1988

Graphical representation of numerical data in a data base /

Dongho Lee
Lehigh University

GRAPHICAL REPRESENTATION OF NUMERICAL DATA IN A DATA BASE

by

Dongho Lee

A Thesis

Presented to the Graduate Committee

of Lehigh University

in Candidacy for the Degree of

Master of Science

in

Electrical Engineering

Lehigh University

1988

This thesis is accepted and approved in partial fulfillment of the requirement for the degree of Master of Science.

May 19, 1988
(date)

Professor in Charge

Chairman of Department

Acknowledgements

The writer would like to thank his parents, Myeongkyo and Youngran Lee, for their support and love. He also would like to express his gratitude to Prof. Donald J. Hillman for reading this paper and his guidance in his graduate study. He would like to thank Insook who encouraged him to continue writing the paper.

Abstract

In management of data bases, numerical data, which user has decided to output to the screen, cannot easily communicate with the user in raw numerical form, because the size of the data set is often beyond the capacity of the screen. In addition, the numerical data are deficient in transferring the meaning of the data. This paper describes an effective way of representing stored data by retrieving a necessary numerical data set from the data base and using a graphical representation method. This graphical representation method enables the user to understand the numerical data easily and readily.

Table of Contents

Chapter 1 Introduction 1

1.1 SRC Project 2

1.2 Paper Overview 4

Chapter 2 Graphing Data 6

2.1 The Power of Graphical Data Display 6

2.2 Graphical Perception 7

2.3 Communication 7

2.3.1 Clear Vision 8

2.3.2 Clear Understanding 10

2.3.3 Scale 11

Chapter 3 CINDAS 14

3.1 Properties and Materials Coverage . 15

3.2 Comprehensive Numerical 15

Chapter 4 Turbo C Graphics 18

4.1 Graphics System Control 18

4.2 Drawing and Filling 20

4.3 Manipulating the Screen and Viewport 20

4.4 Text Output in Graphical Mode . . 21

4.5 Color Control 22

4.6 State Query 24

Chapter 5 Geometry for 3-Dimensional Graphs 25

5.1 Plane Projections of 3 Dimensional Space 25

5.1.1 Parallel Projections 26

5.1.2 Picture Plane 31

Chapter 6 Graphic Representation Tool 34

6.1 Data Set Retrieving 34

6.2 Auto Decision of Characteristic of Data 35

6.3 Read Data Set 36

6.4 3-D Graph 37

6.4.1 3-D Surface 39

6.4.2 Side Faces 42

6.4.3 Axis and scale 44

6.4.4 String output 46

6.5 2-D graph 47

Chapter 7 Discussion 49

7.1 Summary 49

7.2 Further Work 50

Chapter 8 Bibliography 51

Chapter 9 Vita 53

List of Figures

Figure 5.1: Plane projection 26

Figure 5.2: Parallel plane projection . . . 27

Figure 5.3: Unit vector of picture plane . . 31

Figure 6.1: Border of 3-D surface 39

Figure 6.2: 3-D surface 40

Figure 6.3: Putting side faces on the 3-D surface 42

Figure 6.4: Axis and scale 44

Figure 6.5: Output string comments 46

Figure 6.6: Example of 2-D graph 47

Chapter 1

Introduction

Worldwide consumption of packaging integrated circuits in 1990 is expected to reach 66.1 billion units.

As semiconductor technology is entering the VLSI(Very Large Scale Integration) and VHSIC(Very High Speed Integrated Circuit) era, it is generally accepted that the packaging of semiconductor components may be one of the fundamental stumbling blocks to achieving desired performance. VHSIC speed and density requirements are facing new approaches to packaging. In the past, the performance for a majority of devices had been limited more by semiconductor technology than packaging technology. Increasing speed requires that the geometries must shrink. Increasing the number of circuit elements per chip increased the number of pins. As operating speeds increase, a proper package material should be chosen for the semiconductor chip to keep adequate temperature for good performance.

1.1 SRC Project

The goal of the Semiconductor Research Corporation project is to make a totally integrated system including a rule-based expert system to provide semiconductor package design. Besides providing the design, the system tests the mechanical properties of a designed package using finite element analysis and shows the shape of the package that is designed by the expert system with the support of CAD. The system also has a tool to manage a material data base which is used for selecting proper materials for the package.

This integrated system is being developed based on the IBM-AT compatible personal computer. One of the most important factors of an integrated system is the interface between the system and human user using only a screen as an output facility and a keyboard or mouse as an input facility. Once the data set of material is searched and decided to be used as a package material, the user should be able to see and understand the data easily. If an integrated system would output raw data on the screen, the user may have difficulty understanding it without special knowledge about that area, because the data set is composed of numerical data and some comments. The graphical method of drawing the data is one of the most effective display mechanisms. But existing graphic packages are not adequate for this integrated system for the following reasons. First, though commercial graphic packages have an excellent drawing functions, it makes the system more difficult in terms of space. They need a vast of memory space to run the program. Second, the integrated system needs a capability of managing the data base such as searching or skipping the data that are not necessary. However, they are not able to offer those kinds of capabilities. The new version of Turbo C can solve the above problems. It is developed based on the personal computer. One of the functions is applied to handle those types of data management.

A graphical method for representing the numerical data of the CINDAS material data base offers quick understanding of the data. In most cases, a 2-

dimensional drawing capability is enough to build the graph, because most of the data set is composed of 2 different variables (i.e., variation of thermal conductivity is measured by the variation of temperature). However, there could be a difficulty in drawing the graph with only a 2-dimensional drawing capability, because there often appear data sets which are composed of 3 different variables (i.e., one property is measured by the varying of 2 different variables). 3-dimensional graphs have more power to represent the latter case rather than using 2-dimensional graphs.

1.2 Paper Overview

This paper is focused on the capabilities of a graphical method to represent numerical data which are stored in data base. Chapter 2 is a description of the general theory of graphing data. Chapter 3 is description of the CINDAS data base in terms of what kind of data were developed and built into the data base. Chapter 4 is a description of graphic capabilities of Turbo C new version. Chapter 5 is a description for the geometries to make 3-D graphs and 2-D graphs.

Chapter 6 is a description of a graphical representation tool that makes a data set, which the integrated system can use, and draws 3-D graphs and 2-D graphs.

Chapter 2

Graphing Data

2.1 The Power of Graphical Data Display

In the past, the number of values that could be put on a graph was limited by the graph having to be made by hand. Computer graphics has removed these restrictions. Now the number of values is limited only by the resolution of graphics devices and the perceptual ability of our visual system. The role of graphical data display will lead to a deeper understanding of the data that arise in scientific and technological studies. Graphs are exceptionally powerful tools for data analysis. Graphical methods tend to show data sets as a whole, and it allows us to summarize the general behavior. One reason why graphical displays can retain the information in the data is that a large amount of quantitative information can be displayed and absorbed.

2.2 Graphical Perception

When a graph is constructed, quantitative and categorical information is encoded by symbols, geometry, and color. Data from graphs are decoded visually, and a very complex set of perceptual and cognitive tasks are carried out. Graphical perception is the visual decoding of this encoded information. Graphical perception is the vital linking the graph. No matter how intelligent the choice of information, no matter how ingenious the encoding of the information, and no matter how technologically impressive the production, a graph is a failure if the visual decoding fails. To have a scientific basis for graphing data, graphical perception must be understood. Informed decisions about how to encode data must be based on knowledge of the visual decoding process.

2.3 Communication

This section will describe the important factors to communicate the contents of graph to others without distortion.

2.3.1 Clear Vision

Clear vision is a vital aspect of graphs. The viewer must be able to visually understand the many different items that appears on a graph. The data (the quantitative and qualitative information in the data region) are the reason for the existence of the graph. The data should stand out. It is easy to forget this. There are many ways to obscure the data, such as allowing others elements of the graph to interfere with the data or not making the graphical elements encoding the data visually prominent. Data are frequently obscured by graphing them on top of scale lines. Sometimes different values of the data can obscure each other. Superfluity should be eliminated in graphs. Unnecessary parts of a graph add to the clutter and increase the difficulty of making the necessary elements -the data- stand out. A good way to help the data to stand out is to show them with a graphical element that is visually prominent.

Another way to obscure data is to graph too much. It is always tempting to show everything that comes to mind on a single graph, but graphing too much can result in less being seen and understood. A large number of tick mark is usually superfluous. From 3 to 10 tick marks are generally sufficient. This is just enough to give a broad sense of the measurement scale. Now, there are photocopies of tables, computer tapes, disk packs, and telecommunications networks to transfer data. Every aspect of a graph should serve an important purpose. Any superfluous aspects, such as unneeded tick marks, should be eliminated to decrease visual clutter and increase the visual prominent of the most important element.

Data labels are not allowed in the data region. It may interfere with the quantitative data or clutter the graph. There is no reason why markers, keys, and notes need to appear in the data region. These things can go outside the data region.

