introduction to the systems. Terry was very helpful with his spontaneous tutelage when difficulties were encountered.

A deep gratitude is felt for the support of my wife, Wendy. Certainly, a better concentration on professional pursuits is mine when her support is felt. I will be eternally grateful that she "hath chosen the good part" of wife and mother. The increased strain imposed on her by the extra time away from home required to finish this thesis is acknowledged, and her continual encouragement is appreciated. The brief breaks at home during a long span of time spent at school were made refreshing by her and our young son, Bradley. Her typing, proofreading, and editing skills are also appreciated.

The greatest example to me of professional excellence and while maintaining balance in other areas of life will always be my father, Dr. Edward Skidmore. His example has had a subtle yet profound influence on my life.

TABLE OF CONTENTS

Chapter		Page
I.	INTRODUCTION	1
II.	LITERATURE REVIEW	7
III.	COMMERCIAL DIGITIZING SYSTEMS	17
IV.	IMAGE CAPTURE	24
	Equipment	24
	Algorithms	28
V.	DATA CAPTURE	38
	Data Capture Algorithm	38
	Conversion from 2D to 3D	42
VI.	CONCLUSION AND RECOMMENDATIONS	45
LIST OF REFERENCES		49
APPENDIX		51

LIST OF FIGURES

Figure		Page
1.	Original View Drawing	29
2.	Histogram of Image, Two Filter Papers	31
3.	Histogram of Image, One Filter Paper	32
4.	Effect of Weighting Factor (WF) on Histogram, WF = .1111	36
5.	Effect of Weighting Factor (WF) on Histogram, WF = .07	37

CHAPTER I

INTRODUCTION

The increasingly popular flexible manufacturing systems and the fully automated factory concept provide a harmonious means of integrating computer aided design (CAD) with computer aided manufacturing (CAM). However, the tremendous backlog of paper based drawings of existing mechanical parts that are still being manufactured is an obstacle to such integration. The purpose of this thesis is to introduce a potential solution and to present preliminary work done on it.

Commercial systems are currently available that digitize two dimensional drawings for use in CAD systems. (These will be discussed in Chapter III.) However, the forte of the CAD industry is in three-dimensional model building. A 3D model can be viewed as a projection from any angle which allows the user all the convenience and simplicity of working in a pseudo-traditional two-dimensional environment while making other tasks much easier and faster. Additional views can be derived, design changes are effected in all views simultaneously, interference detection can be done on the computer rather than on a physical prototype, the programming of machining tools for 3D operation is easier, and many other design analysis functions are available on three-dimensional, model-based CAD systems.

The object of this work was to capture appropriate data from the three (or more) views used to describe an object in an engineering drawing, such that the data could be used to create a three-dimensional surface or solid model in a CAD database. A vidicon camera was used to digitize the views from which points were manually selected. A set of points associated with a particular line segment or curve was output to a computer. These points could be used to find a nonuniform, rational B-spline representation of the line segment or curve which could be employed in a compatible CAD data base to generate solid or surface models. These models in turn could be used in driving computer aided manufacturing equipment. The matrix form of the spline lends itself well to CAD operations such as translation, rotation, and scaling of a model.

Much work has been done to convert information on paper based engineering drawings to two dimensional information in the computer and current state-of-the-art technology allows for nonuniform rational B-spline representation of curves and surfaces in CAD modeling systems. However, very little work has been done to convert multiple views on paper to a three dimensional model in a CAD system. Such an effort has two principal areas of focus. One is obtaining the data from the views in a useable format and the other is the correlating of the data from different views to develop a spline based model. This thesis deals with the former and leaves the latter to further research.

A general discussion of the digitization method chosen and the rationale for spline representation now follows. Reviews of the literature and current state-of-the-art digitizing equipment are given in the next two chapters. Subsequent chapters describe the equipment used and give details of the algorithms developed.

The common method of digitizing involves scanning a drawing with some type of device which either generates a digital pixel map of the entire scanned area directly (as with a solid-state camera) or digitizes a raster scan obtained with, for example, a vidicon camera. Pixel data or digitized raster data may be considered as a series of dots (or very small squares) that are intrinsically unrelated to each other like the picture on a television screen. The dots which make up the picture only have meaning to a person who relates them to one another with his vision. To be useful in a CAD system the data is converted to vector format.

Vectorizing a drawing relates associated points. A line segment is represented by the coordinates of its two end points and an equation that connects them rather than a series of disconnected dots. Curves may be represented by sequences of very short line segments. The raster to vector conversion is not an easy one and requires significant computing power (COMPUTER AIDED DESIGN REPORT, Vol. 2, No. 10, October 1982). The computer algorithms do not always make the best selection of points to represent a line and some interactive cleanup work is frequently

necessary. When the conversion is complete the vector data is stored and the raster data may be discarded.

The scanning device used for the work of this thesis is a vidicon camera. The image seen by the camera is digitized by a Grinnell system. The output is a pixel map or set of digitized raster data, however, a raster to vector type conversion is never made. Rather specific points are interactively selected and stored for use in a spline representation of the line or curve. Only the point data need be stored and the pixel map may be discarded.

A drawing that includes curved lines to represent a part can be, at best, only a good approximation when it is vectorized. A vector drawing cannot exactly define a curved line or surface. Spline representation has been suggested as a mathematical description that more closely approximates a curve. Traditional splines use knot points and polygon vertices that do not lie on the curve to define it. The implementation of this type of spline is generally a very tedious and cumbersome trial and error process.

Lozover (1982) suggested an automatic process to construct a cubic B-spline representation for a general curve. The algorithm he describes fits a digitized curve with a spline and saves over 90% of the storage required if all the points on the curve were saved. However, this type of spline is still only a good approximation of a curve and cannot exactly define conic sections which is often necessary in CAD/CAM applications.

Versprille (1975) describes a rational B-spline that can exactly define the complete set of conic sections. It is a "polynomial extension of the integral B-spline approximation method for ab initio specification of sculptured curves and surfaces." The rational scheme is obtained by representation of design data in the homogeneous coordinate system, making it very straightforward and simple to transform the curve to see how it would appear in other views. Rotations, translations, and scalings are all easily done with a matrix of data in the homogeneous coordinate system.

Other strengths of this type of spline are that changes are kept local and that the spline passes through every point chosen on the curve. Addition of a new point from the curve for the spline to get a better fit does not affect the entire spline but only the region near the point. The spline will pass exactly through the point. Rational B-splines offer exactness in free form curves in a matrix format that is easily manipulated. They are therefore very appealing.

A variety of CAD and CAM equipment is available on the market today and in any given factory automation scenario there must be compatiblity of information exchange between the two and within each. An effort to standardize graphic computer data format has resulted in the Initial Graphics Exchange Specification (IGES). Many graphics system vendors are making their equipment IGES compatible. IGES Version 2.0 includes provision for rational B-spline surface and curve representation

(Smith, 1983). Version 3.0 which adds solid modeling specifications is expected to include the rational B-spline representation also. So this is yet another motivation to orient this thesis to the implementation of said spline.

CHAPTER II

LITERATURE REVIEW

The general problem of "converting the large collection of existing drawings into computer readable form" and the need for work in this area was well introduced by Oldfield (1980). He indicated that to make such a conversion using a manual transcription or a digitizing tablet would require a prodigious effort so a better method was needed. He proposed a "knowledge-based" or "syntax-guided" system that used automatic pattern recognition of logic schematics. While this method may work well for logic schematics which contain a relatively small number of recognizable symbols, it does not lend itself well to the general class of engineering drawings of mechanical objects due to their great variety in shape.

It is important to note the following statement that Oldfield makes with regard to the value of a fully automatic or semi-automatic method. "While full automation might seem desirable, it is unlikely to be as effective as an interactive method, due to drawing defects such as a break in a line or to the fact that in producing the original data the designer felt the need to depart from strict conventions. In consequence, particular attention must be paid to the user/computer graphical

interface so that the user may be aware of the computer's progress and generally direct it."

Ishii (1981) discusses automatic input and interactive editing systems that are pattern recognition based and applied to logic circuit diagrams. Mohr (1978) and Lin (1976) did similar work using hand drawn diagrams. The work of Zavidovique (1980) also concerned automatic recognition of electronics schematics and he discussed the preprocessing of line drawings which is of particular interest.

Zavidovique comments that "electronics schemes are usually well contrasted . . . so that a mere thresholding allows dissociating drawings from background." The first two steps of preprocessing he identifies as "grey-level scale compression" and thresholding. The grey-level scale compression seems to be a type of low pass filter. Each pixel is replaced by the sum of the grey levels of all or part of the neighboring points. Generally a low pass filter replaces a pixel with a fractionally weighted average of the values in the square array containing it. The effect of a low pass filter on a digitized engineering drawing is to smooth the noise which is desirable, but to blur the lines which is undesirable. The best binary image is produced when the lines are sharp and well contrasted.

The effect of the method used by Zavidovique is to make the center of a relatively light area lighter, up to the maximum intensity or saturation level. He indicates that this saturation effect allows the threshold selection to be non-adaptive. The

grey-level scale compression "reduces the [threshold] choice while increasing sensibility. It is well verified, in practice, that this value remains constant concerning the same series of drawings."

Logic diagrams consist of lines and shapes which have meaning of themselves. Engineering drawings of mechanical objects consist of lines and shapes which only have meaning as they are associated with edges and surfaces. There is a notion of inside and outside. For example, a circle could represent a solid cylinder or a hole in an object depending on whether it is considered to bound material inside of it or outside. Such ambiguities are resolved by giving multiple views. Nevertheless, a great complexity is introduced that is not experienced when dealing with logic diagrams.

Cugini (1978,1) recognized this added complexity and did some work in computer generation and recognition of engineering drawings of mechanical objects. He restricted his work to simplified drawings where dimensions, notes, symbols, etc. were not considered. His approach was to associate two qualities with each point: belonging or not belonging to the object, and visible or invisible.

