THE USE OF COMPUTER-AIDED-DESIGN TECHNIQUES IN DYNAMIC GRAPHICAL SIMULATION

by

Carlos Yi, B.Sc.(Eng), A.C.G.I.

March 1976

A thesis submitted for the degree of
Doctor of Philosophy of the University
of London and for the Diploma of
Imperial College.

Mechanical Engineering Department,
Imperial College,
London SW7.

ABSTRACT

The thesis describes the work carried out in the design and implementation of a general computer graphics system for dynamic graphical simulation studies.

The computer based system represents a tool by means of which scientist and engineer can analyse and present morphological and time-space dependent properties of physical phenomena and engineering structures. The system has facilities for man-machine interaction with visual feedback of information. The system has great potential in engineering simulation, computer-assisted learning and production of animated films.

The use of interactive graphics to produce animated displays is investigated and the parallel roles of animation and simulation are compared.

The state of the animation industry, the advantages of computerised animation over conventional techniques, the implications of using computers in a predominantly creative activity and the consequences of such developments are also discussed.

The system was designed to operate in an unrestricted environment, allowing users to create images by inputing pictorial data directly into the computer, interact with the computer to control the process of picture manipulation and to produce suitable copies of work ready for viewing.

Two systems are described: GAGS-2, a general two dimensional system and GAGS-3, a later development designed to overcome problems related to three dimensional input/output and interaction.

The work described in the thesis was carried out entirely by the author during a period of three years.

The areas which are regarded as major contributions to this area of research are featured in the techniques which were developed in the design of the system.

The main features are:

1 - Level of man-machine interaction which permits a high level of performance from the system.

2 - Use of the described hardware configuration, incorporating a digitiser, to create and control dynamic graphics.

3 - Open-ended software design which allows easy upgrading of the system.

4 - Powerful graphics input and output facilities.

5 - Comprehensive library of graphical data manipulation programs.

6 - A hierarchical data structure which allows one to associate physical properties to individual graphical data elements.

7 - A design philosophy which allows total compatibility with existing animation techniques.

PREFACE

The work described in the thesis started as part of a third year undergraduate special project in the Mechanical Engineering Department at the Imperial College, London. The author's interest in engineering, computers and animation plus the availability of suitable facilities in the college led to a research project in interactive dynamic computer graphics where all these interests could be combined.

The project received financial sponsorship from Video Animation Ltd (VAL), a newly-formed company, with backing from the National Research and Development Corporation and Television International Ltd.

The work was initiated in the summer of 1973 and dealt with the design and implementation of a low cost, general graphics animation system specially aimed at solving problems involving engineering graphics.

The system was based on CADMAC (1) a general Computer-Aided-Design system developed in the same department at Imperial College. The software was designed to suit a less restricted user environment and to enable ways of defining and controlling movements interactively.

By the end of 1973, VAL took delivery of its own CADMAC system. The system soon became operational and was tested in a true production environment with considerable success. Commercial jobs were undertaken by VAL for the BBC (including Open University), in experimental work for the College, for advertising firms and major animation studios. This period proved crucial, enabling a direct contact with the potential user market which helped to define an overall philosophy which was believed to be correct. It was found that simple, easy to understand systems could succeed better by encouraging user participation than more intimidating state of the art developments.

The system was operating satisfactorily but only had limited three dimensional capabilities, so by the end of 1974 work on a general three dimensional package was started. The software has been extensively revised, though some of the basic operating principles remain unchanged and many of the interactive programs have been re-designed to handle the expanded three dimensional data and to allow interaction in three dimensional space from a flat surface.

CONTENTS

		Page
Notation		8
List of Illustrations and Tables		9
1.	Introduction	13
1.1	Animation	17
1.2	Role of the Computer in Graphical Processing	22
2.	Applications of Computer Animation	25
2.1	In Research and Education	27
2.2	In Arts	32
3.	System Design	35
3.1	Current Animation Systems	37
3.2	Animation Techniques - Standards and Requirements	40
3.3	Systems Requirements	47
3.4	Animation Languages	61
4.	The CADMAC-11 System and its Components	64
5.	GAGS-2	69
5.1	Programming Considerations	70
5.2	Operating Modes	74
5.3	Data Structure	80
5.4	Picture Construction	91
5.5	Editing	107
5.6	Picture Manipulation	112

Page

5.7 Output 134

5.8 Camera - Compound Movements 143

6. GAGS-3 147

6.1 Data Structure 149

6.2 File Management 157

6.3 Input Techniques for Data Generation of Three-Dimensional Objects 165

6.4 3D Editing 179

6.5 Three-dimensional Transformations 185

6.6 Display 191

6.7 Eye Coordinate System 201

6.8 Joy-Stick Function 209

6.9 Movement Definition 220

CONCLUSIONS 224

SUGGESTIONS FOR FURTHER WORK 227

ACKNOWLEDGEMENTS 230

REFERENCES 231

Appendix A - Special Functions 236

B - Photographing the CRT 243

C - Cubic Spline 246

D - Subroutine RECORD 259

E - CONSUR - User's Guide 262

		Page
Appendix	F - Perspective Transformation	265
	G - Notes on Symbols	274
	H - Background Plotting with the D.C. Plotter	277
	I - Batch Mode Operation under the VAL System	283
	J - Operating Procedures	287

NOTATION

$\bar{A}$ -	vector
X -	vector matrix
X' -	transformed vector matrix
X" -	2nd transformed vector matrix
T -	transformation matrix
R -	rotation matrix
t -	time parameter
Δ -	difference
$\lvert A \rvert$ -	modulus
$\lvert \bar{A} \rvert$ -	magnitude of vector
$\bar{A} . \bar{B}$ -	dot product
$\bar{A} \times \bar{B}$ -	cross product
$A \propto B$ -	proportional
$[x, y, z]$ -	cartesian coordinates in three dimensions
$[x, y]$ -	cartesian coordinates in two dimensions
$[A_x, A_y, A_z]$ -	cartesian coordinates of point A
$[\bar{i}, \bar{j}, \bar{k}]$ -	unit vector along x, y, z axes
$[\alpha, \beta, \gamma]$ -	angles to the x, y, z axes
$\bar{E}\ [E_x, E_y, E_z]$ -	observer eye position
D -	distance to the projection plane from the eye position
θ -	pan angle
ϕ -	tilt angle

LIST OF ILLUSTRATIONS AND TABLES

Figure 3.1 Conventional animation procedure

3.2 Bar-sheet

3.3 Exposure sheet

3.4 Standard 16mm field guide

3.5 Animation with computer

3.6 Basic operating requirements

3.7 I/O devices

4.1 Block diagram of the CADMAC-11 system

4.2 The CADMAC-11 system

5.1 System interaction

5.2 Digitising table with menu

5.3 Command structure

5.4 Operating sequence

5.5 Picture hierarchy

5.6 Data/file structure

5.7 Components on three levels

5.8 Example of file with macros

5.9 Find mode

5.10 Chebyshev's points

5.11 Cubic interpolation

5.12 Example of scan-processing technique

5.13 Example of shaded shape

5.14 Cross-hatched section

Figure 5.15 Three types of alpha-numerics

5.16 In-betweening fairing

5.17 In-betweening of a cartoon character

5.18 Parametric path representation

5.19 Operation of function library

5.20 Commands menu

5.21 One-pass display processor

5.22 Processor sequence

5.23 Windowing transformation

5.24 Trivial cases of clipping

5.25 Intersecting lines on window

5.26 Plotter with cel on peg-bar

5.27 A 5(6S-4E)-2(10N-10W) pan-zoom

5.28 Exposure table for rostrum

6.1 Basic system architecture

6.2 Storage format for data records

6.3 'I' storage format

6.4 3D display files

6.5 Direct viewing

6.6 Input sequence

6.7 First angle projection

6.8 Third angle projection

6.9 Modified first angle projection

6.10 Contour map

6.11 Semi-oblique view of surface

6.12 Uniform cross-section

6.13 Solid of revolution

Figure 6.14 Parallel editing technique

6.15 Distance of a point from a line

6.16 Right-hand reference set of axes

6.17 Rotation about z axis

6.18 Rotation about y axis

6.19 Rotation about x axis

6.20 3D display

6.21 Perspective - orthogonal projections of a cube

6.22 Eye coordinate system

6.23 Rotation of axes in space

6.24 Orientation of axes for a vertical line of sight

6.25 Linear perspective

6.26 Eye position in terms of pan and tilt angle

6.27 Lens zoom

6.28 Object zoom

6.29 Combined zoom

6.30 Joy-stick patch

6.31 Flight sequence

Table 3.1 Display rate of polygons

5 .1 I codes in data record

6.1 Data packed in I

6.2 ICODES

Figure C-1 Single span in function U = f(t)

C-2 Two spans

C-3 Secondary weighting equation

C-4 Start conditions for cubic spline

C-5 End conditions for cubic spline

C-6 Overlapping long splines

C-7 Parabola through three points

F-1 Projection on plane PP

F-2 Intersection of planes PP and H

F-3 Singular transformations

I-1 BSF command record structure

I-2 Loading sequence in the BSF

I-3 Flow chart of Batch/on-line operation

J-1 Title sequence

J-2 Load sequence

1 INTRODUCTION

There are many areas in science and technology where problems cannot be studied experimentally because it is either too difficult or expensive to build a true model or to carry out the experiment in the actual physical environment. Examples of these are flight simulation, space research, destructive testing, urban planning and landscape engineering. In other cases the phenomena is only describable in mathematical form, a mathematical model which one would like to compare with the actual physics. Simulation represents a powerful way of presenting or studying these physical phenomena. With simulation one is not restricted to a physical world, but can abstract and conceive a new hypothetical universe if necessary.

With the addition of pictures the problems can be presented visually and analysed in a realistic manner. By representing the mathematics in a form that can be visualised an empirical formulation can be compared with actual experimental results; e.g. in the derivation of fluid flow equations, the stream lines are plotted and compared with results from flow visualization experiments; or in the presentation of the effect of pressure applied to a plate with the resulting distortion being magnified on a display and discontinuities or irregularities detected visually. With an interactive on-line system the parameters controlling the model can be varied to obtain a new set of parameters defining a more accurate representation of the true physical phenomena.

Animation has been extensively used in many areas as a form of dynamic graphical simulation allowing a factor of real time to be included in subjects that have time or space dependent characteristics or that would look ambiguous when viewed in a static form. Animation is a very effective way of presenting data using the ability of the human eye to detect movements and pick out details that would otherwise have been missed or been too complicated to be described simply in words.

Many technical subjects have been successfully presented in the form of animation from simple animated brochures to studies of problems of very high complexity. Its potential as a teaching aid cannot be overlooked either, in that many subjects can be explained better in the form of films than by diagrams or descriptive equations.

The medium used has been predominantly film but the trend now is away from it, towards more convenient and economical audio-visual techniques, such as video, which provide a more convenient and portable way of storing and displaying images.

The main problem in the production of such animated films has been the dependence of scientists on professional film-makers still using very awkward and antiquated techniques. Most scientific films are still made by or at least, to some extent, with the co-operation of these film-makers still using conventional techniques which, whilst quite adequate for general purpose films, prove totally incapable of handling complex scientific subjects. Yet scientific animation should be prolific and amongst the easiest to produce. The demand for such films is large in both academic and industrial circles, the quality of the finished product need not be of very high standard and in some cases the action is quite restricted. These movements should be precise which represents no problem because they are often described by accurate mathematical relations and changes taking place within a system can also be represented in a mathematical form.

The areas of simulation, mathematical modelling and animation often overlap. Thus it becomes natural to associate the making of scientific simulation films with the use of powerful calculating machines like a computer. The existence of a suitable computer based system would not only simplify the making of such films but also encourage the non-expert film-maker to try to create his own films to his own requirements.

In most cases the solutions to the problem are well known; only the means of expressing them graphically and dynamically in a controlled manner are lacking. The system described in the thesis provides such facilities. The aims of the project were:

1 - To provide a means of presenting solutions to certain technical problems graphically.

2 - To enable on-line simulation of certain technical processes within a realistic time-span.

3 - To provide means of controlling movements represented by mathematical transformations.

4 - To facilitate the production of difficult technical animation.

5 - To enable scientists and engineers to independently make their own research or teaching films.

6 - To provide alternatives to the use of the film medium in the presentation of dynamic subjects.

The last three are directly related to animation which still plays an important part in dynamic graphics simulation. Here animation is regarded as synonymous with dynamic simulation. As aforementioned, scientific films are still made with the cooperation of the animation film industry, an industry not without problems going through some of the most critical moments in its history. The animation industry is in fact ripe for a revolution, not only to extricate itself from its self-imposed obsolescence but also to ensure its very survival, a revolution where computers are likely to play a very major role.

1.1 Animation

Animation is an art and as such it is not unlike other artforms where the end result is normally the product of many hours of hard and patient work. But animation has the added drawback of incorporating many steps which not only lack in creative value but also are very time-consuming and tedious.

With the exception of a few experimental films most animation is produced by using photographic techniques. This is probably just a historical coincidence,due to the advent of photography and animation at the same time during the turn of the century, but which has meant that for each second of the projected film,24 frames, with only minute differences between them, have to be individually created, composed and finally photographed (a more complete description of animation techniques is given in section 3.2).