Unless special care is taken, overlapping plotting symbols can make it impossible to distinguish individual data points. It is very common for graphs to have two or more data sets superposed within the same data region.

2.3.2 Clear Understanding

Graphs are powerful tools for communicating quantitative information in technical report and journal articles. Communication of the results of scientific and technological studies can be greatly enhanced by graphs that speak to the essence of the results, when the results involved quantitative issues. Graphs and their legends can incisively communicate important data and important conclusions drawn from the data. One good approach is to make the sequence of graphs and their legends as nearly independent as possible and to have them summarize evidence and conclusions.

For a graph to be understand clearly, there must be a clear, direct explanation of the data that are graphed and of the inferences drawn from the data. There is a framework for figure legends that can contribute to such a clear explanation;

1. Describe everything that is graphed.
2. Draw attention to the important features of the data.

3. Describe the conclusions that are drawn from the data on the graph.

When the logarithms of the data are graphed, that is equal increments on the horizontal scale indicate equal increments of the logarithm of the data. On the scale line the tick mark labels show the values of the data on the original scale. The scale label describes the various and its units on the original scale, to correspond to the tick mark labels.

2.3.3 Scale

Scales are fundamental. Graphing data would be far simple if these basic, defining elements of graphs were straightforward, but they are not simple. Scale issues are difficult and subtle. This section is about constructing scale lines, including zero, and taking logarithms.

The interval from the minimum to the maximum of a set of values is the range of the values. It is a good idea to have the range of the data on a graph be included or nearly included in the range of the tick marks to allow an effective assessment of all of the data.

There are a number of constraints that affect the choice of scales on graphs. One is that the range of the tick marks should encompass or nearly encompass the range of data. Another is that it is not desirable for data to be graphed on scale lines. Also, in some cases a particular value is included in the scale. Finally, when different panels of a graph are compared, it will be necessary for the scale to be the same on all panels.

When the data are magnitudes, it is helpful to have zero included in the scale so it allows us to see the value relative to the value of the data. But the need for zero is not so compelling that we should allow its inclusion to ruin the resolution of the data on the graph.

There has been much polemical writing about including zero when graphs are used to communicate quantitative information to others. Too frequently zero has been endowed with an importance it does not have.

It is common for positive data to be skewed to the right. Some values bunch together at the low end of the scale and others trail off to the high end with increasing gaps between the values as they get together. Such data can cause severe resolution problems on graphs, and the common remedy is to take logarithms. Indeed, it is the frequent success of this remedy that partly accounts for the large use of logarithms in graphical data display.

Chapter 3

CINDAS

The Center for Information and Numerical Data Analysis and Synthesis(CINDAS) of Pursue University conducts a comprehensive systematic program on the properties of materials. The program involves the in-depth cognizance and acquisition of the relevant worldwide scientific and technical literature, the exhaustive extraction and compilation of experimental data from the acquired pertinent research documents, the critical evolution, analysis, correlation, and synthesis of the compiled experimental data to generate reliable reference data, the generation of estimated values to fill data page and voids, the nationwide dissemination of resulting data and information through publications and through user inquiry services. The computerization of both the selected experimental data and the reliable reference data to establish computerized comprehensive numerical/technical data bases for on-line access and instant retrieval and dissemination of data and information through computer terminals across the nation.

3.1 Properties and Materials Coverage

The data system contains a number of data bases; notably the thermoplastic and electronic properties data base on engineering materials, the data base on dielectric materials, the thermoplastic, optical, and mechanical properties data base on aerospace structural composites and metals and detector/sensor materials, the thermoplastic and mechanical properties data base on rocks and minerals, and several other smaller data bases such as the data base on fluids.

3.2 Comprehensive Numerical

CINDAS has maintained a comprehensive and authoritative numerical/technical data system on materials properties which contains over 125,000 sets of data, and has been recognized nationally and indeed internationally as the largest single data source and resource on materials properties. The numerical/technical data system contains a number of data bases which are similar to those in the bibliographic/literature data system except that these data bases contain the actual numerical data and technical information on the properties of selected materials. Both the selected experimental data and the CINDAS-generated reliable reference data are included in data bases.

Since, numerical property data without the accompanying adequate information on the test material and on the property measurement are not useful, CINDAS has always paid special attention to extracting such information from the research documents together with the numerical data. Thus, each set of data extracted and compiled by CINDAS consists of numerical data points (as a function of temperature and/or other independent variable) and pertinent information on the property measurement, such as composition, purity, density, porosity, micro structure, material construction configuration, material processing, sample preparation, specimen geometry and dimensions, material history, heat treatment, cold working, surface condition, producer, supplier, method of measurement, test environment, heat flow direction, heating rate, heat-up time, heat-up temperature, holding time at temperature, type of heat source, loading rate, cooling rate, etc., insofar as these are contained in the original document.

CINDAS has been performing critical evaluation, analysis, correlation, and synthesis of compiled experimental data on selected materials to generated reliable reference data, most of the evaluated reliable data generated by CINDAS over the years have been recognized as national and international standard reference data.

Chapter 4

Turbo C Graphics

Turbo C version 1.5 provides a separate library of over 70 graphics functions from high level calls to bit oriented functions. The graphics library supports numerous fill and line styles, and provides several text fonts that can be magnified, justified, and oriented horizontally or vertically. These functions are in the library GRPHICS.LIB, and they are prototype in the header file GRAPHICS.H. In addition to these two files, the graphics package includes graphics device drivers and stroked character fonts.

4.1 Graphics System Control

Turbo C´s graphics package provides graphics drivers for the following graphics adapters; Color Graphics Adapter(CGA), Multi Color Graphics Array(MCGA), Enhanced Graphics Adapter(EGA), Video Graphics Array(VGA), Hercules Graphics Adapter, AT&T 400 line Graphics Adapter, 3270 PC Graphics Adapter.

In order to start the Graphics system, the initgraph function must be called. initgraph loads the graphics driver and puts the system into graphics mode. initgraph can be told to use a particular graphics driver and mode, or to auto detect the attached video adapter at run time and pick the corresponding driver. If initgraph is told to auto detect, it calls detectors to select a graphics driver and mode. Normally, the initgraph routine loads a graphics driver by allocating memory for the driver, then loading the appropriate .BGI file from disk. As an alternative to this dynamic loading scheme, a graphics driver file can be linked directly into an executable program file.

During run time, the graphics system might need to allocate memory for drivers, fonts, and an internal buffer. If this is necessary, it calls graphites to allocate memory, and calls graphfreemem to free it. Finally, close graph is called to shut down the graphics system. close graph unloads the driver from memory and restores the original video mode.

4.2 Drawing and Filling

Turbo C´s drawing and painting functions can draw colored lines, arcs, circles, ellipses, rectangles, pie slices, 2-dimensional and 3-dimensional bars, polygons, and regular or irregular shapes or a combination of these. They can fill any bounded shape (or any region surrounding such a shape) with one of 11 pre defined patterns, or a user-defined pattern. The thickness and style of the drawing line also can be controlled. Lines and unfilled shapes are drawn with the functions arc, circle, drawers, line, liner el, line to, and rectangle. These shapes are filled with floozies, or combined drawing/filling into one step with bar, bar3d, fillpoly, and pieslice. The setlinestyle allows user to specify whether the drawing line is thick or thin, and whether its style is solid, dotted, etc., or some other line pattern defined by user.

4.3 Manipulating the Screen and Viewport

Besides drawing and painting, the graphics library offers several functions for manipulating the screen, viewports, image, and pixels. The function cleardevice is used to clear the whole screen at once; this routine erases the entire screen and homes the current position (CP) in the viewport, but leaves all other graphics system settings intact.

Depending on the graphics adapter, the system has between one and eight screen page buffers, which are areas in memory where individual whole-screen images are stored dot-by-dot. setactivepage specify which screen page is the active one, where graphics functions place their output, and setvisualpage specifies the visual page.

Once the screen is in a graphics mode, viewport (a rectangular "visual screen") can be defined by calling setviewport. The coordinates for all output functions (drawing, filling, text, etc.) are viewport relative.

4.4 Text Output in Graphical Mode

The graphics library includes an 8X8 bit-mapped font and several stroked fonts for text output while in graphics mode.

-In a bit-mapped font, each character is defined by a matrix of pixels.

-In a stroked font, each character is defined by a series of vectors that tell the graphics system how to draw that character. Since a stroked font is defined by vectors, it will still retain good resolution and quality when the font is enlarged. When a bit-,mapped font is multiplied by a scaling factor, the characters´ resolution becomes coarser. For small characters, the bit-mapped font should be sufficient. Graphics text is output by calling either <u>outtext</u> or <u>outtextxy</u> and <u>settextjustify</u> control the justification of the output text.

Character font, direction, and size can be selected by using <u>settextstyle</u>. <u>setusercharsize</u> allows you to modify the character width and height of stroked fonts.