Shirai (1982) described some methods for capturing 3D data from 2D drawings. Image processing techniques were used for the data capture but prior knowledge about the object was necessary in order to give line characteristics such as obscuring, concave, and convex. Sugihara (1979) suggested an algebraic approach that

used edge inequalities similar to the line characteristics mentioned by Shirai. The work of both men was limited to polyhedral objects.

Liardet (1978) proposed graphical input to a CAD system using a program called PICAX (Polyhedral Input to the Computer by AXonometric projections). This "is a program which derives a three dimensional description from a single two dimensional line drawing of a polyhedral object." The single view is an axonometric projection of an object that can be built out of polyhedra. An axonometric projection is one in which three orthogonal axes are all visible as lines emanating from a distinct vertex. For example, a cube set on one of its corners could be viewed as an axonometric projection for a variety of orientations. PICAX requires that all edges of the object, whether hidden or visible, be shown in the drawing. No vertices may coincide with other vertices or edges, and the axes of the drawing must be clearly marked.

The drawing is digitized on a tablet while the program runs. After input the program generates possible faces which are then sorted to find actual faces. This is done using "a lot of general knowledge about polyhedra." The 3D description is verified by producing various views of the object.

This thesis uses an approach that is not limited to polyhedral shapes. Objects such as airplane fuselages or wings that are described by lofting lines are included for consideration.

Cugini (1978,2) does not treat the digitizing aspect but he does propose some ideas for development of an interactive drafting system. Where Liardet was restricted to polyhedra for creating objects, Cugini is restricted to a set of two dimensional primitives and logic operators. He discusses the advantages of this method over the common vectorial description.

The subsequent work of Cugini (1982) discussed two dimensional representations of three dimensional objects as projections. This concept is used in the work of this thesis. A good discussion of the respective dimensions of an object, its modeling space, and modeling primitives and the relationships between them is given. The description of the system and data structures is helpful background for understanding how the work of this thesis might need to be structured for use in a CAD system.

Hosaka (1982) proposed generating 3D computer models of solid objects through handwritten drawings. Additional information is a necessary input "because the computer lacks the ability to recognize complex drawing patterns and the knowledge to understand drawing meaning." Some of the additional information required is a separate coordinate system for each primitive body and a coordinate system with the origin point given in each view. A similar use of coordinate systems to correlate views is used in the work of this thesis. (Although primitive bodies are not used.) The overall method of Hosaka is perhaps similar to generating a model complete with manufacturing

process specifications on a current constructive solid geometry based CAD/CAM system except that he does it with the "freedom" given by paper and pencil. He considers it "almost impossible to extract the meaning of a drawing from a computer's image data" that has been captured through a video camera and image processing techniques.

Lafue (1978) used orthographic views drawn on a digitizing tablet to retrieve the implied 3D structure. Polyhedra serve as the principal input. However, curved surfaces, such as Bezier's patches, are allowed because planarity of a face is not inherently enforced. The user single-points every vertex during digitization. Single-pointing allows the views to be independent where in double pointing the vertex must be identified in two views. The only commom input relating the views is the scale which must be the same, and their position with respect to ground lines defined by the user. Digitization of each view can be done at a different time or on a different tablet. The user does not need to keep in mind the 3D representation during the process. No distinction is made between hidden lines and faces and visible ones.

The program to do all this is called ORTHO and the general algorithm involves generating all possible faces and sorting them into true and false ones. When the algorithm cannot distinguish a face as true or false it requests help from the user.

Clement (1980) discussed digitization of engineering drawings not by tablet but by a scanning device (such as the

camera used in this thesis). He introduces his discussion by addressing the need to computerize the backlog of traditional paper drawings so that they can be accessed and modified by computer aided design and drafting systems. It would also allow more compact and accessible archives of drawings.

He limits the scope of his work to computerizing drawings to get a 2D line model and leaves the 2D to 3D conversion problem to future research. The worth of such research is indicated by his statement which follows. "Higher levels aim to represent not the drawing, but the object that the drawing represents, either as a dimensioned outline or as a three dimensional model. The utility of such representations would be high, as they match better those produced by sophisticated design systems."

Clement's approach is to first separate the drawing area into regions where lines exist and where they do not. Areas containing lines are subdivided into simple and complex line structures. A track following algorithm is used for the simple areas while work on complex areas has not been developed yet. An extension of split and merge techniques is used to identify which points belong to a particular line or are a junction and to make the points fit together better.

Bocquet (1982) describes an automatic process to transform a given mechanical drawing into a 3D database using video cameras for input. After the video shooting an algorithm is applied to identify which points belong to each continuous line. Each line is then classified as a line segment or arc with associated

information of endpoints, center, and radius. The 2D model is then corrected to give accurate parallelism, verticality, horizontality, and continuity or connectivity. This segment data base is transformed into a relational data base by associating each segment with an appropriate closed contour. This data base is now a set of 2D primitives that is put through a set of production rules which serve as a knowledge base for an inference engine in an artificial intelligence (AI) scheme.

Once one view has been completely analyzed the inferred surfaces are projected into another view and compared with the respective 2D data base. The projected view is considered a connection matrix (intersection of surfaces). Generation of a 3D data base for a rigid solid object is then accomplished through iterative comparison. If a comparison indicates conformable views some geometric parameters are set and the process applied to the next view. If a conflict arises a back track operation controlled by the production rules makes new adjustments and comparisons.

Bocquet quotes Aristides A. G. Requicha as saying, "It is virtually impossible to specify and design reliable algorithms for computing properties of solids represented by drawings." Bocquet further says, "The knowledge required to infer 3D surfaces from their 2D projections is somewhat fuzzy, and cannot by expressed as an algorithm." Therefore, instead of using an algorithm in a classical programming language, an AI method was used. A knowledge base or set of production rules specific to a

limited set of mechanical objects was developed. This was used as input to a FORTRAN program. The entire system is considered a Pattern Directed Inference System. Future work is involved in expanding the knowledge base to accomodate more objects than just those made up of the following elementary surfaces: plane, cylinder, cone, torus, sphere.

A critical limitation of this approach is that objects that cannot be practically made from primitive surfaces are not included. The spline representation of a curve or surface cannot be used.

The most recent related work was published by Sakurai (1983). The purpose of his research was to create solid models of component geometry from two-dimensional orthographic projections. Three views were input via a digitizer. The emphasis of the work was on solid model construction which could later be integrated with existing technology for scanning drawings rather than on the digitizing per se. The basic procedure is similar to some of those mentioned previously in terms of candidate vertex, face, and object generation and the truth or falseness of the generations distinguished through a comparative process.

Sakurai makes a point of saying that the modeling method called constructive solid geometry is limited by its ability to convert models created by wire-frame CAD/CAM systems into solid models. His research effort was aimed at developing a modeling method that could interface wire-frame and solid modeling

systems. The rational B-spline method that would be used in an extension of this thesis is believed to be amenable to either or both types of systems.

The system developed by Sakurai allows only lines and arcs to be represented in the views with the hidden ones identified as such. Therefore only planar, cylindrical, conical, spherical or toroidal surfaces are found in the final model and they must also have their axes parallel to one of the axes of the body coordinate system. Again, the limitation of shapes disallows free form curves.

Sakarai limits his work to three orthographic views but suggests that the system is extendable to include more or fewer views, partial projections, auxillary and sectional views. He says this extension is under study.

CHAPTER III

COMMERCIAL DIGITIZING SYSTEMS

Several commercial digitizing systems are currently available. Some are very new on the market and are specifically designed for use on engineering drawings. Others are older systems based on alternate applications. Both types are reviewed and compared in this chapter.

Altman (1978) reported on a system built specifically for application to engineering drawings by Altman Associates, Inc. of Stamford, Connecticut. Input to the system may be from prints, transparencies or originals of any standard size (A to E or 8.5 x 11 to 34 x 44 inches) or a microfilm form (i.e. aperture card) of the drawing. The scanning device is laser based with a claimed absolute accuracy of .010 inches.

Altman observed that unenhanced, vectorized drawings made from a raster scan system exhibited jagged curves rather than smooth ones like in the original. Typical line following algorithms that can correct this problem have been very slow. He increased the speed of the process by implementing an interactive scan - line following procedure that could digitize an average D size drawing in about 8 seconds.

Altman suggests that digitizing systems that make a global raster scan to obtain raw information and then manipulate all

that information to vectorize and digitize it are self-defeating. The desired information exists inherently in vector form and only needs to be extracted from the raster scan. So he suggests doing an "intelligent" or interactive scan. The scanner first makes a high speed, relatively low accuracy pass over a small area of the drawing. If no information is found, no further time and no memory of empty data are wasted on that area. If some information is noticed, its position is stored in memory.

Once the position is stored, the system automatically switches to a high speed line follower mode. Each line is examined and appropriate data is determined and stored. The coordinates of the endpoints and line thickness are stored for straight line segments, radii and centers of curvature for arcs, and coordinates and the number of members for nodes and branch points. All nodes and branch points are fully explored to insure that each line element is followed to its terminus.

The line following scan has sufficient accuracy to ensure that there are no discontinuities between tracking areas. This is done by using a laser beam deflection scheme. A single beam is deflected into three beams, one of which is used to scan the drawing, the others are used to provide continuous dynamic calibration of the scan.

ANA Tech Corporation of Littleton, Colorado (Computer Aided Design Report, 1982) also makes a system that supports laser scanners. Most manufacturers produce laser scanners because they have already been developed and are in use by the publishing

industry. They are used to scan the pages of a newspaper at editorial offices and transmit the data over microwave links to printing plants miles away. These laser scanners have a resolution of 1000 lines per inch and will scan a document that is attached to either a rotating drum or a flat bed.

Although the ANA Tech system supports laser scanners made by other manufacturers, they also have a non-laser scanner made by their own Systems Group. This scanner uses a row of 9 charged coupled device (CCD) cameras to achieve the same resolution. The cameras are fixed and the drawing is pulled beneath them at a constant speed by rollers. Paper based drawings of any size up to 36 inches wide can be used on this system. Raster to vector conversion speed is about 8.3 square inches per second compared to the value of 70 obtained on the Altman system. This difference by about a factor of 10 may be due to a resolution increased by the same factor. The speed of the Altman system was also dependent on the complexity of the drawing where ANA Tech claims virtual independence of drawing density.