This production technique entails a prodigious amount of labour, often highly skilled labour, which tends to reflect on the cost of animated films, more expensive frame per frame of the finished film than any live-action film.

'Animal Farm', the first feature length film made in Great Britain by Halas and Batchelor runs for 75 minutes and required 300,000 man-hours to create 250,000 drawings.

Although the making of a film is always credited to one individual, the director, it is only made possible through the effort of a well coordinated team effort, especially in the case of long animated films. Every stage in the production is intricately planned from the basic conception to the final ready for viewing film. Excellent communication channels, which do not always exist, must be established in order to prevent mistakes and to increase productivity. These communication problems become magnified when members of different professions are involved, as in the case of a scientist trying to convey the contents of a technical film to the producer. Quality control is another important consideration when attempting to produce large quantities of artwork of impeccable standards.

It was his obsession with perfection and desire to maximise profits that led Walt Disney to introduce assembly line techniques to the production of films. These techniques are now well established throughout the world particularly in Russia, Japan and USA and despite the advent of many independent artists and the existence of excellent one-man films this production line approach, with detailed planning at every stage,became regarded as the conventional way of making animated films.

Many of the innovations introduced by Disney are still in use today: production scheduling, transparent plastic cels, special cameras, multi-plane photography, new inks, paints, recording techniques, etc. These techniques have since changed little and are becoming increasingly obsolete. The increase in cost of labour and materials, agravated by the adverse economic climate,where industries like entertainment are the first to be hit,has led to the animation industry's present precarious condition. Many large scale studios have closed down and good quality feature length films never conspicuous before become increasingly rare.

There have been attempts at mechanisation such as the use of photocopying techniques to reproduce artwork in large quantities and standardization of certain cartoon features. Computers have been introduced in some establishments to perform housekeeping tasks like information storage and production control.

There were attempts at automating rostrum cameras by motorising the camera and table compound controls and operating the motors from the console of a specially built analog computer (2). The use of a specially designed, dedicated analog computer proved an extremely expensive arrangement. A cheaper alternative would be to use numerical control by generating instructions punched on paper tape or stored on magnetic tape from a specially programmed digital computer. To convert the coded instructions to rostrum movements very simple logic is required. A micro-processor can be used to interpret the instructions and drive stepping motors to move the camera up or down and the tabletop E-W, N-S. The commands used to generate the n.c. tape can be very simple, simular to the ones used to control the motion of the rostrum (see section 5.8).

Computers have also been used to mechanise some of the creative steps in cartoon production for television. The different positions of the cartoon character's body have been rationalised and classified by numbers. The position of the head, mouth, limbs, body vibrations (as a result of shock, entry, exit), everything is numbered and stored in a data bank. To make a new sequence it is sufficient to plan the sound carefully and then simply draft a script which gives the number of the keys, feed into the computer and the machine will do the rest. Animators no longer have to draw successive movements, but merely make out lists of numbers. This reduces the art of the cartoon to painting by numbers. A more flexible alternative is to use a suitable movement notation (3, 4) which may be numerical or graphical to control the movements of a skeleton on which the body of a character can be superimposed.

Rationalization and machines may save labour and enable mass production of drawings as required by television, thousands of feet per week. However they cannot supply mass production of ideas. The computer is used primarily as a data processor providing no facilities for creative design. What is required here is a system through which the user can create and refine original work and have ways of defining and controlling movements instilled in the artwork. The computer should minimize the frustrations of tedious repetitive work and allow the animator to produce streams of images not necessarily on film with the same facility as he would produce a single picture now.

Two factors should prevail when using a computer to solve a problem: time consumption and repetitiveness. The computer is a machine that can deal quickly with these problems and animation is a process with such problems. The use of a computer when compared with conventional human techniques should take the following into consideration.

Complexity

Quality

Productivity

Cost

The first three should be increased and the last one decreased. Only when the above criteria are met can the use of a computer be economically justified . There are however some cases where this does not occur, as in work of very high complexity, which would not have been envisaged to be done by hand, and the computer can provide the only, albeit expensive solution.

Normal applications of computers relate to technical data like forces, temperatures, velocities, masses, numbers, etc. In animation data is mostly pictorial in nature so a suitable system should be capable of manipulating graphical data in the same way as its scientific counterpart can absorb numerical data. Such a system can be found in the guise of a computer graphics system.

1.2 Role of the computer in graphical processing

With decreasing costs and increasing sophistication computer graphics has become an attractive and competitive solution to many problems involving man-machine interaction. By using the very fast speed of display of most graphical output devices, and the very quick reactions of the human eye it is possible to provide the user with an insight into the processes of a computer. The visual display unit interfaced to a computer provides a direct means of visualizing the results of any computation, and enough information to allow the user to make decisions very quickly on-line.

Computer graphics is one of the most extensive branches of computer technology and represents an increasingly important area of application for computers. Applications of computer graphics include study of molecular structures in chemistry, medical research, computer-aided-design, and simulation in the aircraft, shipbuilding and automobile industries, integrated circuits and printed circuit board design, pipe routing and layout, architectural and highway planning, etc.

As a graphical processor the computer plays two distinct roles: always that of a high powered draughting machine and sometimes, particularly with mathematical subjects, that of a calculating machine which determines the consequences of mathematical and logical statements. In the first case the computer is used as a controller generating at very high speed signals to drive a suitable output device from data stored in its memory. In the latter role the computer typically accepts the description of a hypothetical system such as a mathematical model, determines the successive states of the system by solving differential equations or physical laws and modifies the data base accordingly, and then uses its drawing capabilities to render a series of views depicting the resulting events.

The data processed by a graphical system is pictorial in nature and should be represented in the form of a suitable data structure which can be manipulated by mathematical transformation routines. Graphical data is inherently bulky and inefficient but the large storage capacity of the memory devices normally associated with computers can allow for a considerable amount of data to be accommodated before any overflow occurs.

In order to be able to edit the graphics on line and dynamically a means of interacting with the machine is needed. In interactive graphics the computer has the ability to question, respond or converse with the user. It provides some form of information feedback to help the user in decision making processes like accepting, rejecting or changing the present data. This ability to interact is essential when considering the large number of pictures associated with animation. These pictures, with only slight differences inbetween, can be easily generated by the computer itself from a single initial static picture. A program controls, accurately, the transformations required for each frame, taking animation to a level of mathematical precision. An interactive computer graphics system should satisfy all the necessary requirements for animation work: high speed of response, mathematical precision, large storage capacity, display ability. Visually animation and computer graphics also have many aspects in common:

1 - They are both highly stylised and simplified forms of imagery.

2 - They usually consist of simple line drawings of uniform thickness with areas left clear or uniformly shaded or coloured.

3 - They are both produced for display on a window of fixed size and aspect ratio.

4 - Many of the features incorporated in computer graphics display contain the same basic movements as the rostrum camera: windowing-zoom, translation-panning, rotation-spin or flip.

The display on a graphics terminal may be regarded as a static animation frame. The animation being obtained by introducing small changes from frame to frame. When viewed at normal projection speed these quantised changes will result in a perception of movement. Animation has often been defined as the art that takes place between frames.

It is fortunate that the development of computer graphics has coincided with the current trend for simplicity of form and movement in animated films unlike the more life-like style of the Disney cartoons. In fact the graphical output of the computer display provides a more appropriate style for animation than attempting to imitate live-action films.

2 APPLICATIONS OF COMPUTER ANIMATION

The use of computers in animation has been applied in many cases with considerable success. Computer animation is not new, dating back more than ten years, to the first computer animated film made by E. E. Zajac in 1963 at the Bell Telephone Laboratories in the United States. The film, produced on a SC 4020 computer driven cathode ray tube manufactured by General Dynamics Corporation, showed the simulation of the motion of a communications satellite in three dimensions. In fact computer films are known to have been made on the Whirlwind computer at the Massachussetts Institute of Technology as early as 1951 but these early attempts had little impact. (6, 7).

Research and development in computer animation have been restricted to scientists and engineers in research and educational establishments, but applications for it are now being increasingly found in artistic fields, in the production of cartoons and television commercials. As predicted by Zajac, installations with facilities for making films by computer have increased from a handful to become available in most major universities and industrial laboratories in the United States and Western Europe.

2.1 In research and education

This is the area where computer simulation has been most widely applied. Most films produced to date were of a scientific nature, and since the criteria for these films is not very high, often the simple graphical output from computers represented an ideal style.

In the conventional way of producing animated films every frame must be drawn and then exposed one by one. If the film content is mathematical, elaborate calculations must be performed before any particular frame can be drawn. On occasions the necessary calculations may be so long and laborious so as to preclude attempts to produce such a film at all, using conventional methods. It is in these situations that computerised techniques score over conventional methods. The calculating power of the computer can be used to determine the state of a particular motion sequence frame by frame and using a microfilm recorder or a direct video output eliminate even the necessity to draw and film these frames.

By feeding the Cathode Ray Tube (CRT) with data from a computer it becomes very easy to make animated films illustrating a mathematically complex sequence of calculations. The technique has great potential in enabling research workers to visualise the results of computation and to prepare educational films. The animated display is a natural medium for the recording and analysis of computer output, simulation and data reduction, modelling presentation and elucidation of phenomena of physics and engineering.

The human eye has great pattern recognition ability, for this reason a common way of handling scientific data is to plot it. Numerical computer output is woefully inefficient. The human eye can also quickly pick out a moving object from a static background, and films allow one to take advantage of this ability by adding the dimension of time to the familiar spatial dimensions in studying computer output. With computers it is also easy to extend drawings from the ordinary two-dimensional plane to three dimensions by simply including an extra array for the z coordinate. By means of appropriate visualisation aids and depth cues one can study the behaviour of time dependent tridimensional data.

In scientific research the most obvious film making function of the computer has been in simulation, with results presented graphically and dynamically. Depiction through animation is particularly appropriate where simultaneous actions in some systems must be represented. Examples of simulation techniques used in studies of fluid flow, satellite motion, automobile dynamics and medicine can be found in references (7, 8, 9, 10, 11, 12, 13).

The use of computer animated display of scientific data can generally be regarded as a form of education, where one scientist is passing information to another. From the pedagogical point of view the smooth animation and mathematical precision of computer make an elegant and mathematically beautiful lesson in physics. Its total effect cannot be matched by any amount of gesticulation and hand waving at the blackboard and by few physically realisable demonstration equipment. Here the laws of nature can be abstracted, idealised or revised to a far greater extent than with any other teaching aid or experimental equipment. Students should find easier in some cases to perceive basic physical phenomena from movies than from general laboratory experiments. It is possible to make perfectly comprehensible instructional films without a single word of commentary and subject material that is normally presented by lectures only could be profitably supplemented by films.

One immediately thinks of a whole host of phenomena and concepts that uniquely lend themselves to illustration by computer films. Examples are mathematical functions, limits, ratios and expansions, Newton's laws of motion in two and three dimensions, kinetic energy in physics and chemistry, fluid mechanics and so on. In fact one can argue that a good deal of the instruction in the physical sciences is in terms of films of mathematical models of nature, except the student does not normally see the films, but only hears verbal descriptions and equations representing them.

In computer assisted learning a suitably programmed system could provide a stand alone tutor. The student is being taught by the machine, yet he can have overall control over the teacher and progress as fast or as slow as he can manage. An example of such experimental work was the use of animated displays in teaching engineering drawings. Here one had the three dimensional descriptions of the object being designed displayed on the screen. The object was rotated and orthographic views projected on the major planes which were then unfolded to reveal the finished drawings. The display was shown to groups of undergraduate engineering students, who had first attended the lecture, further illustrating the contents of the lecture. Overall response was good, but due to the size of the screen it was felt that a film would have been a more appropriate format.

Films, for scientific and educational purposes, should preferably be made by the instructors themselves, who will be using them, or at least made with their close cooperation. This would ensure that the film conveys exactly what the instructors intend,instead of a close facsimile that might be produced by a well intentioned film maker. This is where computer animation can be most useful. It gives the scientist or engineer the opportunity of making their own films. The language used in computers is mostly mathematical, a language in which scientists are proficient. Thus the scientist can make movies with the minimum of dependence on directors, producers and animators. Much as writing a book where one submits a manuscript and receives back a proof in print, ready for reading,so can one now submit a program and receive back a proof film ready for viewing. The scientist can be the master of his house; if he wishes he can maintain complete creative control over the film he makes, with the minimum of delay, usually incurred between producer and film making mechanism.

Contrary to being threatened by this development the professional science film maker can also take advantage of its flexibility and power. He has much more freedom to try out his ideas quickly and unlaboriously. The ease, speed and economy of computer animation permit the film maker to take several tries at a scene producing a whole family of clips from which he chooses the most appealing result, a luxury never before possible.

The exploit of the full educational potential of computer animation requires the partnership of three specialties: computing, film-making, and the subject matter, science or technology on which the film is to be made. One hopes to see computer produced films to become a significant adjunct, playing an increasing role, to technical education and scientific research.

2.2 In arts

Despite the emphasis on the use of computer animation for scientific research and educational purposes, one should not overlook its potential in the areas of art and entertainment.