4.5 Color Control

The graphics screen consists of an array of pixels, each pixel produces a single dot on the screen. The pixel´s value does not specify the precise color directly; it is an index into a color table called a <u>palette</u>. The <u>palette</u> entry corresponding to a given pixel value contains the exact color information for that pixel. This indirection scheme has a number of implications. Though the hardware might be capable of displaying many colors, only given time. The number of colors that can be displayed at any one time is equal to the number of entries in the palette (the palette´s size). For example, on an EGA, the hardware can display 64 different colors, but only 16 of them at a time; the EGA palette´s size=16.

The size of the palette determines the range of values a pixel can assume, from 0 to (size - 1). The getmaxcolor function returns the highest valid pixel value (size - 1) for the current graphics driver and mode. The term color, such as the current drawing color, fill color, and pixel color, is like a pixel value which index into the palette. Only the palette determines the true color on the screen.

The background color always corresponds to pixel value 0. When an area is cleared to the background color, that area´s pixels set to 0. The drawing color is the value to which pixels are set when lines are drawn. A drawing color is selected with setcolor(n), when n is a valid pixel value for the current palette.

4.6 State Query

In each of Turbo C´s graphics functions categories, there is at least one state-query function. These functions are mentioned under their respective categories and also covered here. Each of the Turbo C graphics state-query functions is named get<something>. Some of them take no argument and return a single value representing the requested information. Others take a pointer to a structure defined in GRAHICS.H, fill that structure with the appropriate information, and return no value.

Chapter 5

Geometry for 3-Dimensional Graphs

5.1 Plane Projections of 3 Dimensional Space

In producing drawings of three dimensional objects, whether on permanent media or graphical display tubes, two dimensional projections are required. In a plane projection, each object point is projected in a defined manner onto the picture plane, where it is represented by a picture point. If the projection lines joining corresponding object and picture points are all parallel, then we have a plane parallel projection, as shown in Figure 5.1. On the other hand, if the projection lines converge on a common point P, the picture obtained is a central projection or perspective view of the object.

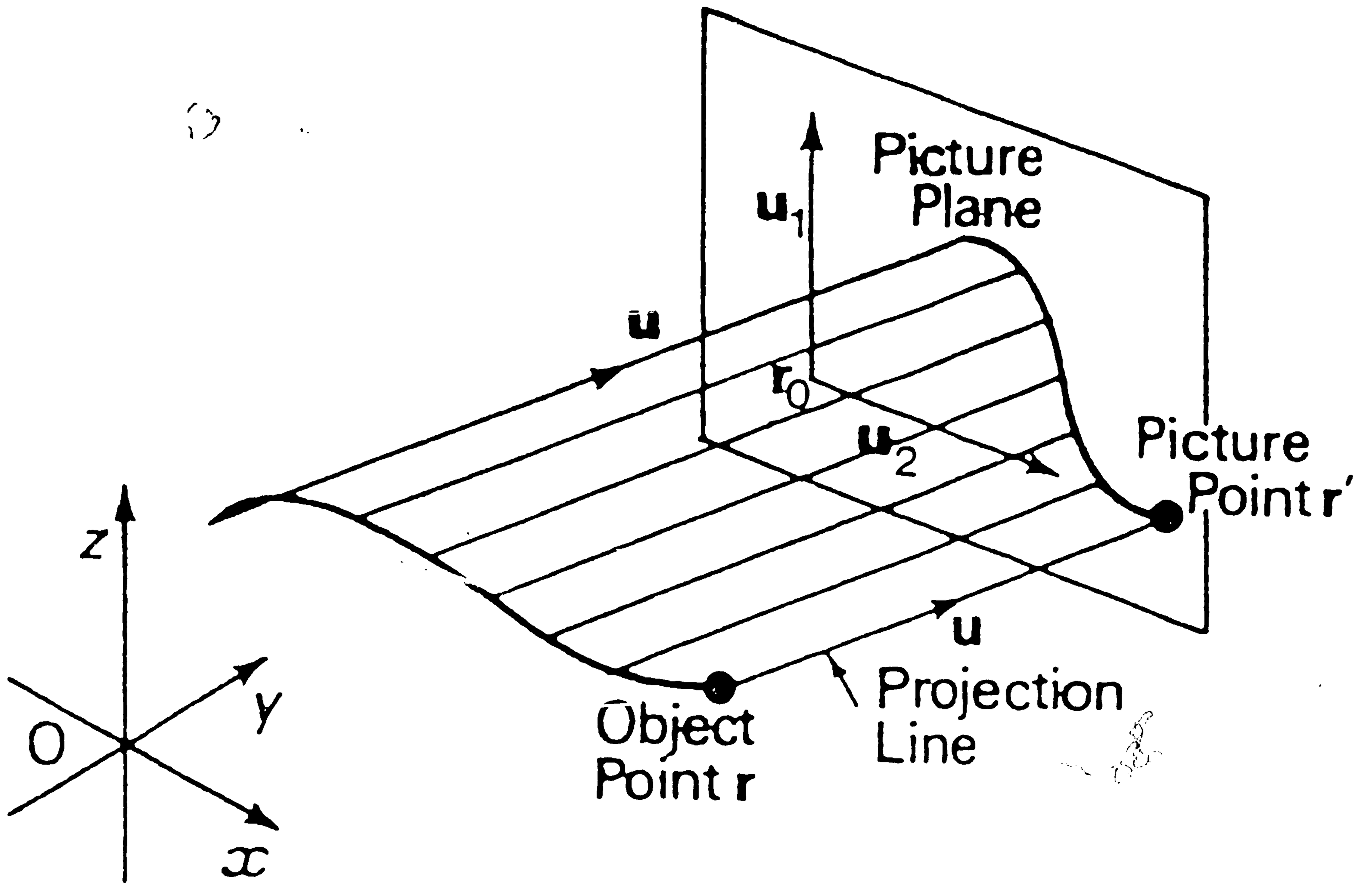

Figure 5.1: Plane projection

But this thesis used the plane parallel projection method to produce a 3-dimensional graph, so the next section will describe parallel projection.

5.1.1 Parallel Projections

In a plane parallel projection, all object points are parallel to some fixed direction u onto a specified picture plane (see Figure 5.1). We describe a point in the picture plane in terms of some prescribed plane coordinate system. The picture plane and its coordinate system may be specified by giving its origin $\mathbf{r}_0$ and the direction $\mathbf{u}_1$ and $\mathbf{u}_2$ of its axes in terms of the underlying three dimensional coordinate system.