The ANA Tech system digitizes paper based line art and text, generates a vector file, and prepares an interactive data base for output to a computer graphics system. Data of different types may be placed into different "layers". This means that text, geometry, and dimension lines can all be placed into different layers and treated (e.g. displayed) separately. Text must be fixed font for the automatic recognition process to be able to replace them with character codes rather than store them

in vector form. Generated files are passed to CAD systems by putting them in IGES format.

Broomall Industries of Broomall, Pennsylvania has developed a digitizing system that primarily serves the mapping industry. For this reason the resolution is high (1000 lines per inch) but there are some drawbacks in application to engineering drawings. The document is placed on a 7 x 10 foot fixed table and a laser scanhead passes over it. No text or symbol recognition facility is available. The principle drawback is that it takes one and a half hours to digitize a single D size drawing as compared to 90 seconds for the ANA Tech system. Interface to CAD systems is not through IGES, but through a Calcomp 925 plot image format. Broomall offers a digitizing service where they will process drawings for a customer.

Scitex was developed by an Israeli company principally for the printing industry, but some modifications have made the system more useable by engineering firms. The drawing is placed on a large drum and digitized by a laser scanner. This takes 30 to 40 minutes for a D size drawing. The operator can do some editing to clean up the data before vector conversion. This includes restoring faded or smudged lines, or separating fused lines, as well as deleting undesirable noise. The vector conversion program can recognize various line widths and crosshatching and place them in separate data layers. Text can also be segregated to a separate layer but, no character or symbol recognition was in use as of 1982.

After vectorization the operator again has the opportunity to correct errors introduced by the process. For example, if a line that should be straight is represented as several line segments , a command called "union" can be invoked to correct it. The union routine scans the vector files for nodal points. If the angle between two lines is below a certain operator specified tolerance, the computer omits the node and creates a single line. Another example is the case of line intersections. A "+" intersection is vectorized to be four line segments joining at a common node, but for CAD purposes it might be desirable to represent it by two intersecting lines. Provision is made for such an adjustment.

Intergraph and Computervision have worked with Scitex to translate Scitex-generated vector files to the format of their respective CAD systems. Computervision's translator was developed by their mapping group, but is not limited to that application. It translates to the format of their CADDS4 software. Drawings can be digitized to Intergraph format by sending them to Chicago Aerial Survey in Des Plaines, Illinois.

Perhaps the newest company to join the ranks of digitizing equipment manufacturers is Skantek Corporation in Warren, New Jersey. The company formed in fall 1982 (ELECTRONICS, 1983) and is devoted specifically to producing an economic digitizer for entering engineering drawings into computer aided design systems. Most digitizing systems use laser scanners, but this one uses a 40 inch fiber optic scanhead containing 10,000 fiber-optic light

pipes (a resolution of 250 pixels per inch). These fibers are used to conduct light from an incandescent light bulb to the drawing while a second set picks up the light reflected from the drawing. These 10,000 picture elements are organized into a 100 by 100 linear grid that is 2 centimeters square. The data is still in analog (raster) form, so it is fed into a charged-coupled device camera to be digitized.

The digitized data is then vectorized and may be separated into various layers such as drawing text, border material, and graphics. The graphics layer may be further subdivided according to line weights. The system is not suited to digitizing drawings on microfilm aperture cards, but any type of drawing sheet material may be used. The drawing is rolled under the scanhead by stepping motor drivers. The visual threshold can be varied to accommodate changes in contrast due to changes in the document medium. For example, an ink-on-mylar drawing would use a different threshold than a pencil-on-paper drawing. The operator identifies the type of medium to the computer by selecting from a menu.

The Skantek system can handle any standard size drawing from A to E, as can all the other systems. Digitizing takes 10 to 30 minutes depending on the size and complexity of the drawing. The system can input to a CAD system through the de facto IGES industry standard or a specialty interface designed specifically for some of the larger CAD vendors.

The last company to be mentioned is Tera Corporation of Berkley, California (ELECTRONICS WEEK, 1984). They have developed an editing work station that does not digitize a drawing, but rather allows editing of a drawing as a raster bit map. Drawings stored on microfilm, sheet material (paper,mylar), or a CAD data base can be scanned and viewed on the editing station monitor. Then they can be edited or modified and reproduced on microfilm or sheet material, or used as input for a CAD data base. The Tera Corporation is mainly concerned with automated records systems and the editing station answers a need in that area. However, it does not serve well the need to have a drawing data base that is defined well enough to drive computer aided manufacturing processes.

The general observation may be made that all of the digitizing systems mentioned answer some need in the industrial design and manufacturing world. However, none of the systems have made any attempt to generate three-dimensional models from the digitized two-dimensional drawings. There is still a gap between 2D drawings not in the computer and 3D models in a CAD system. A bridge between them would be of great utility in the world of computer aided manufacturing.

CHAPTER IV

IMAGE CAPTURE

The first step in converting a 2D drawing on paper into a 3D representation in the computer is digitizing the drawing to obtain an image from which appropriate data may be extracted. This chapter describes the equipment used to obtain an image suitable for further processing. Considerations such as resolution, lighting, and drawing size that are products of the equipment used and affect the suitablilty of the image are also discussed. Finally, the algorithms used to convert a grey scale image given by the camera to a suitable binary image are discussed.

Equipment

The computer system used for this work was a VAX 11/750 with the VMS operating system. All algorithms were implemented in the FORTRAN 77 programming language. The VAX was supported in image capture and display functions by a system made by Grinnell Systems Corporation of San Jose, California. Image processing utilities and subroutines for the VAX, and Grinnell subroutines were used extensively. The name and function of each one used is listed in the appendix. All VAX image processing subroutine names begin with the letters "IM" (e.g IMOPEN, IMDISP, IMTRAN)

and all Grinnell subroutine names begin with "GR" (e.g. GRINIT, GRFAR, GRSEND).

A vidicon camera made by DAGE-MTI, INC. of Michigan City, Indiana was used with a Camera Control box made by MTI Television. The Targ/Gain and Pedestal switches were always left in manual mode. The lens on the camera was made by Nikon of Japan and functioned just like a photographic lens.

The camera was mounted on a light table called Illuma System Quartz Light Control and made by Bencher, Inc. of Chicago, Illinois. The table is made so that either front or back lighting may be used. Front lighting of a paper drawing is undesirable because of the great amount of light reflected from the white paper surface which overpowers the dark lines. Back lighting can have the same effect if the light is too bright. This light table has three settings for light intensity: high, low, and off. The best image was obtained with back lighting when the light intensity was set on low. The low setting could be further quantitized by inserting one or two pieces of xerox paper between the light and the drawing. As will be shown later, this further quantization had little effect on the quality of the image. The optimum lighting scheme may vary according to the paper of the drawing.

The drawing on the light table may be continuously digitized by the Grinnell and displayed on a monitor while the operator positions the drawing and focuses the camera. The monitor is made by Conrac of Covina, California and is used only to display

the images sent to it by the Grinnell. These images are generally 512 pixels by 512 pixels with one of 256 possible levels of grey for each pixel. The images taken with the camera are grey scale images. However, the Conrac monitor has three color planes, red, green, and blue. Other colors are displayed as various combinations of these color planes. The Grinnell can handle color images as well. All programming, image processing commands, data point locations, etc. are given or received via the monitor and keyboard connected to the VMS operating system. The Conrac monitor has no keyboard associated with it.

The only interactive Input/Output device associated with the Conrac monitor is a Cursor Control Unit made by Grinnell. Since this unit will be used in picking data points from the drawing, it deserves further description. The unit is a box with several switches, buttons, a dial, and a joystick, that in general may be controlled either manually or by a program.

The dial is used to select either the QUAD or ZOOM cursor mode. Selecting the QUAD mode enables a possibility of four plus-sign (+) cursors that can be manipulated independently or in groups. Selecting the ZOOM mode enables one cross-hair cursor whose cross-hairs are the length and width of the screen. All cursors can either be displayed or not displayed.

The zoom cursor, when not in the zoom mode, functions like the quad cursors which move over the image. When in the zoom mode, the zoomed image is panned around the cursor. Panning and cursor movement are accomplished through joystick control. The

zoom mode refers to a magnification of 2, 4, or 8 times the original image through direct pixel replacement. An individual pixel has a particular grey level intensity and when the zoom function is invoked that pixel is replaced with 4, 8, or 16 pixels of the same intensity. The replacement pixels are arranged in a square shape so the zoom factor must be a power of two.

There are four switches in the upper left portion of the Cursor Control box labeled 1, 2, 3, and 4. These are used to select any combination of the four quad cursors. The "1" switch must be on when the zoom cursor has been selected on the dial. The ON and OFF buttons are used in combination with the four select switches when using quad cursors. The STEADY and BLINK buttons are used in connection with the select switches to make the display of the selected cursor(s) blink or be continuous. The TRACK or ENTER button is used to locate cursor postions in pixel coordinates continuously or discretely.

The remaining two items on the Cursor Control box to be discussed are the function switches, FUNA and FUNB. These may be used by the operator as input signals to the program. For example, with FUNA and FUNB both on, one branch in the program would be taken where if only one was on, a different branch would be taken. The status of the switches is received when the ENTER button is pressed (another function of the ENTER button).