Computer art dates back to the early 1950's. Though there are no masterpieces to be associated with, it is nevertheless a movement of unique significance both socially and artistically. It is an international movement, motivated by the use of media, technique and method rather than ideology, and into which people from a variety of professions, disciplines and walks of life can participate. Universality of communication is encouraged by the fact that computer languages are international. Few of the so-called computer artists are what one would in fact regard as an artist. W. Fetter (13) has won awards in art competitions with his Boeing graphics developed for purely technological reasons. The main exponents of this computer art are engineers, mathematicians, scientists and some artists, all those who have access, know-how and desire to exploit the computer and its peripheral devices for making pictures. This movement demonstrates that creative activity need not necessarily belong to the conventionally prescribed areas of painting, sculpture, music and poetry. It also demonstrates that creative activity is not the prerogative of those with diplomas from an art college or of those professionally engaged in these fields.

Animation is very much a creative activity, therefore even when the project is more devoted to the technical and mathematical aspects of the subject, the implications of using a scientific media for purely aesthetical reasons can never be avoided.

The question arises in some people's minds about the artistic merits of purely scientific techniques when applied for artistic purposes. One would like to think that the end result, no matter through what medium it was produced, should be judged for its own artistic merits. The fact that these films are produced by a digital computer performing all sorts of mathematical and logical operations should be incidental to the artistic effect thereby achieved.

The computer can be used in many fascinating themes such as translation of musical form into visual terms. Music consists of a dynamic succession of events taking place at determined intervals of time. Its conversion into visual forms is best achieved by an animated display with changes of determined shape and intensity to represent corresponding notes. One experiment carried out was to automatically generate notes from a random numbers program. The notes were printed on a line printer in the form of a musical score. By means of an appropriate sound input device the sound waves can be transformed, through appropriate software into corresponding imagery. This led to another experiment in speech analysis. Speech was converted into coded symbols punched on paper tape. The paper tape was then fed to a program which converted it into a display of moving lips. The program was developed to automate the synchronisation of lips in character animation but could also be developed to help deaf people to learn to speak.

Apart from educational purposes, films of mathematical formulae can be made in which their graphical forms are exploited aesthetically. The film 'Squares', made in the Mechanical Engineering Department using a PDP-8 mini-computer is such an example. Most people, including many artists, are awed by the thoroughly fascinating beauty of computer arts.

However, despite its apparent potential as an artistic medium, work done on computer arts and animation has tended to be restricted to scientists and engineers for research and educational purposes. This is perhaps due to the reluctance on the part of some artists to use devices they do not understand well or to the unfortunate concept that engineering devices are developed for practical purposes only and must be economically viable before allowed to be used in a less utilitarian environment.

The potential of these techniques involving the new technology, and in particular computers, will only be fully exploited when the artists learn to accept and use these new tools as a new creative medium and learn to collaborate with scientists. This could be responsible for providing a link between men of arts and men of science and fulfil their desire to extend their individual activities.

3 SYSTEM DESIGN

There are several important considerations when designing a system to meet wide ranging requirements.

Firstly, it is worth looking at other systems developed so far, areas of main interest, successes and failures and overall performance compared with cost and hardware/software implementation.

For animation purposes the system should be compatible with most of the present techniques to permit easy integration. There are many areas in the conventional production techniques where the computer can make major contributions.

On the hardware side particular attention should be paid to man-machine interaction. Although most probably based on digital hardware, the external operation of the system should resemble that of an analog device since after all, all of man's faculties of thinking and behaviour are still essentially analog. Suitable input and output devices can provide this natural extension.

On the software side provisions for direct control of the process should be available. This could be in either the form of a special purpose language or by means of symbolic commands.

In an interactive system hardware and software have complementary roles. There is direct interplay between them, and in most cases hardware can be emulated by software and software algorithms can be hardwired. It is always desirable to have the most sophisticated equipment operating in near ideal environment, this is however seldom possible with circumstances being dictated by past decisions and financial considerations. Yet by adopting the right philosophy and with a well defined goal in mind it is possible to make the not so ideal equipment operate effectively and produce good results.

3.1 Current Animation Systems

During the past ten years or more many computer graphics systems have been used to produce computer simulation movies. These varied from the multi-million dollar colour display system developed by General Electric Corporation for the NASA, capable of handling complex three-dimensional objects in real-time, to modest graphics installations in some universities where computer films were first experimented with.

Most systems were based on digital computers with the major exception of Computer Image Corporation's Scanimate system. In this totally analog device potentiometers controlled from knobs and slides on a console are used to modify the video signal fed from a video source. The user selects a number of effects on the console and the distorted images are displayed on a colour television monitor. Any desired sequence can then be automatically recorded on 2 inch video tape. The system is simple to operate and can produce some very impressive results. It had considerable commercial success when first introduced but this success was short-lived because it was a rather expensive facility to use, even on hired time, and the results produced showed a sameness in its quality which made them easily recognizable and tiringly repetitious. A bad symptom in an industry relying largely on the impact of novelty for its success.

Computers on which animation systems have been based varied from the initially rather unsophisticated minis to large main frame machines covering nearly the entire range of computers in the market. Early systems were mostly software packages running on machines with special graphics hardware like hardcopy plotters, visual display units or microfilm plotters, but latterly an increasing number of systems have incorporated specially designed hardware to replace many of the slower software functions. A good example is a hardwired solution to display problems by Evans and Sutherland Corporation in Utah, USA. The device called a Watkins Box, a hardware implementation of Watkins' hidden-line algorithm, incorporated very powerful clipping, shading and perspective transformations performed in hardware matrix multipliers. The device provides very fast display of moving, hidden-line removed description of solids, but the cost ran into nearly half-million dollars.

An area which received considerable attention was 'real-time' animation. This is only a relative term since in animation there is no real-time. This is normally meant to describe systems whereby pictures can be displayed without unacceptable flicker and with a true subjective feeling of continuous movement, i.e. at least 10 frames per second. The success of such systems depends on the complexity of the picture, resolution of the display device, the speed of the display algorithm and how far one is prepared to go to implement as much as possible in hardware. A system restricted to simple line drawings and crude movements in two dimensions could be developed quite cheaply but then it would not suit many applications; on the other extreme there are fully coloured, shaded, smoothly animated three dimensional solids. Special hardware is impressive and can speed up display considerably, especially in three dimensional graphics, but there is always a heavy penalty on cost.

Another area of interest which received considerable attention, especially from the University of Utah and the CAD centre at

Cambridge, was to produce displays with a photo-realistic appearance. This involves hidden-line removal, total or partial and smooth shading of three dimensional surfaces. The results can be very impressive but are normally restricted to a few frames because of the many hours of computing and display time required. They have little relevant potential in animated simulation applications.

The above systems worked well to a greater or lesser extent and some produced extremely beautiful results but were all developed for very special applications. None can be regarded as flexible and practical enough to produce results that resemble more the user's intentions than the characteristics of the system.

The systems were developed for particular uses in universities and research institutes to satisfy very special requirements, either to produce research films on a particular subject or to analyse the potentialities of certain techniques. They were mostly systems exploiting some dynamic graphics capabilities rather than a computer-based system specially designed for the purpose of producing animated movies, whether on film, video, display or paper. They were not intended for use by any specific industry, least of all the animation industry, yet it should be remembered that whether the subject matter of the film is technical or general it is still largely produced by a well established industry. Thus it is worth complying with some of their standards and requirements and produce a user oriented system with viable production capabilities.

3.2 Animation Techniques - Standards and Requirements

Animation belongs to a very archaic industry. Little or nothing has changed since its beginnings nearly eighty years ago. Some of the equipment has become more sophisticated and materials have been improved, like better films, paints and drawing materials, but the basic technique of drawing and composing pictures individually and shooting frame per frame on film has remained quite unscathed. Fig 3.1 shows the main steps involved in the production of animated films by conventional techniques.

Films are written in the form of story boards which consist of a series of rough sketches accompanied by commentary or sound-track. These are subsequently planned carefully and timed on a bar sheet. The bar sheet is a long chart with a detailed breakdown of the sound-track and a listing of the key positions (Fig 3.2). The key positions represent the extremes of action in a sequence and contain the rhythm and pace of the entire movement. They are normally drawn by the key animators based on the characters designed for the sequence. This is where the largest concentration of creative work can be found and the computer is of little help, except in some design exercises when the facilities exist.

Assistant animators or in-betweeners use the key frames to produce the intermediate frames. These 'in-betweens' define the smooth action between the extremes and give rise to the actual animation as seen on the screen. Although some creative skill is required this process is mainly a mechanical one, since the characteristics of the movement are already well defined within the key frames. The computer has great potential here as in-betweening is equivalent to an interpolation represented by mathematics which can be used by the computer to generate the intermediate drawings based on the key positions.

The complete artwork is finally transferred to 'cels', transparent celluloid sheets registered by holes on a peg bar, inked and coloured. The cels are still the industry's main working medium and the whole industry is geared to handle them, so one essential requirement for an automated system is to be able to produce large quantities of output of high quality on standard cels.

As each cel is completed and checked an exposure sheet is prepared (Fig 3.3). The exposure or 'dope sheet' contains instructions for the cameraman on how to compose and expose each frame of the film. The instructions consist of zooms, pans, fades, wipes and mixes and are expressed in terms of centre coordinates and field sizes on a field guide (Fig 3.4).

The cels are exposed by using a rostrum camera which consists of a one or two-column stand with the camera mounted on it, allowing vertical up or down movements and a control panel which regulates the exposure of the film and the movement of the film through the camera. The artwork is placed on the compound tabletop which is restricted to horizontal E-W, N-S moves under the camera. The movements of the camera and the tabletop are coordinated with minutely calibrated controls which allow very accurate positioning of the camera with respect to the artwork.

All the stages described above are essential but slow and tedious and very careful checking is required to prevent mistakes.

Most of these stages can be automated since the information required to change the drawings is essentially the same required for timing, checking, exposing and editing the sequences. This information can be centralised on a computer and be used to modify the drawings stored in a memory bank, to automatically produce the in-betweens, breakdown the sound track and to list out exposure tables. Checking would be redundant since the whole process would be automated and the computer is less likely to commit errors. The entire process can be reduced to that shown in Fig 3.5.

There are also important economical considerations in the design of the system. The animation industry is rather small and lacking in funds and cohesion, consisting mostly of a number of small studios and many intermediaries, the agents. It is very labour intensive with little automation, if any at all,and the only capital investment of importance is usually in the form of a rostrum camera. It is unlikely that such a small conservative establishment is willing or able to finance research in computer applications or even invest on such a system. The type of service one is likely to offer is more on a bureau service basis where individual jobs can be handled competently and efficiently by well trained staff. There are however larger organizations,such as the BBC or the National Coal Board,who produce large quantities of high-quality technical films and can afford both the financial and human resources to benefit from a turnkey or lease system, and of course, universities where such facilities are expected to be found.