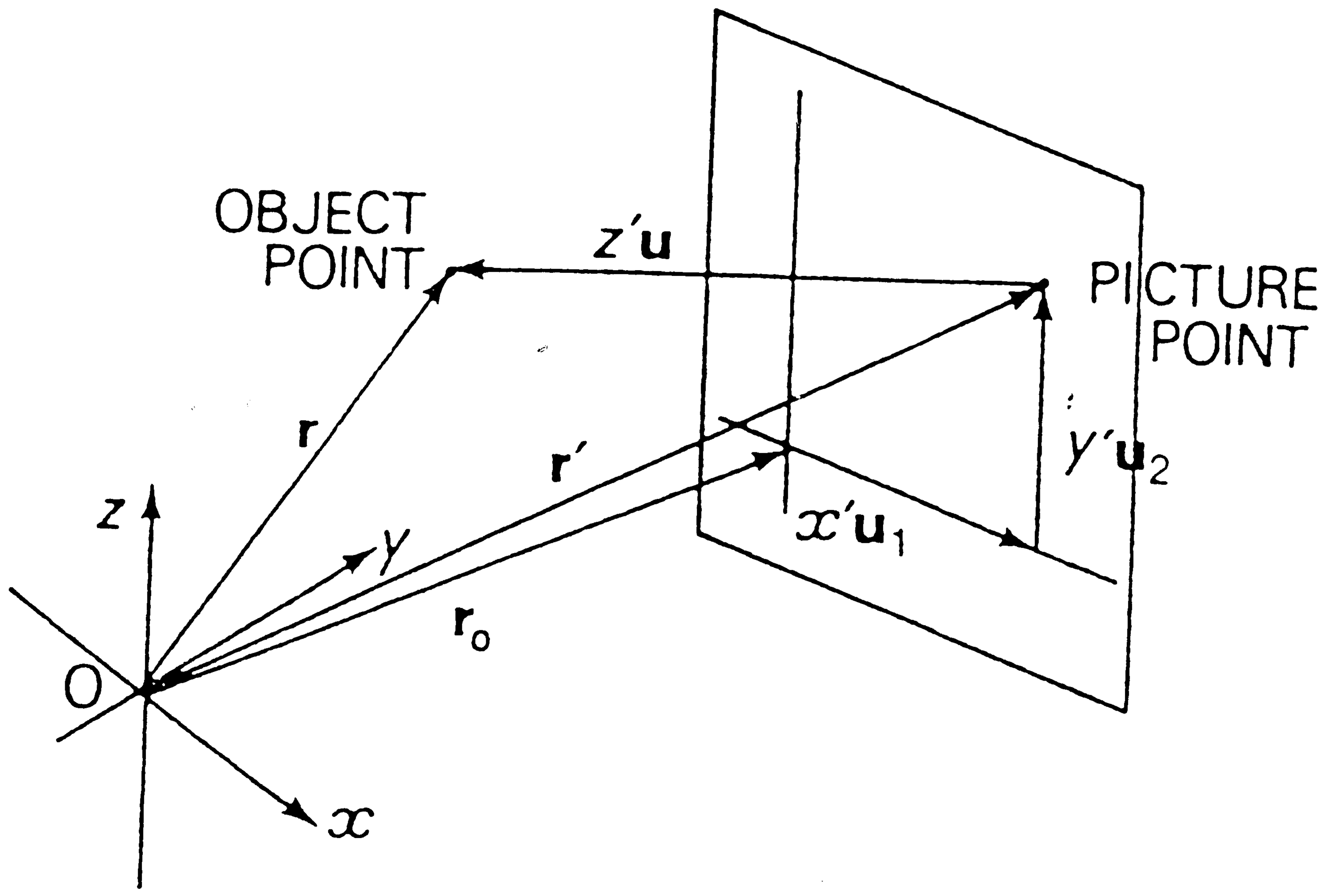

Figure 5.2: Parallel plane projection

Since each object point $\mathbf{r}$ is projected parallel to $\mathbf{u}$ onto the image point $\mathbf{r}'$, we notice from Figure 5.1 that

$$\mathbf{r}' = \mathbf{r} - z'\mathbf{u}$$

for some value of z'. If the coordinates of the image point in the projection plane are given by (x', y') then Figure 5.2 shows that

$$\mathbf{r}' = \mathbf{r}_0 + x'\mathbf{u}_1 + y'\mathbf{u}_2 ,$$

and hence $$\mathbf{r}' = \mathbf{r} - z'\mathbf{u} = \mathbf{r}_0 + x'\mathbf{u}_1 + y'\mathbf{u}_2 . \qquad (5.1)$$

(Note that $Ox'y'z'$ are not necessarily orthogonal axes.)

Now z' may be obtained by taking the scalar product of equation (5.1) with $\mathbf{u}_1 \times \mathbf{u}_2$ to eliminate x' and y'. Then

$$z' = \frac{(\mathbf{r} - \mathbf{r}_0)(\mathbf{u}_1 \times \mathbf{u}_2)}{\mathbf{u}(\mathbf{u}_1 \times \mathbf{u}_2)}$$

By taking scalar products of the same equation with $\mathbf{u}_2 \times \mathbf{u}$ and $\mathbf{u}_1 \times \mathbf{u}$ respectively, expression for x' and y' can be similarly obtained.

Thus $$x' = \frac{(\mathbf{r} - \mathbf{r}_0)(\mathbf{u}_2 \times \mathbf{u})}{\mathbf{u}(\mathbf{u}_1 \times \mathbf{u}_1)}$$

and $$y' = \frac{(\mathbf{r} - \mathbf{r}_0)\cdot(\mathbf{u}_1 \times \mathbf{u})}{\mathbf{u}\,(\mathbf{u}_2 \times \mathbf{u}_1)} \qquad (5.2)$$

In most case the picture plane is chosen perpendicular to the projection lines, so that $\mathbf{u} = \mathbf{u}_1 \times \mathbf{u}_2$, and the equations take the simpler form

$$x' = (\mathbf{r} - \mathbf{r}_0)\,\mathbf{u}_1,$$

$$y' = (\mathbf{r} - \mathbf{r}_0)\,\mathbf{u}_2,$$

and $$z' = (\mathbf{r} - \mathbf{r}_0)\,\mathbf{u}. \qquad (5.3)$$

The information about z' is not needed in order to draw the projection of the object, but it does enables us to reconstruct the object point if we wish to do so, using equation(5.1). The size of z' represent the distance from picture plane to the object point. It can be used to determine whether the point is covered with other point or not as an application.

Using matrix notation, we may write the equation (5.3) in the form

$$R' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} & u_1^T & & 0 \\ & u_2^T & & 0 \\ & u_3^T & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -x \\ 0 & 1 & 0 & -y \\ 0 & 0 & 1 & -z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{5.4}$$

so that projection is equivalent to translating the picture plane and object until the picture plane passes through the origin (matrix T), and then rotating them (matrix A) until the picture axes coincide with the Ox and Oy axes. The transformation can, of course, be written in terms of homogeneous coordinates.

Thus orthogonal plane parallel projections can be carried out using equivalent object transformations, and the object point can be recovered by the inverse transformations.

5.1.2 Picture Plane

To find out the unit vector of the picture plane, the picture plane is supposed to be in x, y, and z cartesian coordinates. The unit vectors of the picture plane are the basis for deciding the angle of eye point. We used these two unit vectors as a function of two angle values which are α between the picture plane and the yz plane and β between the picture plane and the xy plane.

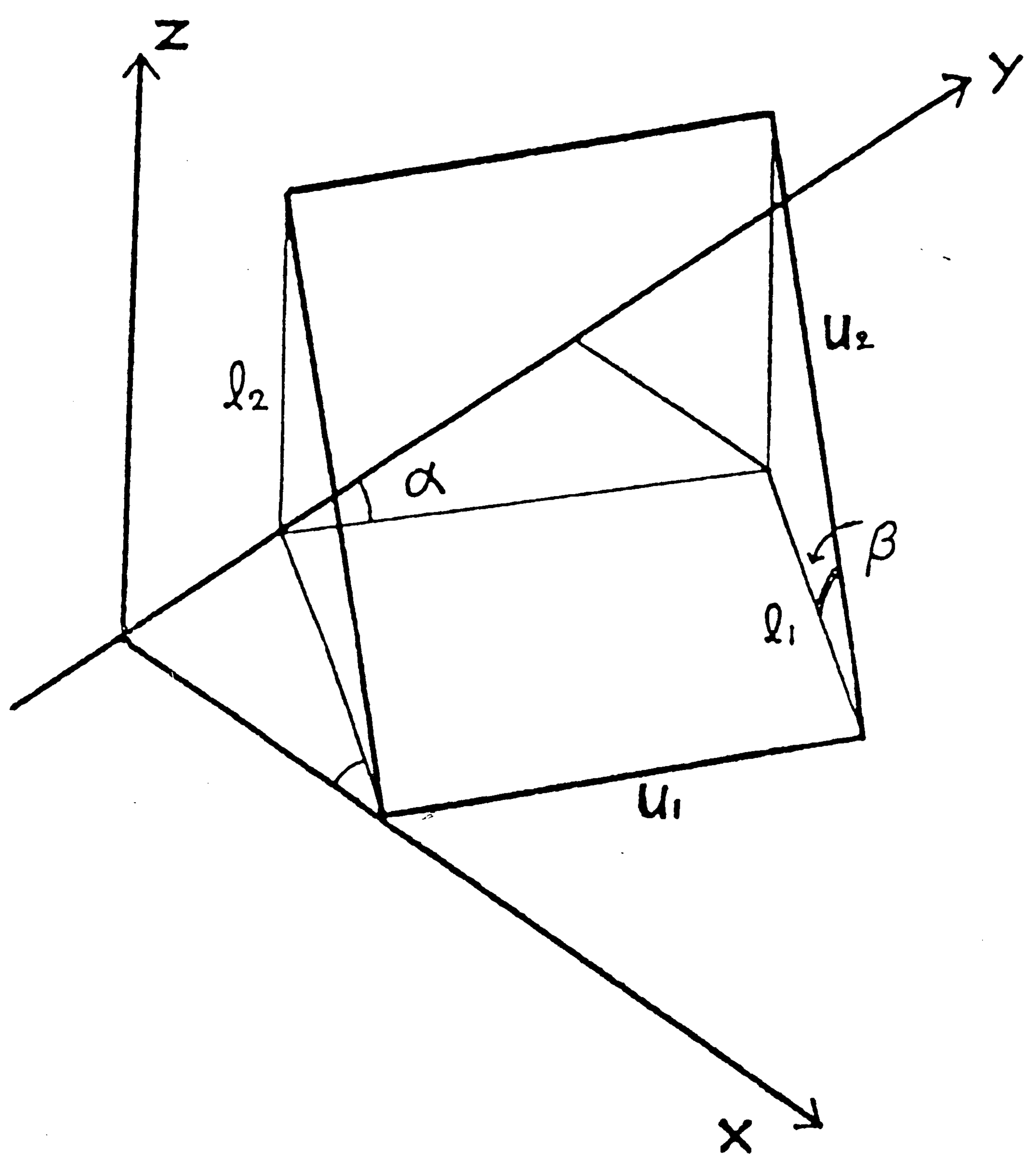

Figure 5.3: Unit vector of picture plane

The unit vector u_1 is on the xy plane. So, u_1 is composed of x and y direction components, and the x and y components of vector u_1 are,

$$|u_1| \sin\alpha = \sin\alpha, \text{ (because } |u_1| = 1)$$

and $$|u_1| \cos\alpha = \cos\alpha,$$

In order to find cartesian components of a vector u_2, the size of $\underline{l_1}$ and $\underline{l_2}$ should be found first.