The Grinnell digitizes the image seen by the vidicon camera into a 512 x 512 array of picture elements (pixels). Typical

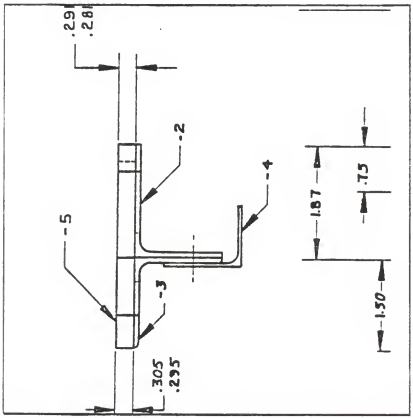
line widths on engineering drawings are 0.015 to 0.020 inches. A reasonable resolution is to represent a line 0.016 inches wide by two pixels (128 pixels per inch). A drawing 4 x 4 inches displayed by 512 x 512 pixels would give this resolution. Hence, only parts with views that fit into this area are used in this thesis. A larger drawing would require a digitizing system with a better resolution or one where multiple images of the same drawing could be correlated with sufficient accuracy.

Algorithms

The view drawing used in the work of this thesis is shown in Figure 1. The drawing was placed upon the light table for digitization. A program called CAMERA was executed. This program allows the operator to focus the camera and align the image while viewing it on the monitor. The image is continuously digitized in this manner until the RETURN key is struck, at which time the image is captured, stored in memory, and displayed on the monitor.

The grey scale image obtained from the camera may be used directly in the data capture phase since the data capture is interactive. The image does not need to be as clean when the operator is choosing data points visually as it does if a computer is doing the data capture automatically. However, a grey scale image requires much more memory to store than does a binary image so it was considered prudent to convert to a binary image before the data capture phase.

Figure 1. Original View Drawing



A useful tool in image analysis is the histogram. This is a graphical representation of the number of pixels at each of the 256 possible levels of grey. The program HISTO was used to make this calculation and plot the histogram. A histogram of the image obtained when two pieces of xerox paper were placed between the light and the drawing is shown in Figure 2. A similarly obtained histogram using one sheet of paper is shown in Figure 3. The histogram obtained when no extra paper was used was nearly a vertical line near a grey level of 255. A cursory comparison of these histograms might indicate that using two pieces of paper results in a more desirable image. There seems to be more separation of light background and dark lines. Or, at least it appears that a threshold choice would be less sensitive because a small difference in threshold would not affect as many pixels. This reasoning is based upon the fact that in typical image analysis the threshold choice for best contrast is the point in the valley between regions. In this case there is no valley but it seems logical to choose the point where the "mountain" begins to rise from the "plains".

A program called THRESH was written to try various threshold levels to see which actually results in the best binary image. The range of threshold levels that produced acceptable binary images was 215 to 235 when no extra paper was used to filter light and 175 to 195 when two pieces of paper were used. The base of the histogram peak in the first case was near 255 and was near 220 in the second case. The range of acceptable values was

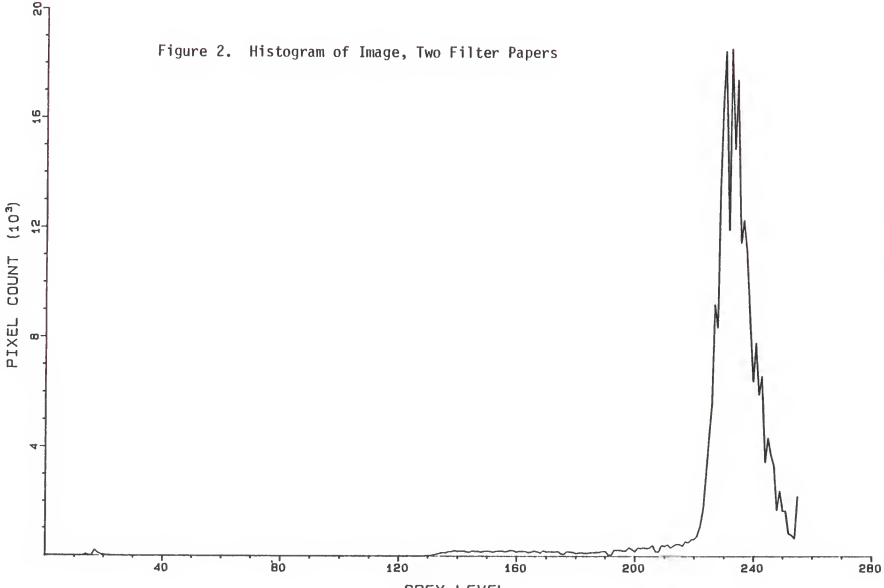


Figure 2. Histogram of Image, Two Filter Papers

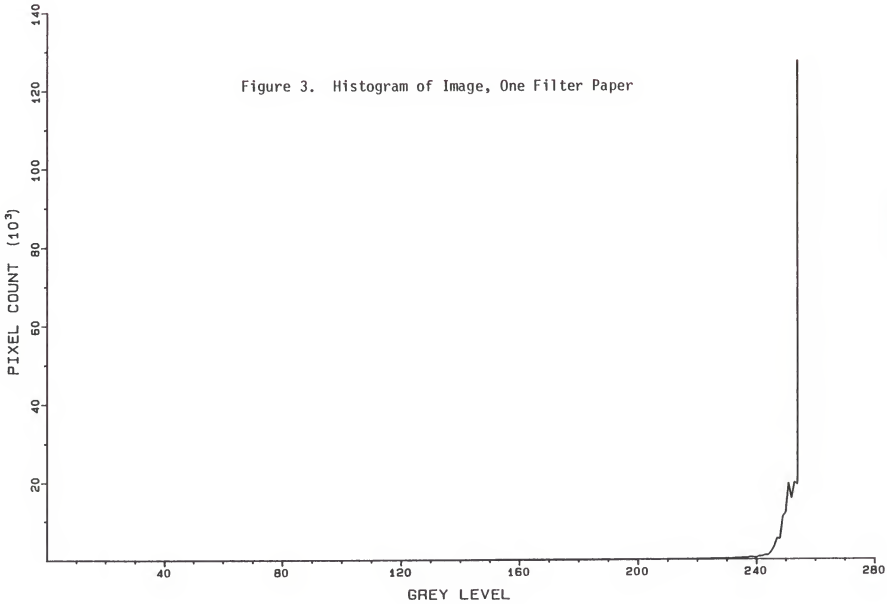


Figure 3. Histogram of Image, One Filter Paper

the same and the offset from the base of the peak was nearly the same in both cases. This means that the intensity of lighting within the range studied had no appreciable effect on the quality of the image obtained. The only effect was on the threshold value. Because the acceptable range was not near the "mountain" there was little effect on the sensitivity of threshold selection. The low end of the acceptable range occurred when the lines in the image began to be broken with spaces. The high end occurred when lines close together became fused in the binary image.

A histogram equalization algorithm was attempted to smooth the histogram and increase the contrast between lines and background, but this only served to blur the entire image considerably. Histogram equalization algorithms are generally used to improve the contrast in an image by spreading a concentration of grey levels over a larger range of values. However, with engineering drawings the concentrated levels are all background. Hence, an equalization does not improve contrast between lines and background but it distributes the background over nearly the entire range of grey which blurs the lines.

A low pass filter was also attempted. Generally such a filter is used to smooth local noise. The first filter attempt searched a 3 x 3 array for the pixel with the median grey level and assigned the center pixel this value of grey. All 3 x 3 arrays in the image were searched. This process did a good job of smoothing local noise but it also smoothed or blurred lines

that were not noise. Particularly blurred were drawing lines that were close together.

A type of filter mentioned by Zavidovique in Chapter II was attempted next. He used an unweighted sum to enhance digitized logic diagrams. When an unweighted sum was used on the drawing shown in Figure 1. (on a previous page), the "enhanced" image was all white. The logic diagrams used by Zavidovique may have been much more dense and therefore, had more dark pixels than in the image used here. The preponderance of background in Figure 1. rendered any unweighted scheme useless so a weighted sum was tried next.

The program called WEIGHT also operated on every 3 x 3 array in the image but each pixel in the array was given an equal weight. The center pixel was replaced with the limited sum of the weighted grey levels. If the sum was greater than the largest possible grey level (255), it was limited to that maximum value. The weight must be equal for all nine pixels in the array or there would be a bias to lines with a particular orientation. This algorithm served to make pixels in light areas lighter because they were replaced by pixels with larger values of grey. (The largest value of grey, 255, is totally white.)

The weighting factor should be chosen so that a threshold value can be chosen to obtain maximum contrast between the lines and the background. A weighting factor of 1 causes almost all pixels to be at the maximum grey level and a totally white image results. A weighting factor of .1111 (approximately 1/9)

produces an image with a smooth histogram. The effect of the weighting factor on the histogram can be seen in Figures 4 and 5. As the weighting factor decreases, the height of the peak increases and the width of the peak decreases. A weighting factor of .07 places the peak in the middle grey area and therefore gives a more shadowed background to the image.

The two filtering algorithms did not produce significantly different effects from each other. A smooth histogram makes the grey scale image less noisy but does not improve the binary image. The execution time for the median filter is about three minutes and the execution time for the weighted sum filter is about one and a half minutes. A suitable binary image is obtainable without the extra execution time required by the filtering algorithms.

This chapter has explained the equipment used in the work of this thesis. The equipment used to obtain a suitable image and the related factors that affect the capture of a quality image were discussed. Algorithms used to optimize the aquisition of a suitable binary image from the grey scale image obtained from the camera were presented. All programs mentioned in the chapter are documented in the appendix. The next chapter discusses the algorithms used to extract data from the image.