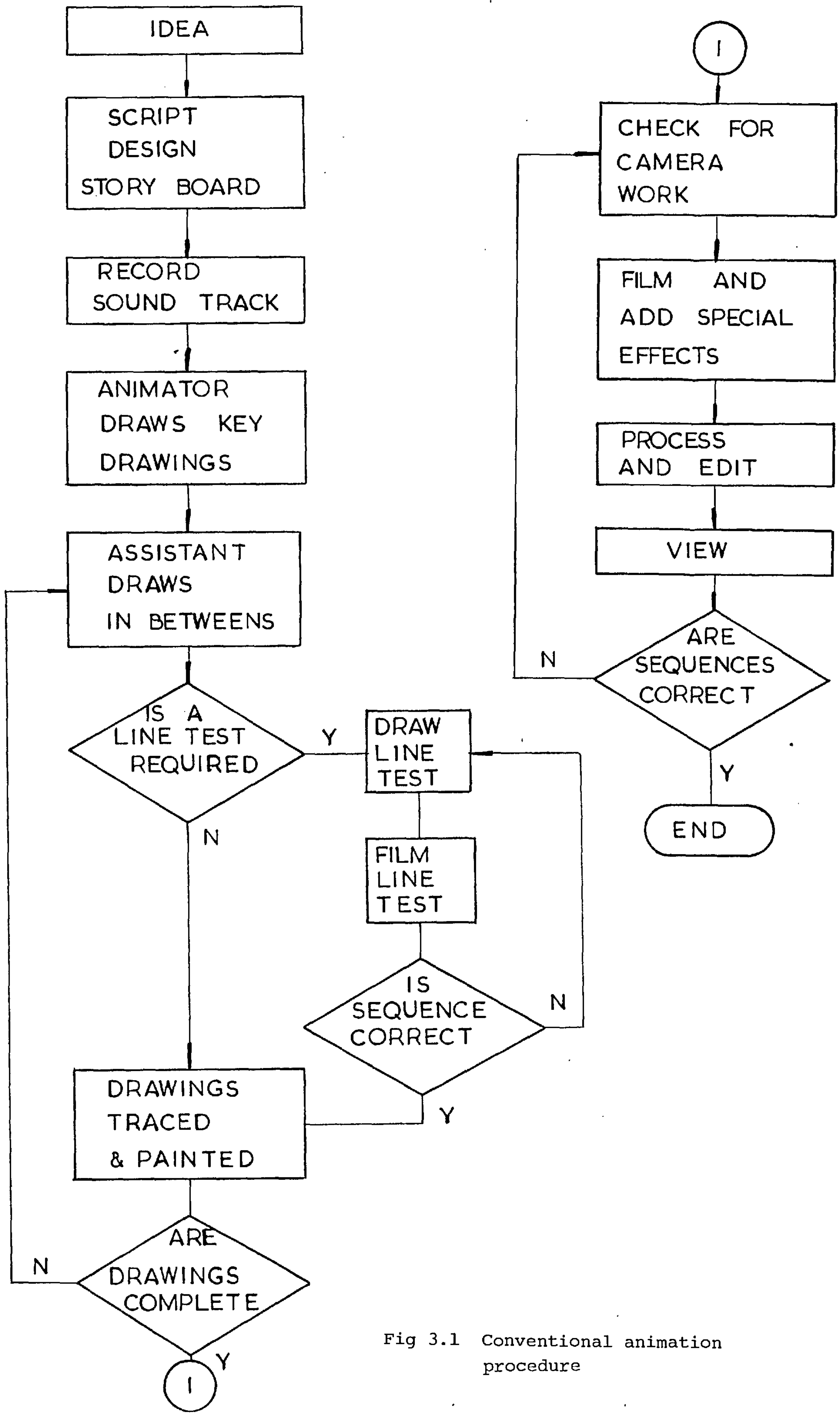


Fig 3.1 Conventional animation procedure

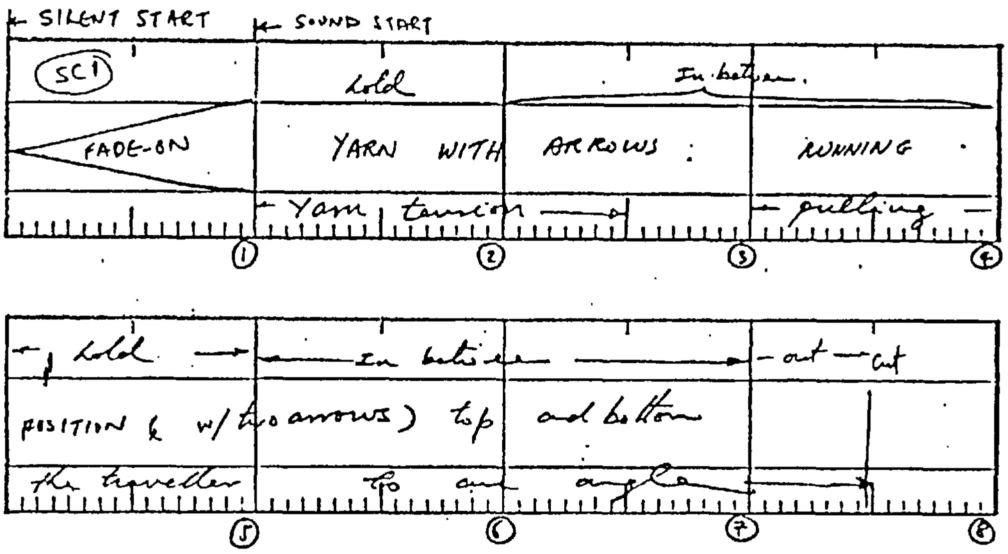
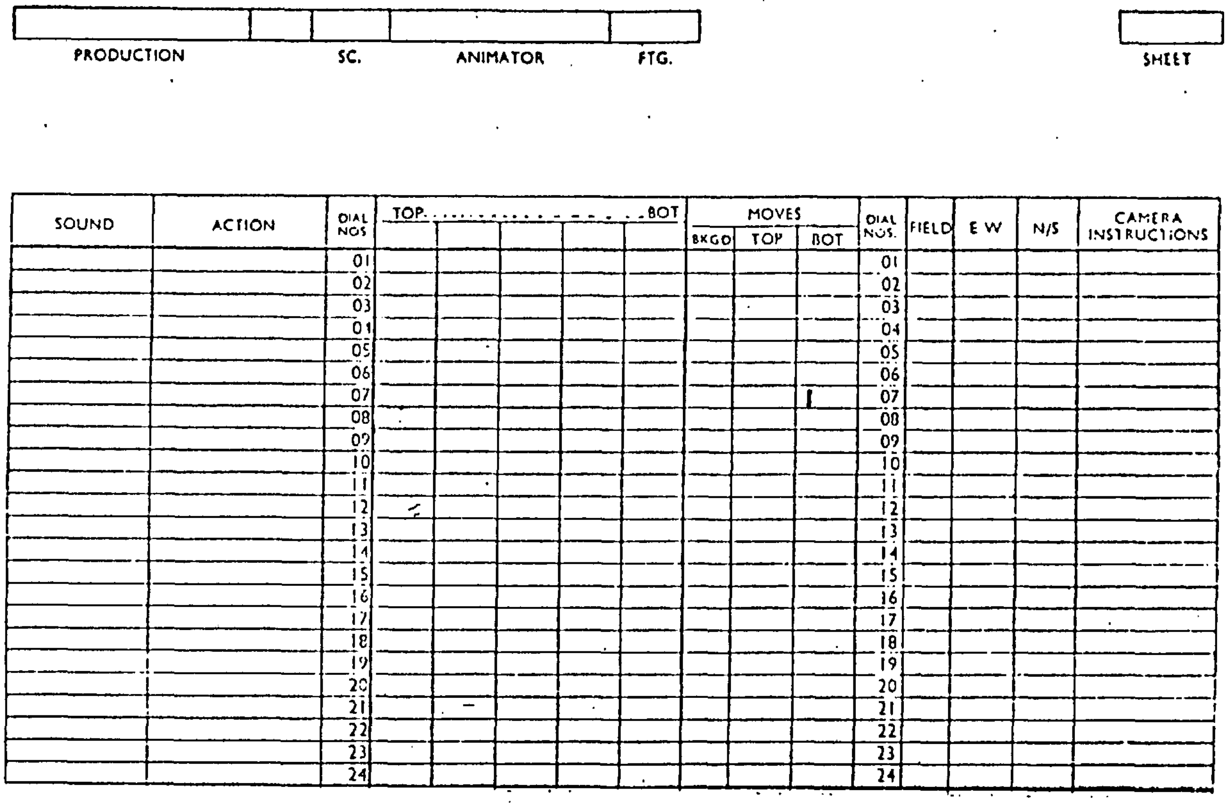


Fig 3.2 Bar-sheet



PRODUCTION	SC.	ANIMATOR	FTG.		SHEET

SOUND	ACTION	DIAL NOS	TOP.......BOT					MOVES			DIAL NOS.	FIELD	E W	N/S	CAMERA INSTRUCTIONS
								BKGD	TOP	BOT					
		01									01				
		02									02				
		03									03				
		01									04				
		05									05				
		06									06				
		07									07				
		08									08				
		09									09				
		10									10				
		11									11				
		12									12				
		13									13				
		14									14				
		15									15				
		16									16				
		17									17				
		18									18				
		19									19				
		20									20				
		21									21				
		22									22				
		23									23				
		24									24				

Fig 3.3 Exposure sheet

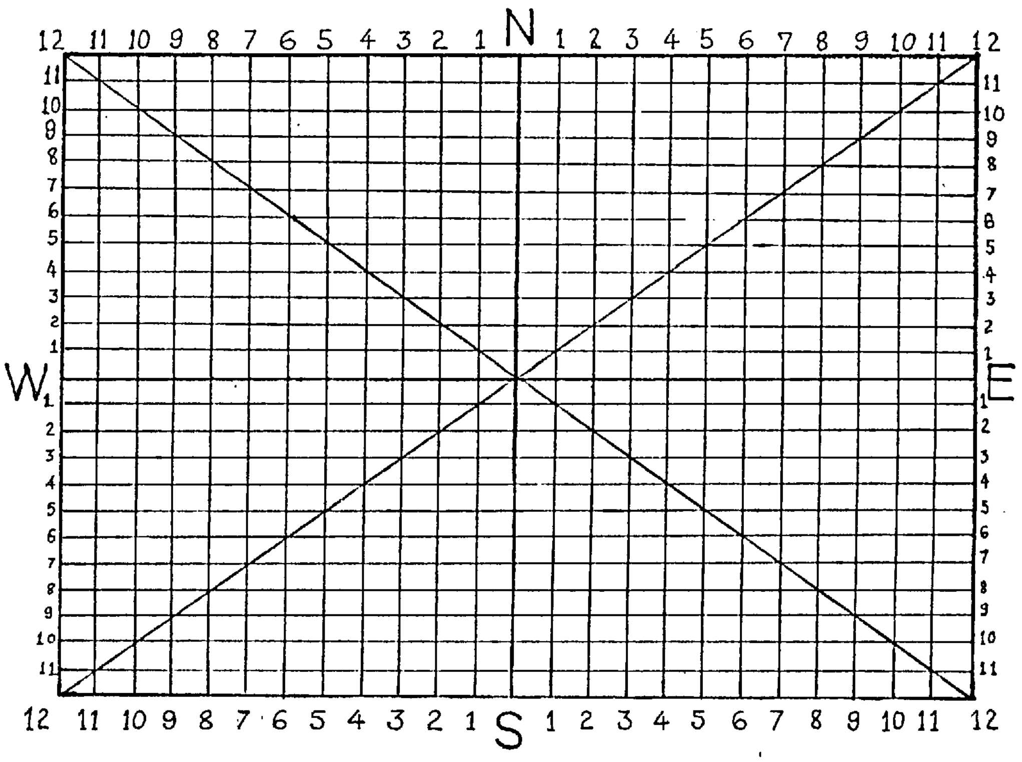


Fig 3.4 Standard 16mm field guide

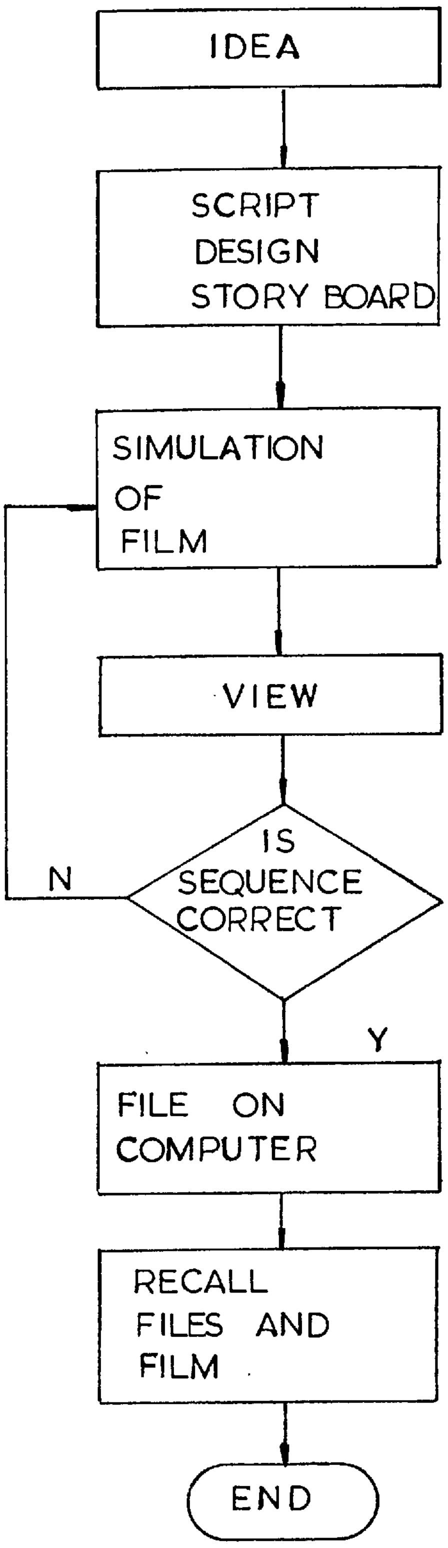


Fig 3.5 Animation with computer

3.3 System Requirements

What is required is a readily available and inexpensive system specifically designed for the purpose of simulation which permits on site programming, previewing and production of good quality computer animated films automatically recorded on devices linked to the computer. As little time as possible should elapse from the initial concept to the actual viewing of the film and the operator should have close control over the entire process.

The system should have fully interactive capabilities enabling direct conversation between man and machine with immediate feedback. The operation should be interactive and iterative with successive refinements being introduced at every step by editing the data and updating the parameters. This can be done through a set of suitable manipulation routines (Fig 3.6). The system must also provide powerful input and output facilities to simplify the task of data preparation and viewing of results and produce a compatible output that can be used by the industry in its present form without the need of intermediate processing.

The obvious choice for this purpose is a hybrid graphical system which satisfies both the requirements as a draughting and calculating machine. It is extremely advantageous to use hybrid techniques by taking advantage of superior analog curve generation capabilities resulting in small storage requirement for a digital computer and fast display time. Hybrid systems are superior but expensive and seldom do the advantages compare with the cost.

Purely analog systems have had considerable success, particularly in processing television information. Analog systems are particularly powerful in image processing because displayed information, especially random produced sketches, are analog in nature. They permit quick and direct interaction between user and machine.