$$\underline{l_1} = |u_2| \cos\beta = \cos\beta,$$

$$\underline{l_2} = |u_2| \sin\beta = \sin\beta.$$

and x, y, and z components of u_2 are,

$$-\underline{l}_1 \cos\alpha = -\cos\beta\cos\alpha,$$

$$\underline{l}_2 \sin\alpha = \cos\beta\sin\alpha,$$

and $$\underline{l}_2 = \sin\beta.$$

Thus, $u_1 = \sin\alpha\,\underline{i} + \cos\alpha\,\underline{j}$,

$$u_2 = -\cos\beta\cos\alpha\,\underline{i} + \cos\beta\sin\alpha\,\underline{j} + \sin\beta\,\underline{k}.$$

And the normal vector of picture plane u is able to be found by the vector product of u_1 and u_2.

$$u = u_1 \times u_2$$

$$= \cos\alpha\sin\beta\,\underline{i} - \sin\alpha\sin\beta\,\underline{j} + \cos\beta\,\underline{k}.$$

Plane projection is the process of finding local coordinate of projected image on this picture plane. The shape of 3 dimensional object which is appeared on picture plane is able to be changed simply by varying the angle α and β.

Chapter 6

Graphic Representation Tool

This chapter will describe graphical representation by using one example. Graphical representation tool can be broadly divided into two parts: retrieving data sets from the data base and making a graph according to the characteristics of the data set which is previously retrieved.

6.1 Data Set Retrieving

The CINDAS data file is composed of small consecutive data sets which consist of the name of the material, kind of property, methods of measuring and developing, and numerical data. Each data set begins with the name of the material and the kind of property. So, these two words are used to distinguish one data set from the data base. By comparing these two words with the given words, the beginning position of the data set, which is required to draw the graph, is able to be found. To draw the graph of CINDAS data on the PC´s screen, all of the contents of the data set are not required, but only the numerical data and some comments, which can explain the meaning of graph, are needed. Only necessary contents are chosen and stored with a specified file name that will be used when the data set is read later in the read procedure. When the contents are stored, the format of it should be matched with the format of reading data later.

6.2 Auto Decision of Characteristic of Data

Numerical data in the CINDAS database are composed of 2 different values x and y, or 3 different variables x, y, an z, or even 4 different variables x, y, z1, and z2. But the graphical representation of most CINDAS data sets is satisfied with 2-dimensional and 3-dimensional graphic capabilities. This graphical representation tool offers both capabilities as a function of the C programming language, so that it should be able to decide which representation method is appropriate to the source data set.

The deciding routine is made relatively simple, because the format of data which is used for drawing a graph is reformed when the data set is read out from the data base. The choice of graphing method, whether 2-D or 3-D, is justified by testing only x variables. If x variables of the data set are composed of non-repeatable values, then the data set can use a 2-D graph capability. Or if x variables consist of numbers appearing periodically, then the data set requires a 3-D graph capability. The number of variables in a data set does not influence the choice of a graphing method. Though the data set has 4 different data variables, if x variables of the data set are composed of non-repeatable data, the graph of the data could be drawn with a 2-D graph function. In this case, only 2 variables are needed to graph, thus the rest of the data are disregarded.

6.3 Read Data Set

The reformed data set is composed of several string lines and numerical data. When string lines are read, the reading procedure is not changed regardless 2-D or 3-D graph. But numerical data are affected by the choice of data characteristics, so that if the data set is for a 3-D graph, the 3 variables x, y, and z should be kept. If there are more than these 3 variables, they are treated as dummy variables when the graph is drawn. In addition to the numerical data , string data also play an important role when the graph is displayed. These data are composed of the name of the material, the kind of property and measured properties (i.e., X : temperature Y : strain rate etc.). String variables are stored in the program with the string of character type line by line.

6.4 3-D Graph

Data for 3-D graphs are composed of many y values per one x value and many x values per one y value. Thus, when a y direction line is drawn consecutive x points at fixed y values are necessary, and making an x-direction line requires consecutive y points at fixed x values. A 3-D surface is made by gathering these x-direction lines and y-direction lines.

When a 3-D graph routine reads data from a fixed data file, the data are used for constructing the x-direction line, and the routine makes another copy of the data for the y-direction line. The data set for the x-direction line are sorted by the y value and data for the y-direction line are sorted by x value. Therefore, the data set for x-direction lines is sorted from the line with low x value to the line with high x value. The data of each line are sorted by y values in ascending order. These data sets have world coordinate values. However, the differences in data value between axes are so large that coordinate values for each axis are revalued with the value ranged from 10 to 90 with a same proportion to world coordinate to reduce the error which may occur on the performing computation of plane projection. Then, this routine finds out these data sets; the data set of the four edge lines of the surface, the data set for side faces, the data set for axes, and the data set for the scale marker. All data sets are taken plane projection. As a result of plane projection, they produce the local coordinate values on the picture plane. Then the value of each variable is revalued to be in the drawing space on the screen.

6.4.1 3-D Surface

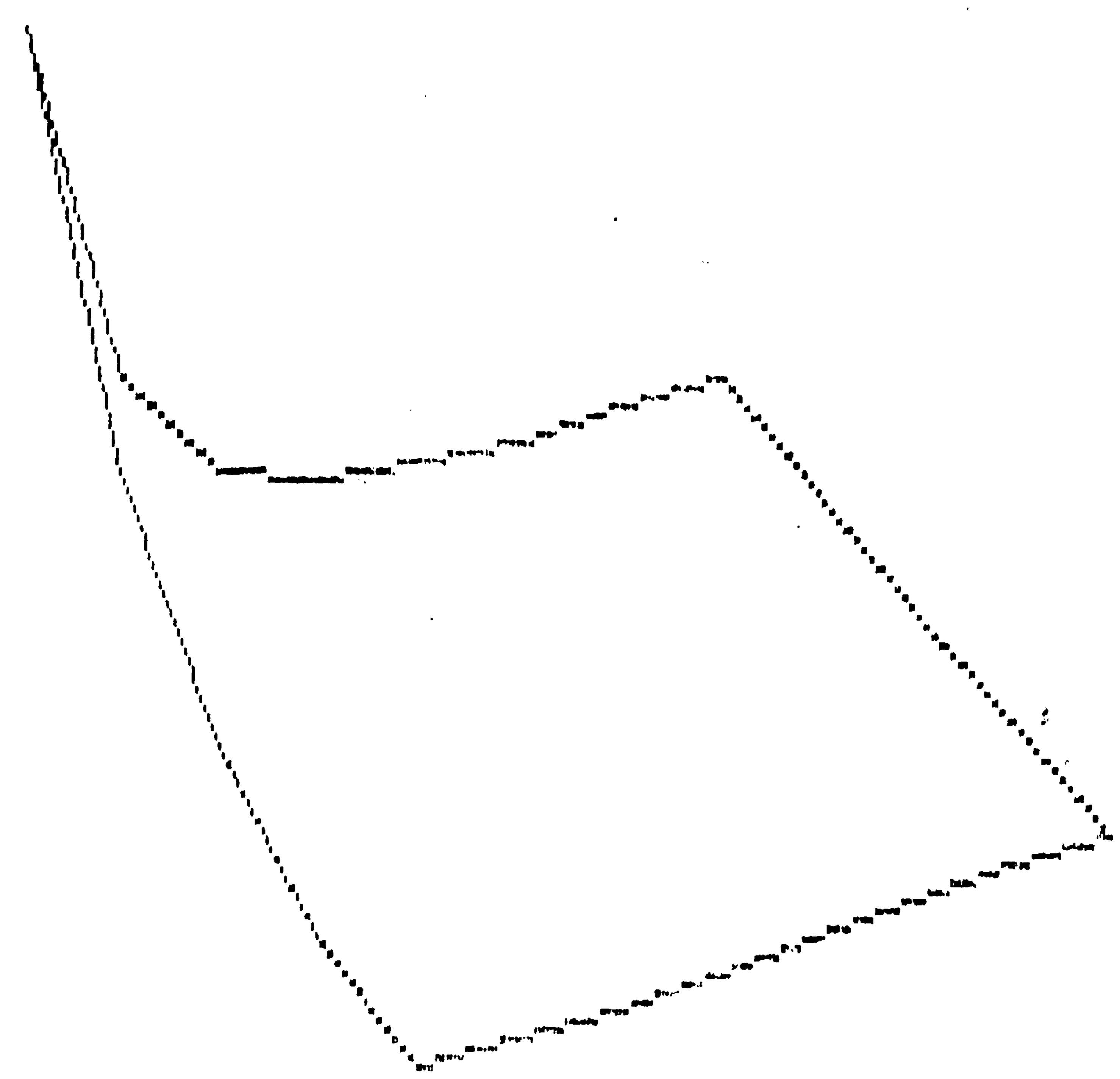

Figure 6.1: Border of 3-D surface

To construct a 3-D surface, first of all four edge lines are found. Each edge line is composed of many points which are decided by x and y axis value; that is, one combination of x and y values from maximum x value, minimum x value, maximum y value, and minimum y value. These data sets also must be taken plane projection and revalued for the screen drawing. And then these data sets are output and painted on the screen using <u>fillpoly</u> function in Turbo C (see Fig 6.1). These lines are then made on the surface. The data set of x direction lines is made by sorting y values. One x direction line data is composed of ascending order of y variable with same x values. The x direction data are a set of these line data.