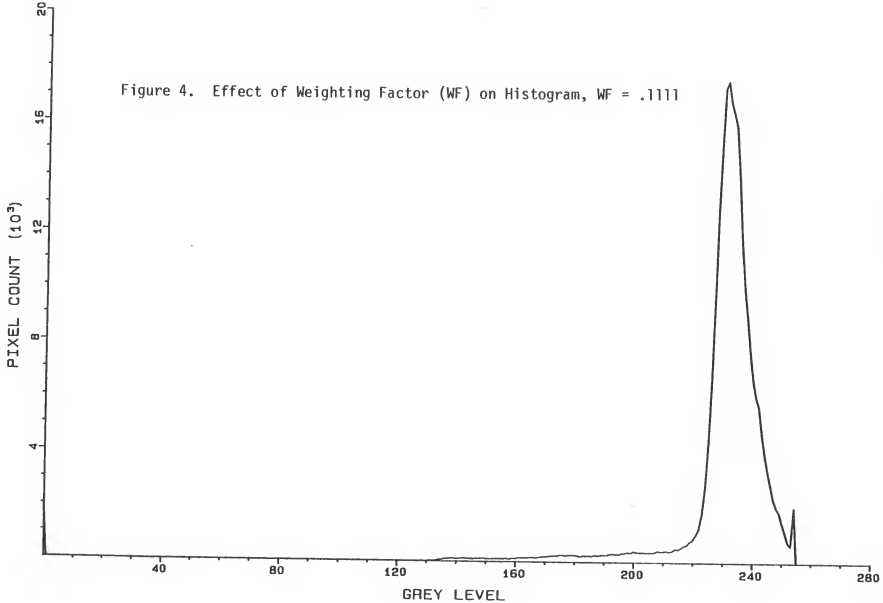


Figure 4. Effect of Weighting Factor (WF) on Histogram, WF = .1111

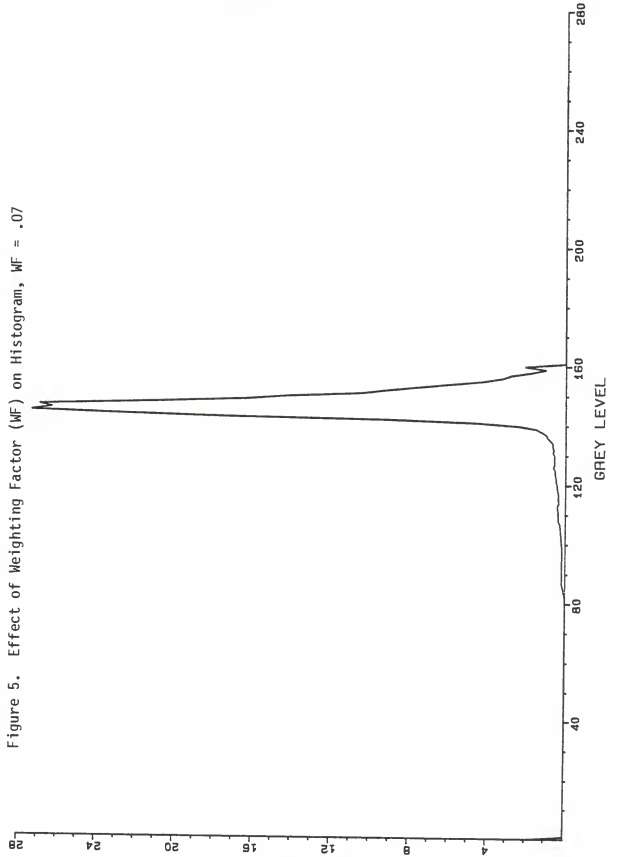


Figure 5. Effect of Weighting Factor (WF) on Histogram, WF = .07

CHAPTER V

DATA CAPTURE

The binary image obtained as discussed in Chapter IV is ready to be used to extract appropriate data. The purpose of this thesis is to present a mechanism for obtaining appropriate data for the computer from the drawing. The definition of appropriate data cannot be exactly defined until further research is done on the 2D to 3D conversion problem. The general concept of using splines to represent lines and curves is sufficient to understand the general mechanism for extracting data. That is to say that if splines are going to be used, then point data is the necessary data. Which points or how many points should be used is the detail which has not yet been defined. An algorithm to extract point data from the binary image is presented first in this chapter. The discussion of a possible method for 2D to 3D conversion follows next. Although this method is not proven, a discussion of it will add perspective to the possible ramifications of this thesis.

Data Capture Algorithm

The program written to manually capture data points from a binary image is called ZOOM. The distinguishing feature of this method is the use of the zoom cursor and zoom function in point selection and extraction.

The binary image of the view drawing to be used is displayed on the image monitor by invoking the VAX utility, DRAW. The program ZOOM is executed which immediately displays the zoom cursor and instructs the operator to check that the "1" switch is on and that ZOOM has been selected on the dial. Then the operator is instructed to locate the cursor in the region where data is to be taken. A zoom factor is chosen and the image is magnified. The operator is then able to observe each pixel distinctly and choose the most appropriate one.

The zoomed binary image is being displayed and no points have been selected yet. The operator now has four choices, one of which is indicated to the computer by the proper on/off combination of the FUNA and FUNB function switches. The choices are to capture a point, begin a new segment, change the zoom factor, or end the session. The computer does not read the switches until ENTER on the cursor control box is pressed.

The first choice presented is to capture a point. The intersection of the cursor cross-hairs may be placed over the desired point by using the joystick. The intersection point of the cross-hairs is exactly the size of one zoomed pixel. If the zoom factor is changed, the size of this point is changed accordingly. Therefore, it is very easy to know exactly which pixel is being selected. The data point is captured by pressing the ENTER button. This causes the selected point to be changed from black to red for visual reference, and the location of the

point in zoomed pixel coordinates to be read back to the computer.

The zoomed pixel coordinates refer to the location in the original image and not just the portion of the image currently being displayed. These coordinates may be used for scaling, orientation, and correlation of views as will be described later. The current version of ZOOM (listed in the appendix) numbers the points sequentially and displays the coordinates on the computer monitor.

The operator should capture data points in a sequential fashion along the curve. A set of such points can be used in a mathematical spline function to represent the curve. The program refers to such a set as a segment. The segments are numbered sequentially and displayed with the point numbers and pixel coordinates. The operator may begin a new segment or a new series of data points by using the correct combination of function switch settings and pressing ENTER. Beginning a new segment resets the point counter to zero. The points are numbered sequentially within a segment only.

The operator may desire to change the zoom factor for two reasons. One is to see more of the view by reducing the factor. This may be necessary in order to interpret a portion of the drawing correctly by seeing it in relation to the rest of the drawing. Some interpretation may be necessary to clarify line associations. After clarification is obtained the operator can go back to the segment he was working on without it having been

erroneously incremented. If the operator wishes to see more of the view for the purpose of moving to another region, a new segment must be started. The operator may also wish to increase the zoom factor to magnify the current segment to assist in selection of a pixel.

When the operator wishes to end the data capture session, both function switches must be turned off and ENTER pressed. The data captured during the session is displayed as described above. The data will need to be stored in a data file in order for it to be useful to a CAD system. The present program has made no attempt to output to a data file because the necessary file structure has not been defined. There must be allowances made for the insertion or deletion of points in proper sequence along a curve. The data structure must include more levels of hierarchy than only point and segment. Just as a set of points make a segment or line, a set of segments define a view or a surface, and a set of views or surfaces define an object.

The VAX utility, STOIMG, could be used to store the image with the selected data points shown, if a grey level had been used instead of red. The image is no longer a single color image and cannot be stored directly in one image file. The VAX stores the intensity (0 to 255) of each pixel of each color plane in a different file. There is a 512 x 512 array of intensity values stored for each color plane. So three files of 512 blocks each are required to store one color image.

However, there is an indirect way to store a color image in one file if less than 256 intensity levels of all color planes are used. For this case there are only six intensity levels. White is represented by an intensity level of 255 in each of the red, blue, and green planes. Black is represented by an intensity level of 0 in each plane. Red is represented by an intensity level of 255 in the red plane and 0 in the other two planes.

A program called RCOLOR was written to read the intensity of each color plane, run it through a ruling to encode the value, and store a single array of encoded values. The stored image must be run through a similar ruling before writing the image to the monitor for display. The ruling assigns an integer number to each pixel that indicates which color planes should be displayed at an intensity of 255 for that location. The programs RCOLOR and WCOLOR (to decode and write the image) are listed in the appendix and give the integer assignments for each color.

The encoding scheme has an execution time of 10 to over 70 minutes depending on the load on the system from other users. Reading the color planes is relatively fast (approximately 15 to 20 seconds) but merging three 512 x 512 arrays into one takes time. A similar length of time is required to decode the one array into three arrays in order to display the image.

Conversion from 2D to 3D

The following discussion concerns a potential method for converting 2D views on paper into a 3D object in the computer.

It is presented here for the perspective it gives to the data capture phase of the conversion.

The eventual correlation of multiple views requires that a common origin be established between all views before digitization or that the relationship between local origins of each view be known. Drawing common coordinate axes on each view prior to digitization helps align the drawing with respect to the camera. This simplifies the relationships between local coordinate axes, and between pixel coordinates and the local coordinates.

The local coordinates can be related to the pixel coordinates by finding the pixel location of three points; the local origin and a scale point on each axis. Data can then be extracted from the first view using a program like ZOOM. Greatest efficiency results when data is extracted first from the view containing the most information about the object.

Points in one view are displayed as lines in another view. The lines are displayed one at a time and the intersection points of the lines of the view and the lines representing points are digitized. Some of these intersection points may be ambiguous and others may be obviously valid points on the object. If there is only one intersection point, it is a valid point. If there is more than one, a third view is used to resolve the ambiguity. If there is still ambiguity, other views must be used or the operator must make a judgment. If the point in the second view was deemed valid this could be confirmed in the third view. As

ambiguities are resolved the invalid points are dropped from the point data set.

Artificial edges are not considered in this method. An artificial edge results from an object such as a cylinder. The cylinder appears as a rectangle when observed perpendicular to its axis and so the engineering drawing shows edges where there really are none. These "edges" appear at different places in different views. Further research is necessary to resolve this difficulty.

This chapter has described an algorithm for extracting point data from a binary image. A discussion of a potential method of converting 2D data to 3D data was discussed to show the application of the data capture algorithm.

CHAPTER VI

CONCLUSION and RECOMMENDATIONS

The need for research in converting two-dimensional engineering drawings on paper to three-dimensional models in a computer aided design system has been shown. An introduction to the direction of such research was presented with preliminary results.

A review of the literature showed that very little work has been done in converting free form curves on paper to a representation in the computer. The work has been limited to polyhedra, a set of geometric primitives, or a set of recognizable symbols (e.g. logic symbols).