Weighing against these factors are the greater accuracy and storage capacity of digital devices. Digital computers are easily available, more flexible and cheaper than analog devices and often it is the cost of analog peripherals and digital-analog converters that represent the larger proportion of the initial cost of the system. Also the future of any form of information processing, including image processing, seems to be in digital techniques.

The choice of a digital computer will depend on two factors: performance and cost. From a display point of view a stand alone system based on a mini-computer is preferable to a large time-shared system, unless the animation terminal is given maximum priority and full attention from the executive for brief intervals of time, or the executive must assume full responsibility for generating some of the display functions. Alternatively one could have a large system with smaller intelligent graphics terminals in full control of display and interaction with the user, and the larger processor only being used for major data processing. With current price trends it is in fact more cost effective to have a dedicated system than to try to share the resources of a large machine not originally designed to respond at the speeds required in interactive graphics. Prices of mini-computers have fallen continuously since their introduction. Prices are expected to bottom out in the near future but minis are already extremely competitive compared with large main frames, especially for on-line applications, where they provide superior response time and present fewer interfacing problems. But probably the main advantage of minis over main frames is in the less restricted user environment where they can operate.

The selection of hardware should be based on a number of considerations:

1 - Reliability - select an established well tried system that is expected to cause least hardware trouble.

2 - Good servicing and software support to reduce minimum down time in case of breakdown.

3 - Equipment well established enough to be regarded as standard or to fall within some widely adopted standards. This offers wider choice of peripherals and accessories and easier interfacing.

4 - Comprehensive research and development plans to allow for future expansion and upgrading.

5 - Flexibility to accommodate any advancements in computer equipment and peripherals, new interfacing techniques and especially new audio-visual equipment.

The basic requirements for a general animated graphics system can be summarised as:

1 - A self-sufficient and inexpensive system.

2 - An ability to draw and refine pictures defined individually or built up from components.

3 - The availability of immediate visual feedback.

4 - A comprehensive library of manipulation routines.

5 - The ability to define picture changes and dynamics of picture changes via a set of parameters.

6 - A choice of output media to record results.

These are requirements common to most interactive computer-aided-design systems. The system requirements should be met with flexibility to enable a possible transfer to another configuration. The philosophy adopted should transcend any hardware or particular installation. Although this is not always possible great care and selectivity is required in the choosing of hardware. It is seldom possible to obtain the ideal hardware and even the best hardware can find itself becoming outdated and obsolete. Hardware can be later replaced or upgraded but should only at minimal cost. Improvements can be made on the system by incorporating more advanced hardware. This may represent little practical difference except to impress some users, but which may prove to be an important psychological factor in winning over their approval.

The performance of the system will depend mainly on the level of man-machine interaction. This interface is provided by input and output devices. To make the operator feel part of the system these devices should provide an extension of the user's own abilities. It is essential to remember that an interactive system is not complete unless incorporating the user. The system by itself is open-ended until the user, who represents the closing link, comes into play.

The next two sections describe a number of input and output devices normally associated with interactive graphics. These are shown diagramatically in Fig 3.7.

3.3.1 Input devices

These have two primary functions:

1 - To input graphics data

2 - To enter commands.

A number of devices exist on the market which have the flexibility for both entering graphical data directly from pictures into the machine and positioning accuracy to activate a menu of commands. They are namely the light pen, and the digitising table or tablet. Other devices exist, like the mouse or joystick, but these seldom have the accuracy required for animation work.

The light-pen is a device simple to use and has the advantage of working on the actual viewing surface but for drawing purposes it lacks the accuracy and resolution of the digitiser. It requires a refresh type of display which seldom is large enough to contain the data and display a comprehensive list of commands. Well programmed light-pens normally have a very high response speed and are a delight to use. They are particularly powerful in interactive processes like on line editing. Commands can be entered by selecting from a list displayed on the screen or via a teletype or a combination of both. The keyboard is essentially a command console but can also be used for inputing data in numerical coordinate form.

The digitiser is a very accurate device but it lacks the natural feel of the light-pen. It is ideal for entering data by tracing from original drawings or to digitise accurately from engineering drawings. Digitisers normally only allow discrete points to be input, but a suitably programmed one can allow streams of data points to be read in as the digitising cursor is moved. Commands are normally entered by digitising from a reserved menu command area on the table.

The main problem in animation work is the handling of large amounts of data of a random analog nature by non-technical people. The prejudice that some of these people may have towards computers can only be overcome by providing tools that are familiar. In the case of the designer data input should be easy and free from constraints like drawing with a pencil on paper. For such purposes the data tablet represents the ideal device. A number have lately appeared in the market, some of which allow the use of an ordinary pen or pencil drawing on an actually visible surface.

There are other devices which although not specifically designed for the purpose of graphical data input can also be used to provide such data. Bulk storage media such as magnetic tapes or paper tapes can be used to transfer data obtained from other sources, normally an off-line digitiser, a data generation program in another machine or experimental data obtained on site.

There are other more direct means of data input like optical digitisers, based on video techniques or solid state like charge coupled devices, but these are still at their early stages of development, lacking in resolution, expensive, and tend to produce excessive amounts of data.

3.3.2 Output devices

Two types of special output devices are required for the purpose of animated graphics:

1 - A visual display unit (VDU) to enable direct viewing of data, providing means of visual communication and on-line interactive editing of graphics sequences.

2 - A permanent recording device to ultimately transfer the picture to the medium of photographic film, video tape or harcopy which can but need not use the same mechanism.

Two types of VDU are normally associated with computer graphics: Refresh display Tube and Direct View Storage Tubes.

A refreshed display tube consists of a low persistance phosphor coated cathode ray tube (CRT) where the image stored in a special buffer is continuously redisplayed (refreshed). The refreshed display offers 'real-time' movement and dynamic editing. However to have a flicker-free image the entire picture must be redisplayed many times per second which entails high data transfer rates. With the picture stored digitally, usually in a special store as point and line coordinates, fast logic is needed to convert the digital representation to beam voltages and a high power, fast deflecting CRT is needed to draw the picture. This is normally achieved by having a hardware display processor and an independent display file which contains very simple coded instructions for the display processor. The display speed can be very high up to 20 frames per second but depends on the number of lines per page.

Storing the picture on the face of a Direct View Storage tube eliminates the need of a display memory, reduces the requirements for high speed logic and also the flicker problem is not present. Storage tubes can also be refreshed by setting it in write-through mode and continuously refreshing the screen. This can only be done with software and the display deteriorates very rapidly when the amount of data is increased. Table 3.1 shows the results of displaying polygons on a Tektronix 611 storage tube driven by software in a PDP 11/40 computer with floating point arithmetic processor. The disadvantages of storage tubes are lack of contrast and selective erasure, which precludes dynamic editing. The last disadvantage can be partly overcome by software techniques, but in order to display the new picture with the alteration added to it the whole screen must be erased and the entire picture redisplayed.

Despite the lack of contrast storage tubes are ideal for photographing directly since they do not show any scan-lines or band effects as present with refreshed devices when photographed at high shutter speed. The camera is mounted rigidly and squarely in relation to the screen face and operated by signals from the computer with the shutter activated by a solenoid controlled from the computer. The film is advanced automatically by the camera's own motor or by a separate motor driven by computer signals. To obtain longer sequences each frame can be repeated two, three or even four times without any increase in computer time, but at expense of smoothness in movements, rather jittery, but quite acceptable.

Filming straight from CRT's is unlikely to produce excellent results but does provide a cheap and convenient alternative to microfilm plotters. Microfilm gives a very high quality output, is fast enough to produce in a matter of seconds simple line drawings at rates of several frames per second on 16 or 35mm black and white film or produce a very high quality image with full grey-scales made up of a mosaic of closely spaced dots. One limitation of microfilm plotters is the monochromic output. Colour can however be added by optical printing which is a rather costly process or by recording through colour filters a facility now available in some recorders.

An alternative which meets both requirements outlined at the beginning of this section is to produce a video compatible signal which can be displayed on a standard television monitor and recorded on video tape. Video requires the source data to be in a special raster scan format. Data in raster format is normally generated by hardware in the display processor. The picture is first drawn on a high-precision CRT. This image is then scan-converted via a high resolution vidicon to a video signal which is stored in a particular section of a video tape or disc. This video recording must subsequently be edited by a video editing machine to yield the single frame information. The scan conversion process can also be achieved by software techniques, a slow but more flexible alternative, and sometimes the only one when the output device allows write mode only. Pictures generated in raster format also have the advantage of being suitable for processing by three-dimensional hidden-line algorithms based on the scan-line property (see section 6.6.1). Video has the advantage over photographic film in that it can be mixed with ordinary televised images, be easily processed by modern video facilities, like electronic colouring and editing, and can be made PAL compatible for broadcasting purposes. If results are required in video format tele-cine can be used to transfer film to video. It is always preferable to convert film to video than vice-versa because of the superior quality obtainable from the first one.

New devices have been recently introduced which enable digitally coded data to be displayed directly on standard television monitors. Examples of these are Digital Equipment Corporation's VT30, Nuclear Enterprises Ltd 9062 Colour Monitor Driver, Process Peripherals G.L. Ltd Series 114 Multichannel Computer Graphics System and Ram.tek GX 100 Graphics Display. The teletext facilities currently being developed by the BBC, IBA and GPO-also use digitally coded signals (14). The pictures are presented in dot matrix form on the monitor, the matrix consisting of one of a user defined set of characters stored in a special solid state RAM memory. A maximum number of eight colours are normally permitted and these are coded as foreground and background colours for each character. The device controller scans coded data, either in core or in another RAM memory and extracts the corresponding matrix character from the list to display it in the correct location in the selected colours. The main limitation of these devices is the poor resolution, normally restricted to 280 x 512 bits per page, which results in a very rugged appearance.

For hard-copies on paper,or cel,pen-plotters are used. For combination with conventional animation single static views should be plotted on standard animation cels which can be used without the need for any intermediate processing. The cels are painted by hand and photographed on a rostrum camera superimposed on to a suitable background. Apart from being transparent, cels have another advantage over paper in that ink can be removed easily from its surface, which allows for final touching-up of the artwork and manual hidden-line removal.

		NO. OF SIDES					
		3	4	5	6	7	8
NO. OF POLYGONS	1	*	*	*	*	*	*
	2	*	*	*	*	+	+
	3	*	*	+	+	+	-
	4	+	+	+	-	-	-
	5	+	+	-	-	-	-
	6	+	-	-	-	-	-
	7	+	-	-	-	-	-
	8	-	-	-	-	-	-