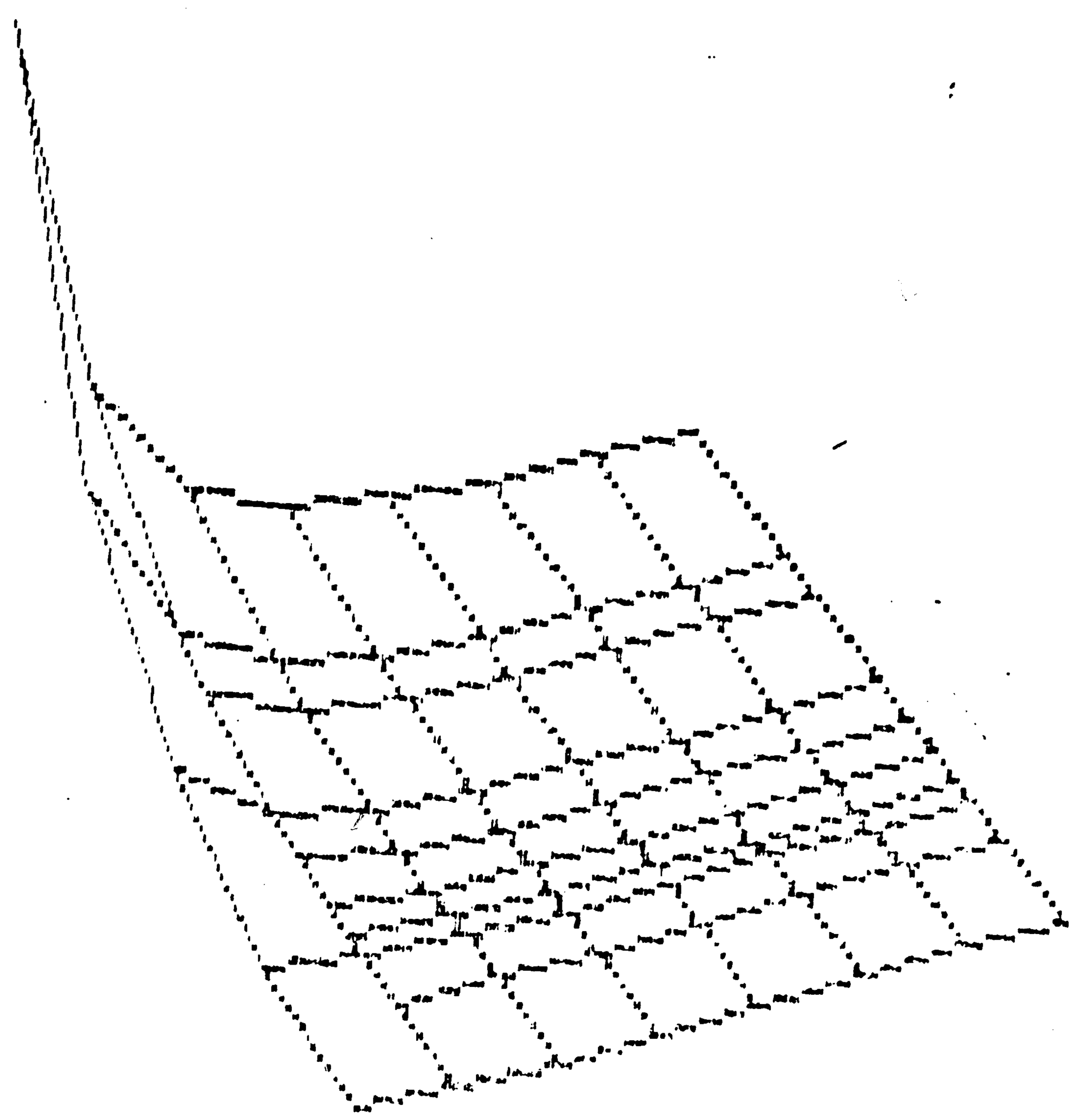

Figure 6.2: 3-D surface

The drawing procedure is consecutive connection of adjacent two points. If the y value of the next point is larger than that of the current point, two points are connected with specified line style, line width, and color. Otherwise, the current position on screen is moved to the next point without connection. The same process is done on the y direction data set (see Fig 6.2).

6.4.2 Side Faces

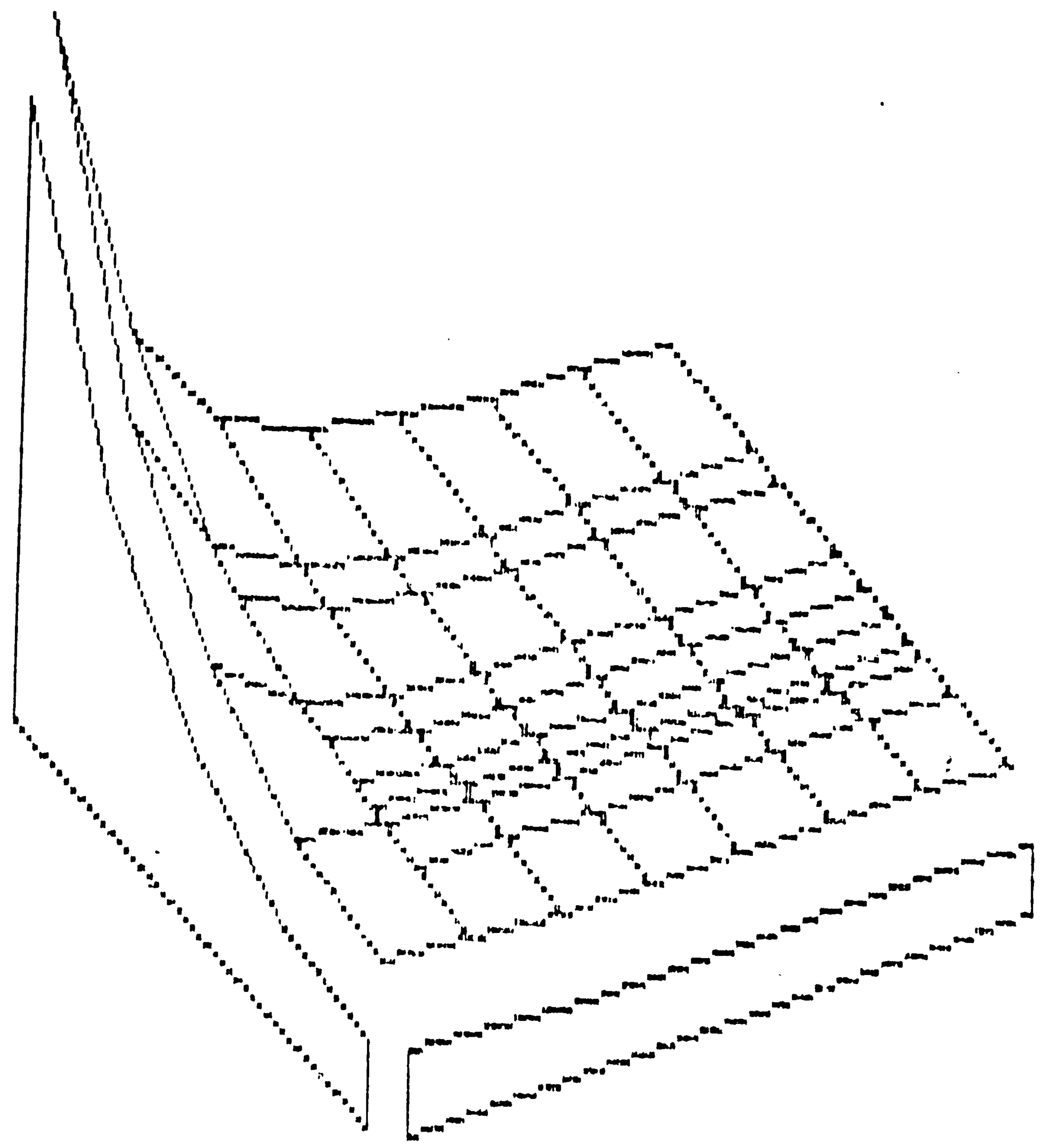

Figure 6.3: Putting side faces on the 3-D surface

If the 3-D graph which is built with only a 3-D surface does not offer a 3-D effect to the user, side faces are used to make an effective 3-D graph. If this 3-D graph is built in the real world, there are 4 side faces. But only 2 side faces, at most, are seen in the projected graph because the other 2 side faces are occluded by the 3-D surface or other side faces. This presentation tool can output various shapes of one graph by changing the angle of the picture plane. If it is derived to do so, 4 different side face data sets are made and hidden surfaces are removed when it is drawn. But changing the shape of the graph, such as rotation, is not needed. The role of the graph can be performed only with a fixed angle of the picture plane (Figure 6.3).