A review of commercial digitizing systems available showed that there has been much work done in converting 2D engineering drawings on paper to 2D drawings in the computer. These systems are not fully automatic as they require operator interaction. The operator cannot feed drawings into the system, leave it unattended for a period of time, and then come back to a completely finished drawing in the computer. Very little work has been done by this industrial sector on the 2D to 3D conversion problem.

The acceptance by the CAD industry of curve and surface representation as nonuniform, rational B-splines was indicated by

the inclusion of this type of spline representation in the de facto industry standard for graphics data (Initial Graphics Exchange Specification). Nonuniform, rational B-splines were presented as an attractive representation because conic sections can be exactly defined by them, they pass exactly through each knot point, they can define free form curves, and because the CAD industry has begun their adoption. Additionally, the matrix representation of the spline in homogeneous coordinates facilitates coordinate transformations which are necessary in the 2D to 3D conversion.

The review of the literature and commercial digitizing systems provided a background from which ideas were developed to attack the 2D to 3D conversion problem. The equipment and algorithms used to capture a suitable image and prepare it for data extraction were presented in Chapter IV. A vidicon camera was used to scan an engineering drawing of a single view of an object which was then digitized.

The digitized view was displayed on a monitor with a zoom capability. The image could be zoomed, examined, and discrete pixels selected as data points to represent the line or curve. The algorithms to do this were presented in Chapter V. A potential method of converting the 2D data to 3D data was presented for the perspective given to the data capture phase.

The future research recommended by this thesis is a natural follow up of the data capture work herein presented. The 2D to 3D conversion method suggested seems to be a viable answer to the

problem of putting paper-based engineering drawings into a three-dimensional CAD database. Some specific recommendations based on the results of this work are given below.

The lighting facet of good image capture does not seem to be very critical. This is a fortunate circumstance because special lighting is often difficult or costly to achieve in a real world situation. Rather than pursue optimum lighting, it may be more worthwhile to investigate an automatic thresholding algorithm that will locate the base of the peak in a histogram and choose an offset that results in a good binary image. Such an algorithm should be tested by using drawings of various line density and under various lighting situations.

Since there are only two function switches on the cursor control box, only four combinations of settings are possible. If the data file is to be structured in a point, segment or line, view or surface hierarchy, then more function switches are required or the present uses must be achieved in another way. Perhaps the zoom factor can be changed or the session ended by another means. Perhaps all funtions could be menu driven from keyboard input.

When the 2D to 3D conversion algorithms have been developed, it may be useful to display the spline functions on the view in a green color. The spline points would be in red, the spline lines in green, and the original lines in black. This would give visual confirmation of the accuracy of the spline fit.

A practical digitizing system needs to accomodate standard size engineering drawings. The possibility of customizing a currently available digitizing system should be considered. Another possibility is to mount a camera on a highly accurate X-Y "plotting" machine or a single axis plotter that rolls the drawing underneath it. The multiple images acquired of a single view would have to be accurately correlated.

LIST OF REFERENCES

1. Altman, N.G., "Automatic Digitizing of Engineering Drawings", Electro '78 papers presented at the Electronic Show and Convention, IEEE Electrical Res. Assn., Boston, MA, May 1978.

2. Bocquet, J.C., and Tichkiewitch, S., "An 'Expert System' for Reconstruction of Mechanical Object from Projections", Advances in CAD/CAM (Proc. of 5th Int. IFIP in USSR), May 1982.

3. Clement, T.P., "The Extraction of Line-Structured Data from Engineering Drawings", Pattern Recognition, Vol. 14, Nos. 1-6, pp. 43-52, 1981, Printed in Great Britain.

4. COMPUTER AIDED DESIGN REPORT, Vol. 2, No. 10, October 1982.

5. Cugini, U., Mussio, P., Cavagna, C., and Meraviglia, A., "On an Image Generation and Recognition System", Artificial Intelligence and Pattern Recognition in Computer Aided Design, Latombe, ed., IFIP 1978.

6. Cugini, U., Dell'Oca, M., Mirioni, A., and Mussio, P., "An Interactive Drafting System Based on Bidimensional Primitives", Proc. Int. Conf. Interactive Techniques in Computer Aided Design, Bologna, Italy, Sept 21-23, 1978, IEEE ACM.

7. Cugini, U., Cosmai, G., Mussio, P., and Napolitano, A., "An Interactive Drafting System Based on Two Dimensional Primitives", ACM IEEE Nineteenth Design Automation Conf. Proc., Las Vegas, Nevada, June 14-16, 1982.

8. ELECTRONICS, Sept 8, 1983, p. 52, Oct 20, 1983, p. 14.

9. ELECTRONICS WEEK, Sept 3, 1984, pp. 165-166.

10. Hosaka, M., and Kimura, F., "Using Handwriting Action to Construct Models of Engineering Objects", Computer, Vol. 15, No. 11, Nov 1982, pp. 35-47.

11. Ishii, M.,et al., "Automatic Input and Interactive Editing Systems of Logic Circuit Diagrams", ACM Eighteenth Design Automation Conference Proceedings, June 1981.

12. Lafue, G., "A Theorem Prover for Recognizing 2D Representations of 3D Objects", Artificial Intelligence and Pattern Recognition in Computer Aided Design, Latombe, ed., IFIP 1978.

13. Liardet, M., Holmes, C., and Rosenthal, D., "Input to CAD Systems: Two Practical Examples", Artificial Intelligence and Pattern Recognition in Computer Aided Design, Latombe, ed., Grenoble, France, Mar 1978.

14. Lin, W.C., and Pun, J.H., "Machine Recognition and Plotting of Hand-Sketched Line Figures", Proc. IEEE Int. Conf. Cybernetics and Society, Nov 1976, Washington D.C.

15. Lozover, O., and Preiss, K., "Automatic Construction of a Cubic Spline Representation for a General Curve", Comput. and Graphics, Vol.7, No.2, pp.149-153, 1983.

16. Mohr, R., and Masini, G., "Drawing Analysis and Computer Aided Design", Artif Intell and Pattern Recognition in Comput Aided Des, IFIP, Mar 1978.

17. Oldfield, J.V., and Tudhope, D.S., "The Missing Link in an Interactive Graphics CAD System - Diagram Input", MIDCON/80 Conf Record, Nov 1980, Dallas, TX.

18. Sakurai, H., and Gossard, D., "Solid Model Input Through Orthographic Views", Computer Graphics, Vol. 17, No. 3, July 1983.

19. Shirai, Y., "Image Processing for Data Capture", Computer, Nov 1982, p. 21.

20. Smith, B.M., "IGES: A Key to CAD/CAM Systems Integration", IEEE Computer Graphics and Applications, Nov 1983, pp. 78-83.

21. Sugihara, K., "Algebraic Approach to the Analysis of Line Drawings of Polyhedral Scenes", Trans. Inst. Electron. and Commun. Eng. Jpn., Vol.E62, No.3, March 1979.

22. Vesprille, K.J., "Computer Aided Design Applications of the Rational B-Spline Approximation Form", a dissertation, Systems and Information Science, Syracuse University, Feb 1975.

23. Zavidovique, B., and Stamon, G., "An Automated Process For Electronics Scheme Analysis", Proc. 5th Int. Conf. on Pattern Recognition, Dec 1980.

APPENDIX

All computer programs were written in FORTRAN 77 and are listed in this appendix. A program is run by entering the following command: R(UN) fn, where fn is the program file name. VAX image processing utilities are not documented here but a description of how to get a listing is given. The utilities are run or executed by simply entering the program name. For example, entering ZOOM instead of R(UN) ZOOM would cause the utility, ZOOM, to be executed rather than the program by the same name.

The VAX image processing utilities are in the directory, UTILSRC:[IMAGEUTIL]. The command SET DEFault followed immediately by the directory name changes the default directory. Entering DIR .FOR will list all the FORTRAN files in the default directory. A file may be viewed on the monitor by typing, TYPE filename.FOR, or it may be printed by typing, PRINT filename.FOR. The VAX image processing subroutines may be obtained in a similar manner. They are in the directory, IMAGEDIR.

The following VAX image processing subroutines were used:

IMTRAN	transfer image file between VAX and Grinnell
IMDISP	read from or write to image display monitor
IMOPEN	open an image file for reading or writing
IMINIT	initialize Grinnell.

The following Grinnell subroutines were used:

GRSxxx	pertains to the system
INI	initialize
BFD	buffer dump
END	end session
GRDxx	pertains to the Image Video Digitizer
OP	control operation (single image, summation, etc.)
SH	control shifting of digitized data
TH	control threshold
DG	trigger digitization
GRZxx	pertains to zoom cursor
ON	zoom and pan function on/off
WO	pan window of 512 x 512 on/off
CO	cursor display on/off
FC	determines zoom factor
CW	wait for ENTER to be pressed
CR	read cursor location.

```
C  Program name: CAMERA.FOR
C    Programmer: Cary B. Skidmore
C                M.S. student in Mechanical Engineering
C          Date: November, 1984
C ***************************************************************
C  This program lets the operator observe a live picture and
C  then capture it.  The image seen by the camera is displayed
C  on the Grinnell until return is struck.  The image is then
C  captured and stored in memory.  The captured image is also
C  displayed on the Grinnell.
C
C  NOTE: The image needs 512 blocks of memory to store it.  Be
C  sure that there is enough space on your 'disk'.
C ***************************************************************

      INTEGER*2 IMAGE(512,512)
      COMMON IMAGE
      CALL IMINIT ('ERASE')

C INITIALIZATION FOR IMOPEN

      NELEM=512
      MELEM=512
      NLINE=512
      IBOTY=0
      IBOTX=0

C CALL GRINNELL SUBROUTINES

      CALL GRDOP(1,0)
      CALL GRDSH(1,0)
      CALL GRDTH(1,0,0)
      CALL GRDDG(1,2,,7)
      CALL GRSBFD

C CONTINUOUSLY DIGITIZE THE IMAGE UNTIL RETURN IS STRUCK

 1    FORMAT (A)
      TYPE *,' '
      TYPE 1,'$ STRIKE RETURN TO TAKE A PICTURE'
      ACCEPT 1, DUMMY

      TYPE *,' '
      CALL IMOPEN(1,'WRITE','IMAGE FILENAME PLEASE--> ',
     &                      'NONAME',NELEM,NLINE)
      TYPE *,' '

C FREEZE PICTURE

      CALL GRDDG (1,1,0,7)
      CALL GRSBFD
      CALL IMDISP('READ','INTEGER*2',
     &          IMAGE,NELEM,NELEM,NLINE,IBOTX,IBOTY,'WHITE')
      CALL IMTRAN(1,'WRITE','INTEGER*2',IMAGE,MELEM,NELEM,NLINE)
      CALL GRSEND
      END
```

```
C   Program name: HISTO.FOR
C     Programmer: Cary B. Skidmore
C                 M.S. student in Mechanical Engineering
C           Date: November, 1984
C ****************************************************************
C   This program computes the histogram of an image stored
C   in memory.  The histogram is displayed on the Tektronix
C   4014 display or plotted on the HP-7475 plotter depending
C   on how the operator answers the prompt.  (Note: the SELANAR
C   HiREZ monitor is Tektronix 4014 compatible.)
C ****************************************************************

      INTEGER*2  BLACK,WHITE,THRESH,IMAGE(512,512)
      PARAMETER  (BLACK=0,WHITE=255)
      INTEGER    NELEM,NLINE,I,J,HIST(BLACK:WHITE)
      NELEM=512
      NLINE=512
      MELEM=512

C  Read the image array stored in memory into IMAGE
      CALL IMOPEN(1,'READ','INPUT IMAGE FILENAME --> ','NONAME'
     &                                              ,NELEM,NLINE)
      CALL IMTRAN(1,'READ','INTEGER*2',IMAGE,MELEM,NELEM,NLINE)

C  Clear the histogram array
      DO I = BLACK, WHITE
      HIST(I) = 0
      END DO

C  Count the number of pixels at each intensity level
      DO J = 1, NLINE
            DO I = 1, NELEM
            HIST(IMAGE(I,J)) = HIST(IMAGE(I,J))+1
            END DO
      END DO

C  Call the histogram plotting subroutine.
      CALL PLOT(HIST,'GREY LEVEL','PIXEL COUNT')

      END

C ****************************************************************
      SUBROUTINE PLOT(IARRAY,XTITLE,YTITLE)
C  This subroutine makes calls to The P System of Generalized
C  Plot Routines which are documented in the User's Reference
C  Manual located in DU 268.  These routines were written by
C  the Electrical and Computer Engineering Department of Kansas
C  State University.
C
C  This program must be linked by using the following statement:
C        LINK fn,EESDYER:CPLOT]PLIB/LIB
C ****************************************************************
```

```
      INTEGER IARRAY(0:255),IGOT,I
      REAL RARRAY(0:255)/256*0./,FIRSTX,DELTAX,XARRAY(0:255)
      REAL FIRSTY,DELTAY,FIRDEL(4),DIVLNX,DIVLNY
      EQUIVALENCE (FIRDEL(1),FIRSTX),(FIRDEL(2),DELTAX),
     &            (FIRDEL(3),FIRSTY),(FIRDEL(4),DELTAY)
      CHARACTER*(*) XTITLE,YTITLE

      CHARACTER*1 ACHAR

      DO I=0,255
      RARRAY(I)=IARRAY(I)
      END DO

      DO I=0,255
      XARRAY(I)=I
      END DO

1     FORMAT (A)
      TYPE *, ' '
      TYPE 1,'$ INPUT 4014 TO DISPLAY OR 7475 TO PLOT--> '
      ACCEPT *, IDEVIC
      TYPE *, ' '
      TYPE *,' A carriage return will clear the plot from'//
     &       ' the screen.'
      TYPE *, ' '
      TYPE 1,'$ INPUT SCALE FACTOR,
     &  .7 FOR PLOT, 1. FOR DISPLAY --> '
      ACCEPT *, SCALE
      TYPE *, ' '
      TYPE 1,'$ INPUT PEN VELOCITY (1.0 TO 38.1) --> '
      ACCEPT *, VEL


      CALL PINIT(IDEVIC,' ',SCALE,'A')
      CALL PSTVEL(VEL)
      CALL PSCALE(XARRAY,256,30.,FIRSTX,DELTAX,DIVLNX)
      CALL PSCALE(RARRAY,256,20.,FIRSTY,DELTAY,DIVLNY)
      CALL PORIG(4.,4.)
      CALL PAXIS(0.,0.,XTITLE,' ',220,2201,30.,0. ,FIRSTX,DELTAX,
     &                                                 DIVLNX)
      CALL PAXIS(0.,0.,YTITLE,' ',120,1201,20.,90.,FIRSTY,DELTAY,
     &                                                 DIVLNY)
      CALL PLINE(XARRAY,RARRAY,256,FIRDEL,0,' ',DIVLNX,DIVLNY)
      CALL BELL

C  Wait for carriage return before continuing
      CALL GETUTX(1,' ',1,ACHAR,IGOT)
C  Erase screen before closing plot file
      CALL PCLRSC
      CALL PCLOSP

      RETURN
      END
```

```
C  Program name: THRESH.FOR
C     Programmer: Cary B. Skidmore
C                 M.S. student in Mechanical Engineering
C           Date: November 1984
C ****************************************************************
C  This program converts the grey scale image currently
C  being displayed by the Grinnell into a binary one.  All
C  pixels below the selected threshold level are set to a
C  grey level of 0 (black) and all others are set to 255 (white).
C  The binary image is then displayed.
C ****************************************************************

      INTEGER*2 IMAGE(512,512)
      MELEM=512
      NELEM=512
      NLINE=512
      IBOTY=0
      IBOTX=0

      CALL IMINIT ('NOERASE')
      CALL IMDISP('READ','INTEGER*2',IMAGE,MELEM,NELEM,
     &                  NLINE,IBOTX,IBOTY,'WHITE')

    1 FORMAT (A)
      TYPE 1,'$ INPUT THE THRESHOLD VALUE DESIRED --> '
      ACCEPT *,ITHRESH
      IBLACK=0

      DO J=1,512
            DO I=1,512
            IF (IMAGE(I,J).GE.ITHRESH) THEN
                  IMAGE(I,J)=255
            ELSE
                  IMAGE(I,J)=0
                  IBLACK=IBLACK+1
            END IF
            END DO
      END DO

      CALL IMDISP('WRITE','INTEGER*2',IMAGE,MELEM,NELEM,
     &                  NLINE,IBOTX,IBOTY,'WHITE')
      CALL GRSEND
      END
```

```
C  Program name: WEIGHT.FOR
C     Programmer: Cary B. Skidmore
C                 M.S. student in Mechanical Engineering
C           Date: November 1984
C ****************************************************************
C  This program is for image enhancement prior to thresholding.
C  An image stored in memory is displayed and the grey level
C  of each pixel is replaced by a weighted sum of the 3 x 3
C  array enclosing it.  The user is prompted for the weighting
C  factor.  The resulting image is displayed.
C  ****************************************************************

      PARAMETER (MELEM=512,IBOTX=0,IBOTY=0)
      INTEGER*2 IMAGE(512,512), TIMAGE(512,512)
      COMMON IMAGE, TIMAGE
      INTEGER*2 SUM(9)
      NELEM=512
      NLINE=512
      CALL IMINIT ('ERASE')

C  FIRST THE IMAGE TO BE ENHANCED IS DRAWN
      TYPE *,' '
      CALL IMOPEN(1,'READ','INPUT IMAGE FILENAME --> ','NONAME',
     &                              NELEM,NLINE)
      TYPE *,' '
      CALL IMTRAN(1,'READ','INTEGER*2',IMAGE,MELEM,NELEM,NLINE)
      CALL IMDISP('WRITE','INTEGER*2',IMAGE,MELEM,NELEM,NLINE,
     &                  IBOTX,IBOTY,'WHITE')

C THEN THE IMAGE IS ENHANCED
   1  FORMAT (A)
      TYPE 1,'$ INPUT WEIGHTING FACTOR --> '
      ACCEPT *,WEIGHT

      DO J=2,511
         DO I=2,511
         INDX=1

C  LOOP FOR 3 X 3 MATRIX

            DO K=I-1,I+1
               DO L=J-1,J+1
               SUM(INDX)= IMAGE(K,L)
               INDX=INDX+1
               END DO
            END DO

C  APPLY WEIGHTING FACTOR AND SUM VALUES

            CENTER=WEIGHT*(SUM(1)+SUM(2)+SUM(3)+SUM(4)+
     &                SUM(5)+SUM(6)+SUM(7)+SUM(8)+SUM(9))
            IF (CENTER.GT.255) CENTER=255
            TIMAGE(I,J)=CENTER
            END DO
      END DO
```

```
C THEN THE ENHANCED IMAGE IS DISPLAYED
      DO J=1,512
            DO I=1,512
                  IMAGE(I,J)=TIMAGE(I,J)
            END DO
      END DO

      CALL IMDISP('WRITE','INTEGER*2',IMAGE,MELEM,NELEM,NLINE,
     &                        IBOTX,IBOTY,'WHITE')
      CALL GRSEND
      END
```

```
C  Program name: ZOOM.FOR
C    Programmer: Cary B. Skidmore
C                M.S. student in Mechanical Engineering
C          Date: November 1984
C ***************************************************************
C  This program uses the zoom cursor and cursor control box
C  functions to read back pixel locations of chosen points.
C  The image from which points are desired must already be
C  displayed.  The selected points turn black pixels to a red
C  intensity of 255.  White pixels are not affected because
C  they already have the red plane at an intensity of 255.
C  The pixel, therefore, does not turn red but its location
C  is still read.
C
C  The ouput of the program is a sequential list of segments,
C  points within segments, and pixel location.  Pixel location
C  0,0 is at the bottom left of the monitor.
C
C  NOTE:  The program is currently dimensioned for 50 points
C         in each of 50 segments.
C ***************************************************************

      INTEGER XLOC,YLOC,ENTER, STATUS,FUNA,FUNB,I,POINT,MORE,
     &                                          CHANGE,L
      DIMENSION SEG(50),X(50,50),Y(50,50),P(50,50),MAXPNT(50)

C Initialize Grinnell and display zoom cursor
      CALL GRSINI
      CALL GRZON(1,1)
      CALL GRZWO(1,1)
      CALL GRZCO(1,1)
      CALL GRSBFD

      SEGMNT=0
      MORE=1
      L=0

20    SEGMNT=SEGMNT+1
      I=1
      POINT=1
      L=L+1

30    TYPE *,' '
      TYPE *,'CURSOR SELECT should be in the ZOOM position'
      TYPE *,'        and switch 1 in the up (ON) position.'
      TYPE *,' Position ZOOM Cursor in desired proximity.'
      TYPE *,' Enter a ZOOM factor: 1, 2, 4, or 8.'
      ACCEPT *,ITEST
      IF (ITEST.EQ.8)IVAL=3
      IF (ITEST.EQ.4)IVAL=2
      IF (ITEST.EQ.2)IVAL=1
      IF (ITEST.EQ.1)IVAL=0
      CALL GRZFC (1,IVAL)
      CALL GRSBFD
```

```
      TYPE *,' '
      TYPE *,'FUNA on, FUNB off then ENTER to select'//
     & ' a point.'
      TYPE *,'FUNA off, FUNB on then ENTER to change'//
     & ' ZOOM factor.'

      TYPE *,'FUNA and FUNB on, then ENTER to start'//
     & ' new segment.'
      TYPE *,'FUNA and FUNB off, then ENTER to exit.'
      TYPE *,' '
      TYPE *,' '
      TYPE *,'   SEGMENT          POINT'//
     & '      X COORDINATE    Y COORDINATE'
      TYPE *,' '


10    CALL GRZCW(1,STATUS)
      CALL GRZCR (1,XLOC,YLOC,ENTER,FUNA,FUNB)

      IF (ENTER.EQ.0) GOTO 10
      CHANGE= FUNA+FUNB+FUNB

      IF (CHANGE.EQ.0) THEN
          GOTO 50
      ELSE IF (CHANGE.EQ.3) THEN
          GOTO 20
      ELSE IF (CHANGE.EQ.2) THEN
          GOTO 30
      ENDIF

      CALL GRFAR(1,255,0,1,XLOC,YLOC,1,1)

      SEG(L)=SEGMNT
      X(L,I)=XLOC
      Y(L,I)=YLOC
      P(L,I)=POINT
      WRITE (*,*) SEG(L),P(L,I),X(L,I),Y(L,I)

      MAXPNT(L)=I
      POINT=POINT+1
      I=I+1
      GOTO 10

50    DO 40 N=1,L
         TYPE *,' '
         TYPE *,'   SEGMENT          POINT'//
     & '      X COORDINATE    Y COORDINATE'
         DO 45 M=1,MAXPNT(N)
         WRITE (*,*) SEG(N),P(N,M),X(N,M),Y(N,M)
45       CONTINUE
40    CONTINUE

      CALL GRZON(1,0)
      CALL GRZWO(1,0)
      CALL GRZCO(1,0)
      CALL GRSEND

      END
```

```
C  Program name:  RCOLOR.FOR
C    Programmer:  Cary B. Skidmore
C                 M.S. student in Mechanical Engineering
C          Date:  November 1984
C ************************************************************
C  This program reads all three color planes, converts them
C  to a single coded array and then stores that array.  The
C  color image being displayed must only have colors at inten-
C  sity levels of 0 or 255.
C ************************************************************

        PARAMETER  (MELEM=512,IXLOC=0,IYLOC=0)
        INTEGER*2 RED(512,512),GREEN(512,512),BLUE(512,512),
     &                  CODED(512,512)

        NLINE=512
        NELEM=512

        CALL GRSINI

C  Read all color planes
        CALL IMDISP('READ','INTEGER*2',RED,MELEM,NELEM,
     &              NLINE,IXLOC,IYLOC,'RED')
        CALL IMDISP('READ','INTEGER*2',GREEN,MELEM,NELEM,
     &              NLINE,IXLOC,IYLOC,'GREEN')
        CALL IMDISP('READ','INTEGER*2',BLUE,MELEM,NELEM,
     &              NLINE,IXLOC,IYLOC,'BLUE')
        TYPE *,' IMAGE COLOR PLANES HAVE BEEN READ'

C OBTAIN CODED ARRAY
        DO I=1,512
                DO J=1,512
                IRED=0
                IGREEN=0
                IBLUE=0
                IF (RED(I,J).EQ.255) IRED=1
                IF (GREEN(I,J).EQ.255) IGREEN=2
                IF (BLUE(I,J).EQ.255) IBLUE=3
                IVALUE=IRED+IGREEN+IBLUE
C WHITE= 1+2+3=6, BLACK=0
                CODED(I,J)=IVALUE
                END DO
        END DO

C Store coded image
        TYPE *,' '
        CALL IMOPEN(1,'WRITE','INPUT IMAGE FILENAME --> ',
     &  'NONAME',NELEM,NLINE)
        TYPE *,' '
        CALL IMTRAN(1,'WRITE','INTEGER*2',CODED,MELEM,NELEM,NLINE)
        CALL GRSEND

        END
```

```
C  Program name:  WCOLOR.FOR
C    Programmer:  Cary B. Skidmore
C                 M.S. student in Mechanical Engineering
C           Date:  November 1984
C ****************************************************************
C  This program writes an image to the Grinnell for display.  The
C  image must be one that was read using the program RCOLOR.  The
C  image array is first decoded and then written to the image
C  monitor.
C ****************************************************************

      PARAMETER (MELEM=512,IXLOC=0,IYLOC=0)
      INTEGER*2 RED(512,512), GREEN(512,512), BLUE(512,512),
     &                        CODED(512,512)
      NLINE=512
      NELEM=512
      CALL GRSINI

C  Read image stored in memory
      CALL IMOPEN (1,'READ','INPUT IMAGE FILENAME --> ',
     &          'NONAME',NELEM,NLINE)
      CALL IMTRAN(1,'READ','INTEGER*2',CODED,MELEM,NELEM,NLINE)

C  Decode array to three arrays, one for each color plane
      DO I=1,512
            DO J=1,512
            IVALUE=CODED(I,J)
            RED(I,J)=255
            GREEN(I,J)=255
            BLUE(I,J)=255
            IF (IVALUE.EQ.0) THEN
                  RED(I,J)=0
                  GREEN(I,J)=0
                  BLUE(I,J)=0
                  ENDIF
            IF (IVALUE.EQ.1) THEN
                  GREEN(I,J)=0
                  BLUE(I,J)=0
                  ENDIF
            IF (IVALUE.EQ.2) THEN
                  RED(I,J)=0
                  BLUE(I,J)=0
                  ENDIF
            IF (IVALUE.EQ.3) THEN
                  RED(I,J)=0
                  GREEN(I,J)=0
                  ENDIF
            ENDDO
      ENDDO
```

```
C  Display color image by writing each color array
C  to its respective color plane.
       CALL IMDISP('WRITE','INTEGER*2',RED,MELEM,NELEM,
     &                   NLINE,IXLOC,IYLOC,'RED')
       CALL IMDISP('WRITE','INTEGER*2',GREEN,MELEM,NELEM,
     &                   NLINE,IXLOC,IYLOC,'GREEN')
       CALL IMDISP('WRITE','INTEGER*2',BLUE,MELEM,NELEM,
     &                   NLINE,IXLOC,IYLOC,'BLUE')

       CALL GRSEND

       END
```