*	No flicker
+	Acceptable flicker
-	Unacceptable flicker

Display rate: Approximately 200 vectors/second

Table 3.1 Display rate of polygons

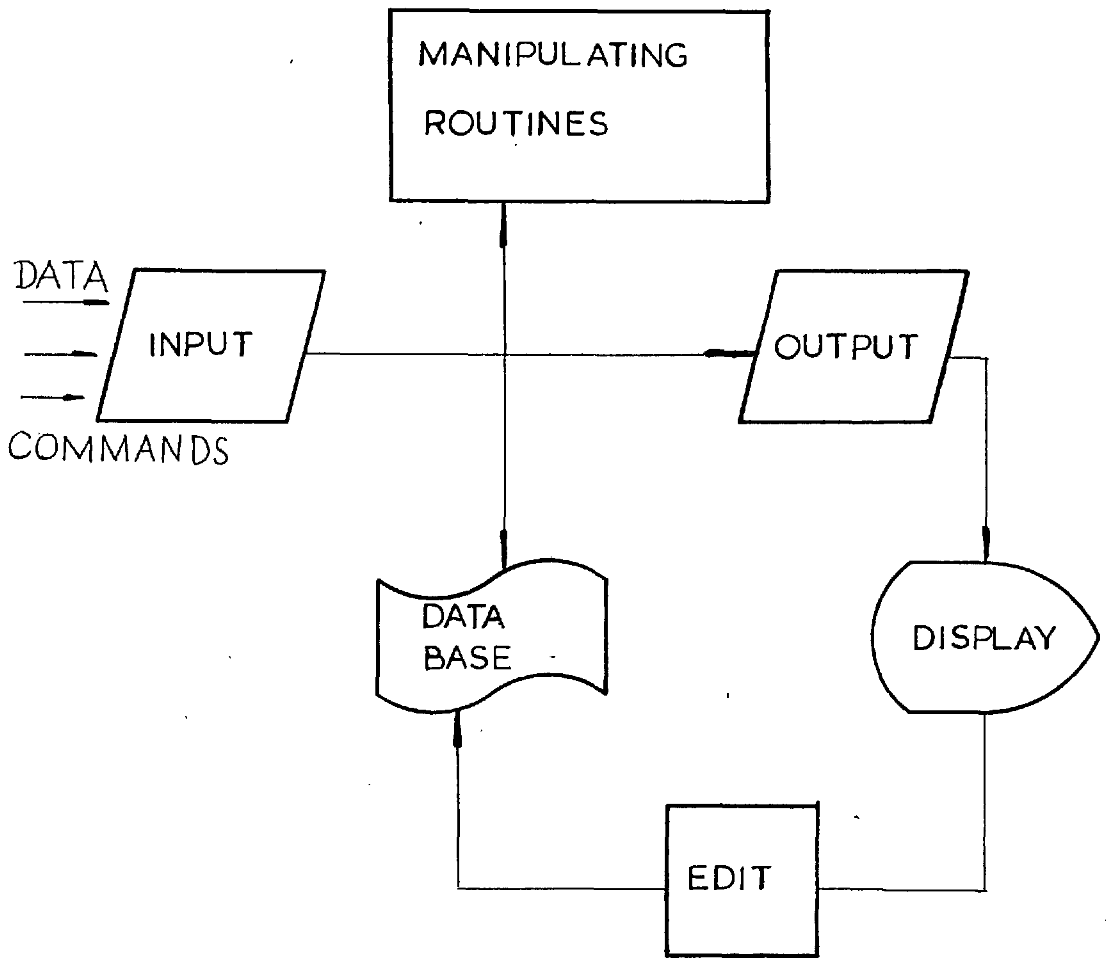


Fig 3.6 Basic operating requirements

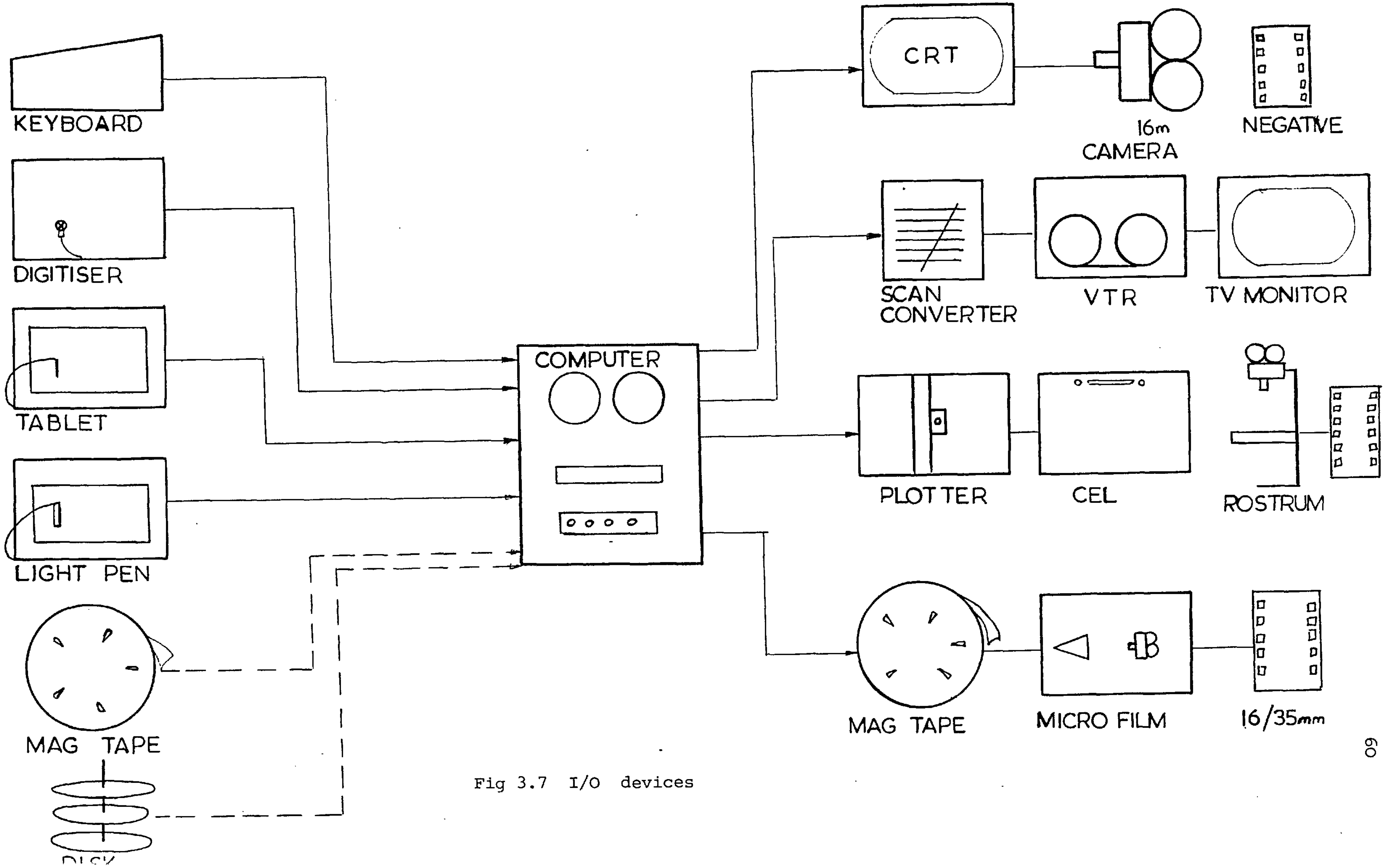


Fig 3.7 I/O devices

3.4 Animation Languages

In the systems described earlier the major emphasis has always been to obtain the formal definition of an animation language by means of which the user could communicate with the system. These have tended to fall into two main categories:

1 - A special purpose animation language with a well-defined syntax, either based on a high-level language or having its own structure based on assembly language. The language possessed its own compiler to read and interpret the commands supplied by the programmer (15, 16, 17).

2 - A software package consisting of a number of callable subroutines supplied in a library form. A program is written to call the required functions, linked with the library and normally run in batch mode. The language most commonly used is FORTRAN, therefore it requires from the user a certain knowledge of the language, particularly the subroutine CALL statement. The arguments in the statement are used to transfer the parameters controlling the action. This type of language evolved directly from general graphics packages written to drive special graphics output devices like plotters, VDU's and microfilm plotters.

Neither type had great success mainly because, contrary to expectation, few users were capable or willing to learn the new language. Syntax which to the programmer seemed simple and easy often proved beyond the comprehension of the common user.

With special purpose languages the language itself proved to be the main obstacle because few users like the imposition of a fixed syntax and are prepared to master it. To remain practical the language either had to be rather restricted, which showed in the lack of variety of work produced, or so complex as to make it very difficult to learn. The languages had facilities to control movements and to generate pictorial data but did not allow free hand sketches to be directly input. If the languages were to be made more comprehensive the vocabulary would have to be expanded and this would increase the difficulty in mastering and using the system.

The second type has had some success, especially in mathematical films, but does assume some preliminary programming knowledge on the part of the user. It is ideal for the scientists intending to produce their own research or teaching films, but has little scope for the layman.

The major drawback of these languages is the lack of interaction. They were designed to be run on large computers operating in batch mode. It is a very frustrating experience to be unable to preview a film and introduce minor but vital modifications during the process of its creation to end up with an imperfect film after long hours of preparation. The addition of macro definitions and increased flexibility to allow the user to tailor the language to his own needs would probably make them more acceptable but would further restrict them to users with programming experience.

In the design of GAGS, the system described in this thesis, the use of a special purpose language, whether a new one or a well established one was rejected from the start. This not only relieved the need to define a formal language, but also the effort of implementing one. After all, animation is used because images are believed to convey a message better than words, thus it is a mistake and a contradiction in itself to try to use words in an image creation process. The pictures themselves are as often as possible used for communications purpose and where words are needed it is only to present the user with a list of options for selection. A special purpose language is really only convenient to the programmer anyway, for to produce drawings one does not need a complex vocabulary and syntax but just a pencil, paper, a bit of skill and a goal in mind. The use of a language would take an initially trivial task to a level of unnecessary complexity.

4 THE CADMAC-11 SYSTEM AND ITS COMPONENTS

The system described here (GAGS) was implemented on a CADMAC-11 system. CADMAC is a low cost fully interactive computer-aided-design system built around a mini-computer (1). It is self-sufficient to operate in a stand-alone mode but can be linked to a larger computer for greater computing power.

Fig 4-1 shows a schematic diagram of the CADMAC animation system.

The system comprises a Digital Equipment Corporation's PDP 11/40 mini-computer of 32k, 16 bit word core capacity, a DEC writer, a high-speed paper tape punch and reader and two disc units with 1.2 million words each. The software is based on DEC's Disk Operating System (DOS) monitor possessing a FORTRAN compiler. The assembly language of the PDP-11 is MACRO-11.

The computer is interfaced to the peripheral devices via a CAMAC (18) crate, housing 24 printed circuit type modules. The modules act as controllers and are peculiar to the computer and peripherals used. This offers great flexibility and will permit expansion of the system at minimal cost.

The main peripheral devices are the reversible digitising/ plotting table and the Visual Display Unit (VDU).

The digitising/plotting table consists of two surfaces, one above the other. The one on top is for the digitised input with a magnetic cross-hair pen, and the lower one for output where hard copies are produced. The top surface is hinged at the rear of the table so easy access may be obtained to the plotter. There is a choice of output pens, mounted on a carriage running in the y direction on top of a gantry moving in the x direction. Signals from the computer through the CAMAC crate, or the field set by a high-frequency current in the digitising pen cause servo motors to drive the carriage and the gantry, either for plotting or digitising. The digitising pen normally contains up to 8 buttons and when one or a sequence of buttons is pressed a command is executed by the computer.

On the table there is the menu which is a reserved area divided into a number of squares, each square identified by a name or number. A point digitised within any one of these squares causes a routine to be loaded and executed within the computer.

The requirements for visual display are met by either a Tektronix 611 storage tube or a standard television monitor with a scan converter. The 611 storage display unit is the one more frequently used. It has a display area of 21 x 16.2 cm containing approximately 1000 units along the horizontal axis and 800 along the vertical. The origin, i.e. the coordinate position (0, 0) is located on the bottom left hand corner of the screen. The scan-converter is a Tektronix Type 4501 - scan conversion unit. A more advanced one is currently being developed.

The output of the system for animation work is either on film or cel. The former is achieved by either filming directly from the screen or by the use of the CALCOMP 1670 microfilm plotter available at the University of London Computer Centre. The microfilm plotter provides very high quality pictures with up to 30 grey scales. The disadvantages of the system are high cost, lack of perforated 35mm film, only 16mm or unperforated 35mm are available, and sometimes poor registration and poor processing performance. For greater details of photographing the Tektronix 611 storage tube refer to Appendix B.

The system also incorporates a high-speed flat-bed plotter where hard copies on paper or plastic cels can be obtained. The table can operate on-line or in background mode, by plotting only when the system is idle and allowing digitising or data processing to proceed simultaneously. Appendix H contains details of background plotting.

The electrical sketchpad and pen were developed in collaboration with the University of Essex (19). The device uses a current division method to determine the coordinates of the pen position. The pen is wired to a control box to inject a current at the point of contact on a special conductive surface. This allows very accurate positioning with a resolution of approximately 10 bits which is comparable with the screen. The performance is excellent with no noise problems. A new larger version is being developed which does not require the use of special conductive paper.

The CADMAC-11 is a well-tried and established system with a reasonable suite of standard programs to interact with the peripherals. The use of it in animated graphics further proves its versatility.