6.4.3 Axis and scale

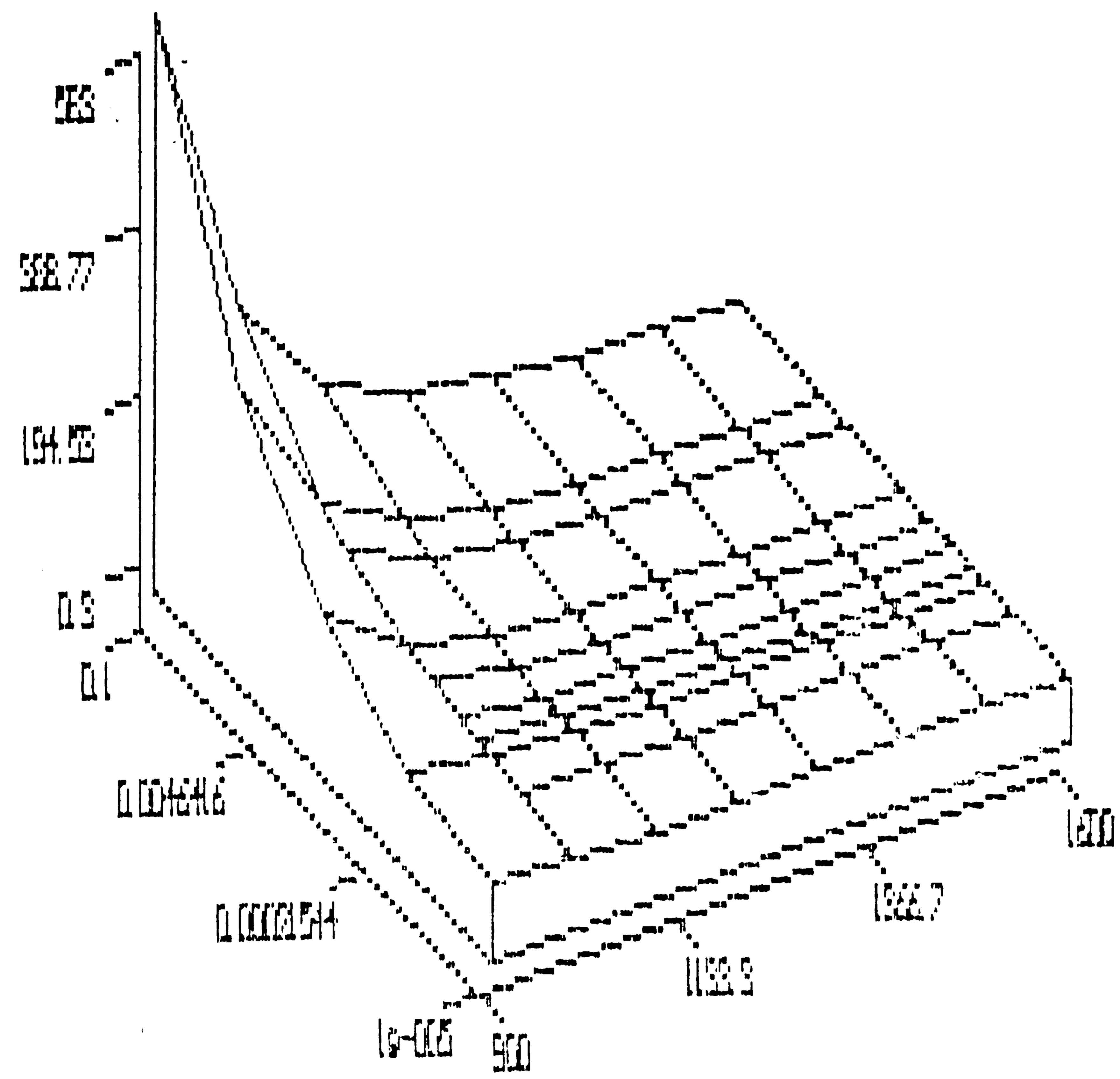

Figure 6.4: Axis and scale

Scale and axis are essential elements of the graph and represent the ruler along the graphed data in graphical data display. The data set for the axis is made from the data of side faces. It is not always true that the axis should be located at zero position. For scales to possess the zero value is not always necessary. The reason for graphing the quantitative information is to communicate quickly the meaning of data. If the meaning of the graph can be delivered better without possessing a zero point, the axis should not have a zero point. Though easily understood numbers (i.e., 10, 15, 100, 200) are desirable as a scale mark label, this graphical representation tool does not offer such scale mark numbers because the range of data is not limited and scale mark labels are decided automatically to refer to the maximum and minimum number of input data. Too many scale marks and scale mark labels confuse the user. The number of scale mark labels is limited 4 on each axis in this program for the reason of space to output and effective expression of scale. When the scale label is output on the screen, the color of it is matched with the color of the face which is made corresponding to that axis. In a 3-D graph, in order to give a real effect, the data of all axis and scale marks have been supposed to exist in world coordinates and then plane projection is taken to be seen as tilted as the same degree of orthogonal axis (see Figure 6.4, the scales are made by example of the one property of silicon).

6.4.4 String output

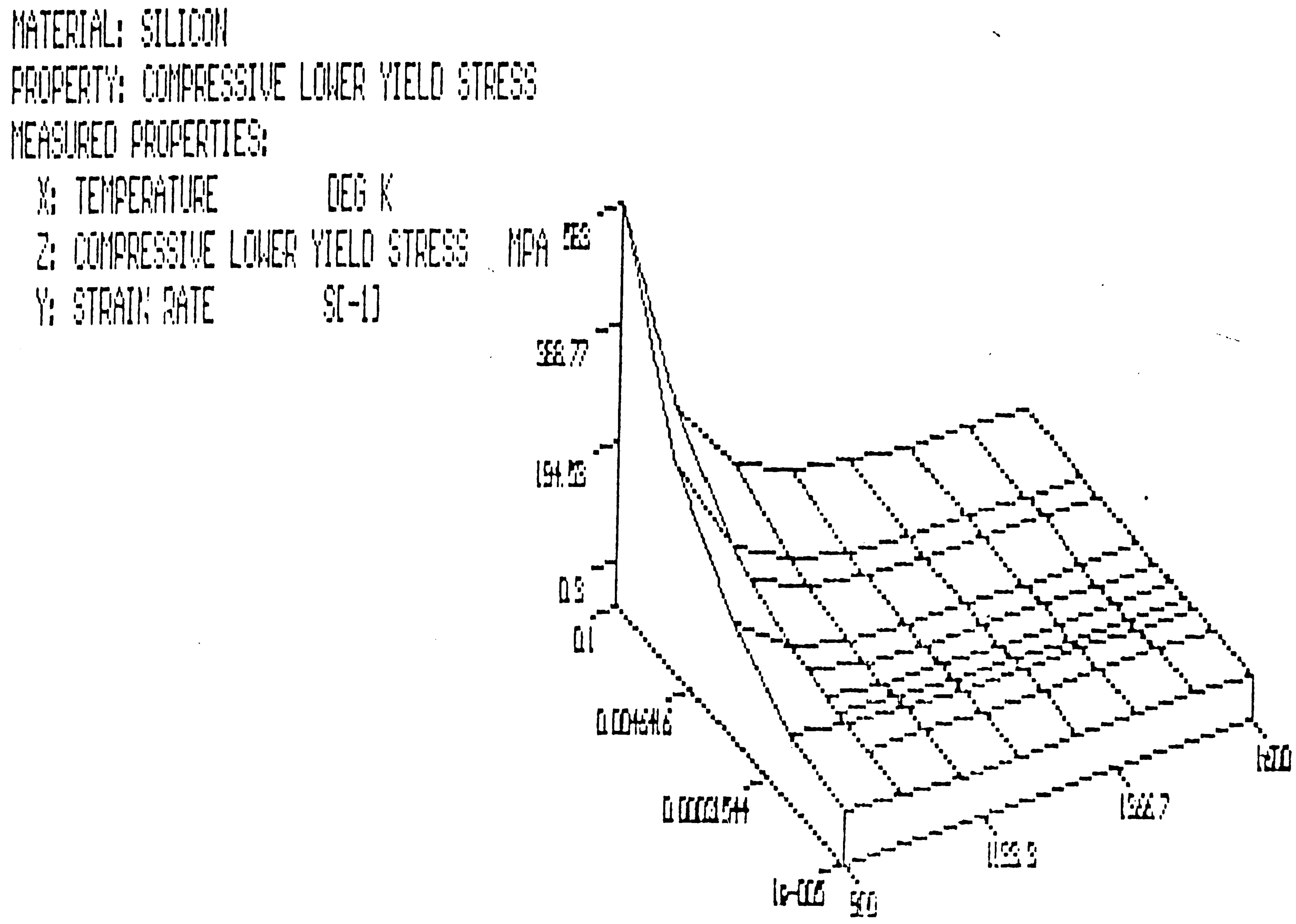

Figure 6.5: Output string comments

In order to output the string values on a graphics mode screen, the variable type of string should be character type. In case of output scale, the type of scale label also should be changed into character type. But these string variables are stored in character type pointer. There is no necessity for type changing. Before outputting, the string values, the size, character type, and color of character are decided. When outputting those character sets, the position of the output string is decided so as not to obscure the graph region (see Figure 6.5).