VITA

Cary B. Skidmore

Candidate for the Degree of

Master of Science

Thesis: DATA CAPTURE OF MULTIPLE VIEWS ON AN ENGINEERING DRAWING FOR THREE DIMENSIONAL CAD MODELING

Major Field: Mechanical Engineering

Biographical:

Personal Data: Born in Stillwater, Oklahoma, August 11, 1959, the son of Edward L. and Velma W. Skidmore. Married Wendy J. Kluber, August 13, 1981, the daughter of Edward F. Kluber, Jr. and Iris M. S. Kluber.

Education: Graduated from Manhattan Senoir High School in May, 1977; studied metallurgical engineering at the Colorado School of Mines from August, 1977 to May, 1978; received a Bachelor of Science degree from Kansas State University in May 1983, with a major in chemical engineering; completed requirements for Master of science degree at Kansas State University in December, 1984, with a major in mechanical engineering and a course emphasis in automatic controls.

Professional Experience: Worked summer of 1978 as a melter for Texas Foundries in Lufkin, Texas; worked summer of 1984 as a student engineer for Boeing Military Airplane Company in Wichita, Kansas, was in the Advanced Manufacturing Systems Research and Development group working on the REACH project (Robotically Enabled Assembly of Cables and Harnesses).

Professional Organizations: Member of American Institute of Chemical Engineers, and Society of Manufacturing Engineers Machine Vision Group; currently an Engineer in Training.

DATA CAPTURE FROM AN ENGINEERING DRAWING

by

CARY BRADFORD SKIDMORE

B.S., Kansas State University, 1983

AN ABSTRACT OF A THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Mechanical Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1985

ABSTRACT

The work reported in this thesis addresses the problem of converting two-dimensional engineering drawings on paper to three-dimensional models in a computer aided design database. The general method of solution is introduced or proposed but this thesis concentrates on the data capture phase rather than the conversion phase. A review of the literature and commercial systems available for digitizing engineering drawings is given. An engineering drawing is digitized by using a vidicon camera in conjuction with a Grinnell digitizing system. The image is converted to a suitable binary image from which data points are extracted. These points may be used to find a nonuniform, rational B-spline representation of curves and surfaces for use in the CAD database.