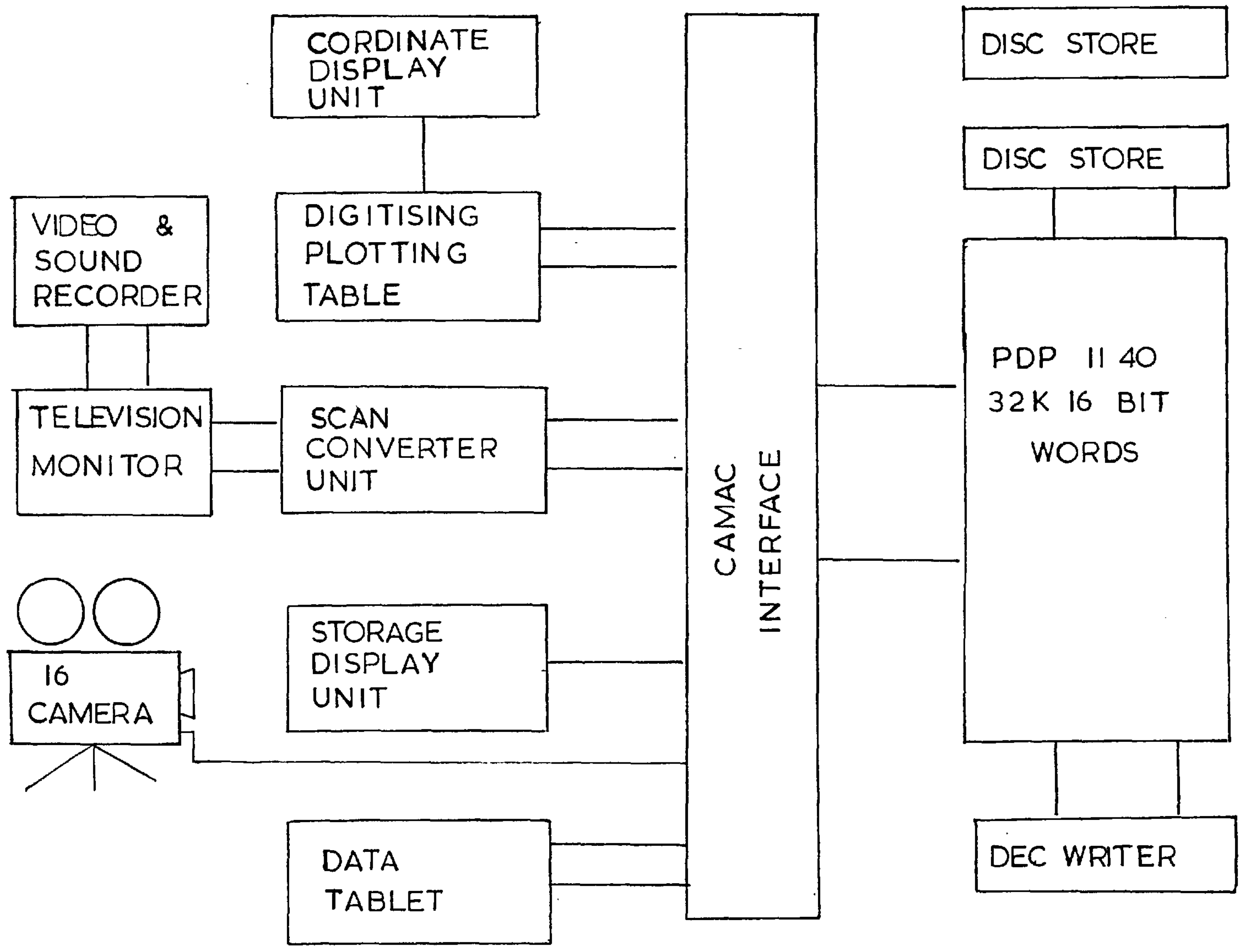


Fig 4.1 Block diagram of the CADMAC-11 system

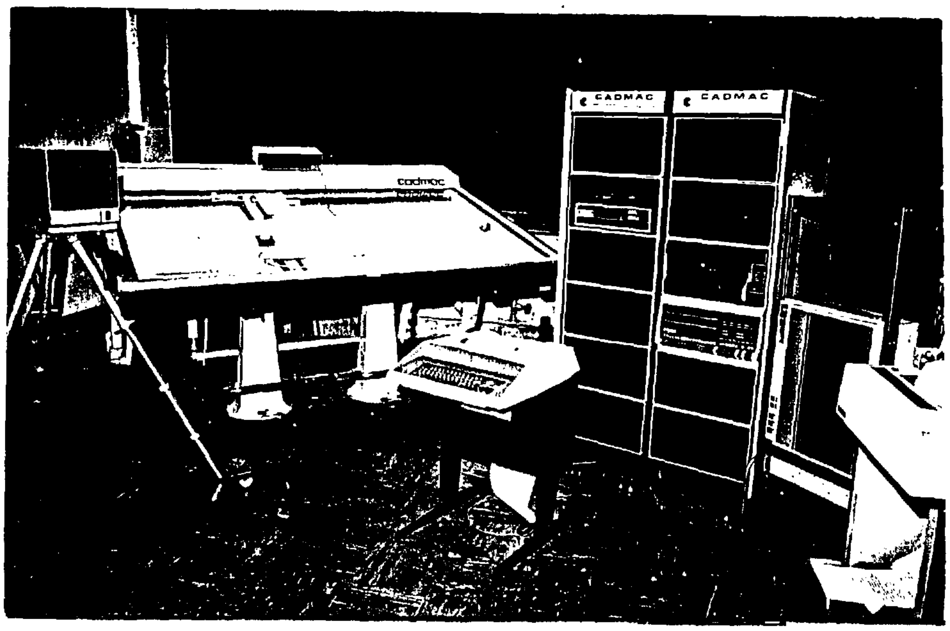


Fig 4.2 The CADMAC-11 system

5 GAGS-2

GAGS-2 (General Animated Graphics System) is a two dimensional system based on the original computer-aided-design software developed to operate on the CADMAC system (20, 21, 22) and incorporates some of the original graphics facilities for data manipulation. However many of the input and output facilities were found to be inadequate when required to handle large amounts of pictorial information as in the case of animation. A number of special graphics input facilities have been added to facilitate the construction of pictures. The output stages were made more efficient by using faster algorithms to generate pictures at a higher rate and to accept parameters in animation terms.

The application software includes a suite of special functions used for the manipulation of the pictures. Although the data is strictly two dimensional many of the effects obtained appear tri-dimensional.

The software description pertains to a particular implementation on a PDP 11/40 computer. The algorithms are general enough to make it transportable to another system with relative ease, however, the parts corresponding to data input and output and data storage are confined to a particular hardware configuration.

5.1 Programming Considerations

The GAGS system was written based on the FORTRAN overlay facility specially written for the CADMAC system (21).

The system is modular in concepts, each module corresponding to an individual overlay stacked in an overlay file. There is minimum interaction between modules to reduce interdependence. This facilitates debugging and upgrading. Each module operates independently on a common data base. For communication purposes common blocks have been set up in the core resident area of the program so that information is not lost when a new program is overlayed in core. Sometimes however it is necessary to transfer data from program to program, permanent disc files can be used for this purpose. Thirteen fixed length high-speed scratch files are available on disc but more files can be defined by the programmers as required. The existence of such files should be transparent to the user (Fig 5.1).

From the user's point of view, the system behaves like an applications graphics system. It is however open ended enough to allow an applications programmer to modify or add to the program to suit individual requirements, without need to involve in design of data structures or I/O programs. The size of the system has been kept small enough to enable the whole concept to fall within the grasp of a single individual so that, if required, fundamental changes to the system can be made.

The system was written in a conversational mode using a user-oriented policy. Questions and answers are always stated in simple clear mnemonic English or in animation or cinematographic terms.

This allows the user to communicate with the computer thereby giving flexibility to his task. It contains convenient ways of specifying and moving objects mathematically and ways of representing the object in a viewing plane. It should allow the user to imagine himself as a designer working on a drawing board, a scientist at the testing rig, an animator editing a film sequence, and a cameraman with commands for panning, zooming, rotating, dissolving, fading, etc.

It operates in an interactive, iterative manner where each process can be continuously monitored until the desired results have been achieved. Interaction between user and machine is however kept to a minimum and on an elementary level. Restricting the capabilities of the system can have its advantages by reducing mistakes and confusion. The programs were written trying to anticipate what information would be needed and provide these by default whenever possible, or by automatically directing queries to the user. The approach would be effective in so far as it allowed the user to continue his work without interruptions and without concerning himself with details of computer processing or trivial matters which the computer can be made to sort out by itself.

All parameters for a sequence are requested by the computer on the teletype or on the screen so the user need not prepare any data or even remember the order in which data must be entered in the programs. As far as possible such programs are concise and self-explanatory. The user only has to read limited amounts of instructions. The procedures in the programs are transparent to the user who is only required to be aware of the facilities available and the function of a few buttons and tools. Answers should preferably be in the form of

YES or NO acknowledgement except when parameter values are requested.

Whenever possible input is free format. This removes the need to pay attention to decimal points, leading or trailing zeros and counting columns when dealing with data input. Designers are used to imprecisely defined numbers which are a frequent source of errors. Provision for typing errors and, for people familiar with the system, option to suppress the operating instructions are also incorporated.

The system provides the facilities for the construction and manipulation of static pictures, to represent and specify picture changes and dynamics of picture changes. User should be able to select from a wide range of manipulation functions to obtain the desired results. By changing one or more parameters independently a new set of results incorporating these changes can be obtained. Programs transform pictures into sequences of visual images, storing and retrieving these pictures from auxiliary memories facilitating both their playback for immediate viewing and transfer to permanent recording media.

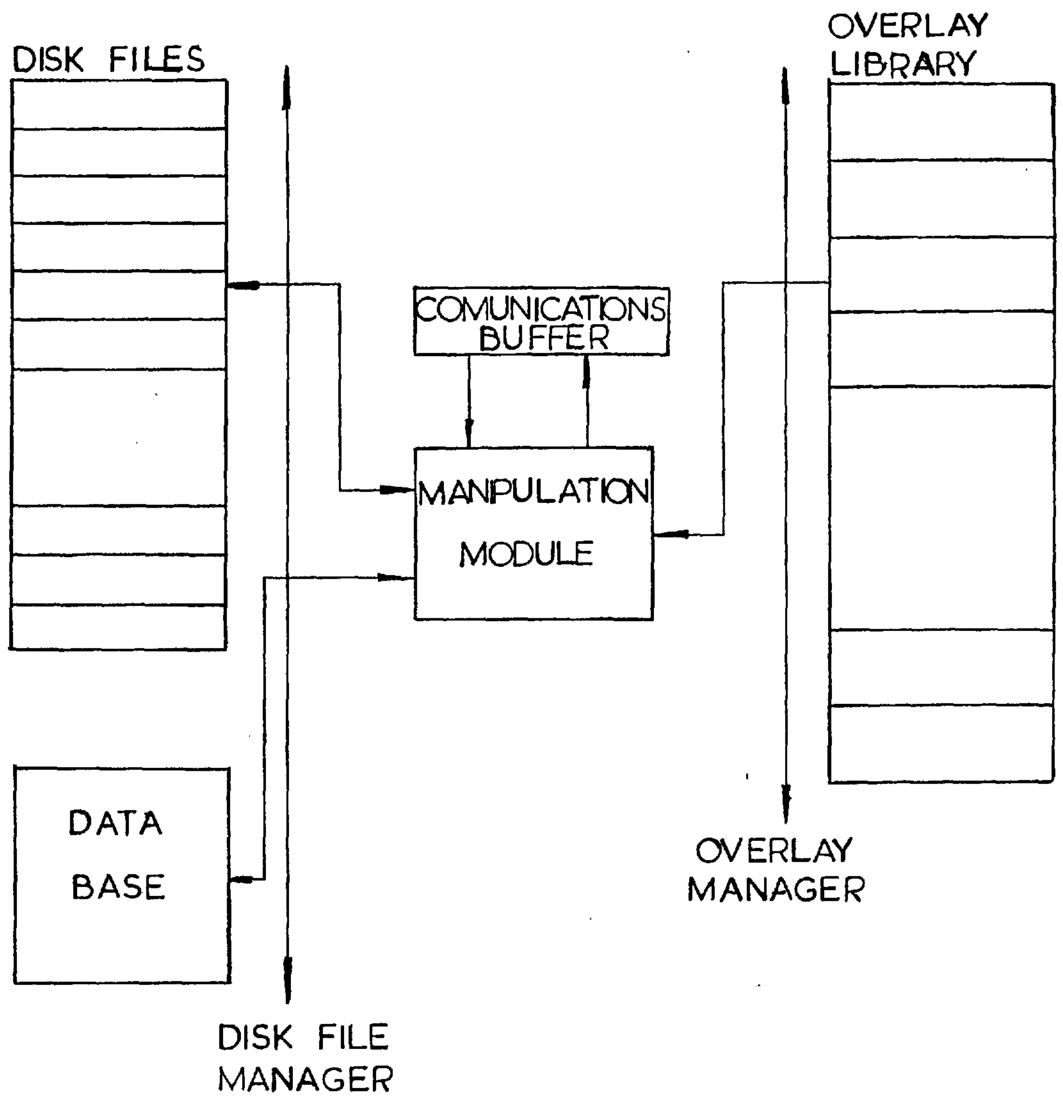


Fig 5.1 System interaction

5.2 Operating Modes

The system is designed to operate in two modes: a drafting mode and a command mode. Sometimes these two functions overlap during the operation of the system with data and command being entered simultaneously but normally they are well differentiated by the physical location of each operation.

The pen is the main input device by means of which the system is monitored and controlled by the user. It is also the device by means of which the pictures can be drawn into the computer without interruption. The computer records the essential information from the act of sketching by sampling the status of the pen at regular intervals and storing the position in coordinate form. Free hand sketches can be done crudely as the computer instantaneously straightens lines, interpolates and adjusts scales (these are options that can be set by the user). The pen can also be used to make alterations and additions to the picture.

Input can be via the digitising table or on the data tablet. When the input is being made via the table the command area is on a menu situated to one side of the digitising area (Fig 5.2). The menu is an area divided into a number of 20 x 20 mm squares. Each square is identified by name or number and corresponds to a specific command. A point digitised within any one of these squares causes a program to be loaded and executed within the computer. In GAGS-2 there are 100 command squares, 40 reserved for general use and the remainder for applications programs. The remainder of the squares are divided into 3 types: 60 levels, 100 files and 40 symbols.

For input via the tablet a thin strip of the drawing area is allocated as a switch so by touching the corresponding area the tablet may be switched from drafting to command mode, transforming the surface into a command menu or vice versa. The menu on the tablet is identical to the one on the table but more restricted in size. If the tablet receives no contact for a certain time control is automatically returned to the table. Short breaks in the sketching act are regarded as broken lines so control is retained by the tablet.

Additional commands may be entered via the keyboard. The user can interrupt a process from the keyboard by using the Disk Operating System command structure. Certain types of data like alphanumerics must be entered via the keyboard. Messages from the computer are normally logged on the keyboard but can also be displayed on the screen. Replies can be via either the keyboard or the pen.