6.5 2-D graph

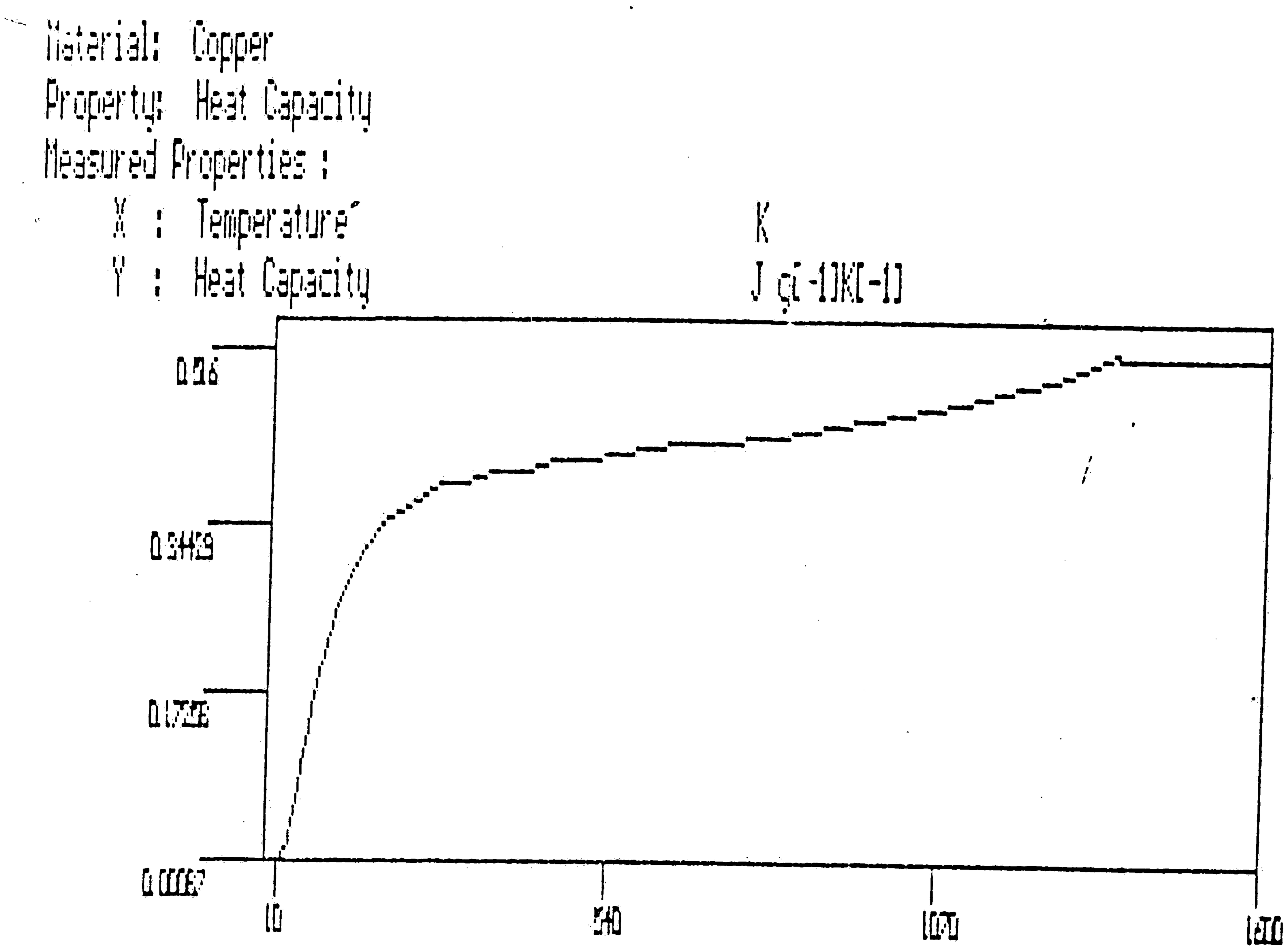

Figure 6.6: Example of 2-D graph

2-D graph drawing is not much different from 3-D graph drawing, except the plane projection from 3-D graph procedure. When the data are read in the 2-D graph procedure, only 2 variables are effective to draw the 2-

D graph. The data set is sorted by ascending x variables to draw the line from low x value to high x value. If there exist other variables, they are treated as dummy variables as in the 3-D graph procedure. And data values are also re-valued with the value of range from 10 to 90 proportional to original value. The drawing process for a 2-D graph is simpler than that of a 3-D graph. The representation tool makes a plot area by finding maximum and minimum x and y variables with the fillpoly of the Turbo C function, a putting the axes and scale on the plot area. Drawing a line of quantitative data is the process of consecutive connection of adjacent two points from minimum x value to maximum value. As a last process, string comments are output at a safe place (see Figure 6.6).

Chapter 7

Discussion

7.1 Summary

This paper was about the graphical representation method in the SRC project. The CINDAS data base arrives at Lehigh University via the computer network ´bitnet´ and used as a material data base for semiconductor package material. The graphical representation tool offers the user an effective way of understanding numerical data of the CINDAS data base by retrieving a data set required for building the graph according to the characteristics of data set. The graph is made using one of 2-D or 3-D capabilities including scale, axes, and some comment lines. The advantages of this tool are: it is developed based on a personal computer, it needs a smaller memory space than commercial graphic package, and it makes the user understand numerical data in a short time.

7.2 Further Work

Until now this program does not have a decision routine for the proper type of scale (i.e., log scale , normal scale, etc.) for the each data set. The type of scale used is to be decided by a human's analysis of data. Especially, when the range of each axis data value varies greatly, for example, the data set of the compressive lower yield stress of silicon are composed of x-data range from 900 to 1600, y-data from 10^{-5} to 10^{-1}, and z-data varying from 0.5 to 583, it is very difficult to decide which axis is fit to which type of scale.

Besides drawing 2-D and 3-Dgraphs, this graphical representation tool can be applied to the areas such as drawing 2-D phase diagram of composite 2 different material and even representation of 3-D phase diagrams of composite 3 different materials is possible by using a slightly modified program.

Chapter 8

Bibliography

[1]. Wiliam S. Cleveland, "The Elements of Graphing Data", Wadsworth Advanced Books and Software, 1985.

[2]. E. R. Tufte, "The Display of Quantitative Information", Graphics Press, 1983.

[3]. I. D. Faux and M. J. Pratt, "Computational Geometry for Design and Manufacture", John Wiley and Sons, 1981.

[4]. Brian W. Kernighan and Dennis M. Ritchie, "The C Programming Language", Prentice-Hall, 1978.

[5]. -----------, "Turbo C Version 1.5", Boland International Inc., 1987.

[6]. James J. Filliben, "Dataplot-An Interactive High-Level Language For Graphics, Non-linear Fitting, Data Analysis, and Mathematics", Computer Graphics, vol.15, no.3, Aug., 1981, pp.199-213.

[7]. Anders Vinberg and James E. George, "Computer Graphics and the Business, Executive - The New Management Team", IEEE CG&A, Jan.,1981, pp.57-71.

[8]. James F Marshall and Robert P. Sheppard, "New Application of Tape Bonding for High Lead Count Devices", Semiconductor Pressing, 1984, pp.500-511.

[9]. C. Y. Ho and Y. S. Touloukian, "Critical Evaluation, Correlation, Analysis, and Synthesis of Thermoplastic, Electrical, Thermoelectric, and Optical Properties

Data", Proc. of the Eighth Symposium on Thermoplastic properties, vol.2 pp.419-430.

Chapter 9

Vita

Dongho Lee, the son of Myeongkyo and Youngran Lee was born on May 21, 1961 in Seoul Korea. He attended Korea University and received his Bachelor of Engineering Degree in Electronic Engineering in March of 1984. He had been in Army service as a duty for one year. In 1986, he continued at Lehigh for his Master of Science Degree in Electrical Engineering.