The pen also incorporates a facility which enables entered data to be interpreted as a command directive or reference point. This is done by digitising the point using one of the several buttons on the pen. There are eight buttons each with a different function:

1 - Store coordinates of pen position. If on menu obey menu command.

2 - Break line - start a new digitising sequence.

3 - Macro - enables a series of macro manipulation facilities.

4 - Window - scales pictures to fit a given viewport.

5 - User defined button - a floating assignment button which can be made to correspond to any menu command. May be redefined at run time.

6 - Control drafting mode - restricts the angles at which lines may be drawn.

7 - Find mode - locates and positions pen accurately on nearest point.

8 - Drive mode - allows accurate dimensional drafting in cartesian or polar coordinates.

The commands in the system follow a tree-like structure. A particular example is shown in Fig 5.3. As the system interacts with the operator it requests instructions or can be interrupted by a command. The requests are normally presented as a list of options on the screen from the which the user selects one. The process is repeated for each step until it is completed from which it exits then or may be restarted again. Fig 5.4 shows a typical operating sequence.

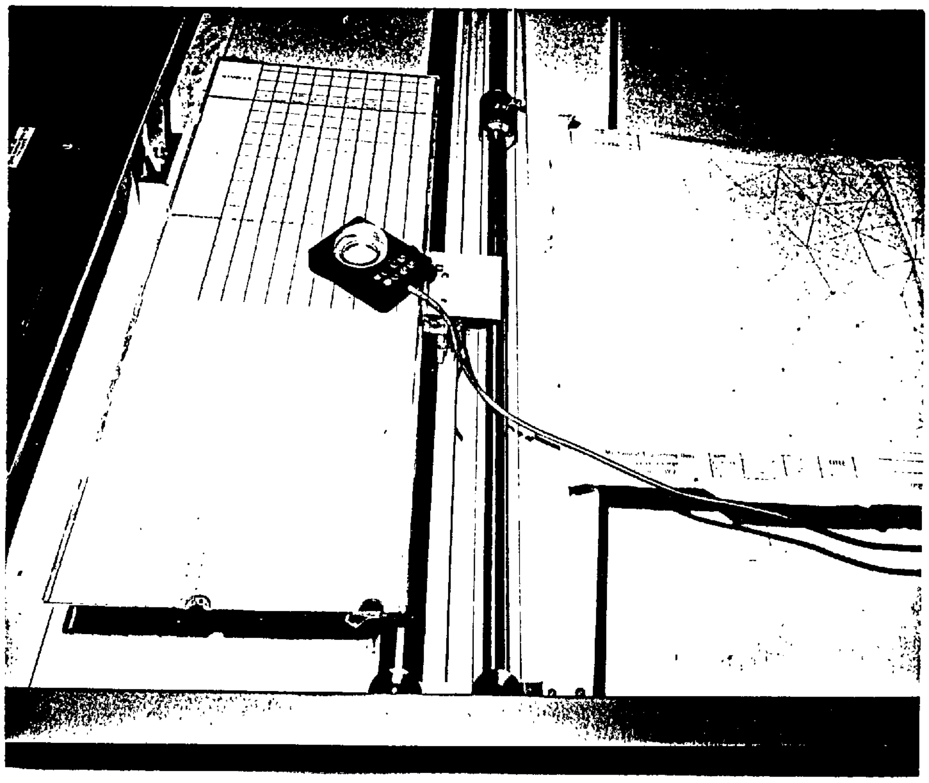


Fig 5.2 Digitising table with menu

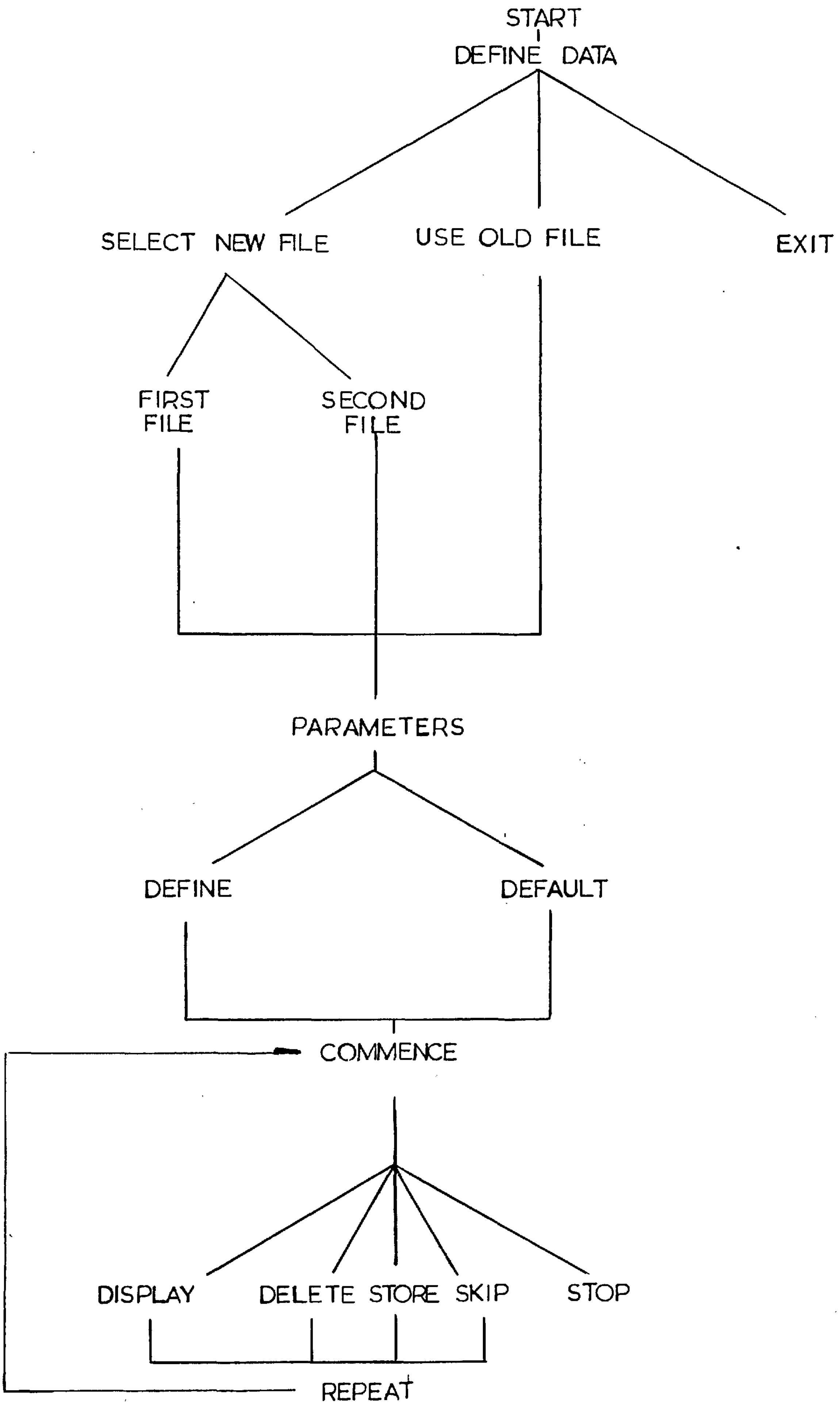


Fig 5.3 Command structure

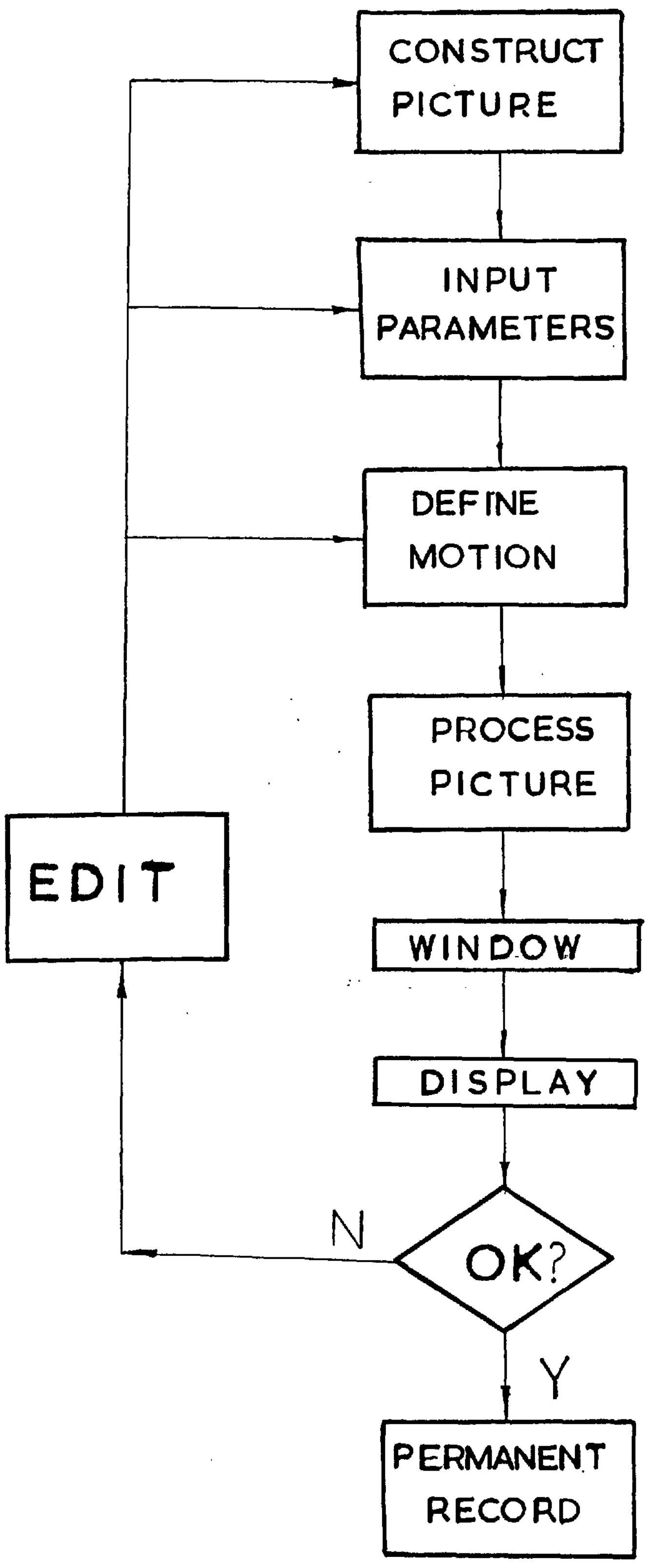


Fig 5.4 Operating sequence

5.3 Data Structure

The system uses a hierarchical structure of data to facilitate the manipulation of pictures expressed in terms of pictorial primitives (Fig 5.5). These pictorial primitives can be used singly or combined to create pictures of increasing complexity (Fig 5.4). The point represents the basic element and can be used to define the end points of a straight line, a reference point on a curve or a point on a matrix. Lines are made up by joining the end points or can be defined by a series of points on a matrix. Picture elements are made up of lines. It will be seen later how these elements can be defined as symbols or macros depending on some common geometrical or structural property.

5.3.1 Data records

Data is stored in the form of $\left[I, X, Y\right]$ records. I is the data pointer which qualifies the contents of X and Y. These can be parameters or reference data but normally consist of the cartesian coordinates of the position vector of a point. Coordinates are stored in terms of table dimensions with an accuracy of up to one tenth of a millimetre. The useful length of a table is 0.79m therefore the maximum resolution is 7900 addressable points along the X-axis.

Data digitised from the table is stored in the form of individual points each in a data record. The points are linked by the information contained in the I term of the data record. Strings of data belonging to the same unbroken line, called elements, are separated by delimiters also stored in I form. Table 5.1 describes the meaning of the I codes. Data records can be individually random accessed but it is normally done in sequential address mode since this results in much faster processing.

Blocks of data which contain common features can be defined as macros which are an integral part of the data structure. Macros will be dealt with in greater detail in the next chapter. For classification purposes complete pictures which can be manipulated as one single unit are called picture components.

5.3.2 File management

Picture components are stored in files which are reserved blocks allocated on disc or magnetic tape for the purpose of data storage. There are two types of files (Fig 5.6):

Temporary - used as scratch files for temporary storage of data during the processing stages. The contents of these files are lost when the system is shut down.

Permanent - named or numbered files which contain the final data and are not destroyed when the operation of the system is terminated.

The data structure is identical for both types of files, the only difference being in the way they are accessed and managed.

Thirteen temporary files are available on the disc. These files can only be accessed by program interaction. The user cannot assign them directly. Two of the temporary files are reserved for use by the system: one for temporary storage of data during overlay swapping, and the other is known as the Workspace where data being currently digitised, displayed or processed is stored. Although this is a temporary file, the contents of the file may be totally or partially recovered after a system breakdown. A third file is also reserved for a special purpose: it contains the data used to generate bold type alphanumerics as to be described in section 5.4. The remaining ten scratch files are free to be used by the programs for temporary storage or for data transfer between overlays.

Permanent files can be varied in size and number and are only restricted by the size of the bulk storage device. A disc cartridge can hold up to nearly 192,000 coordinate pairs. There are two types of permanent files: user-defined and reserved. Files can be named or numbered for identification purposes. User-defined files are always named, which is essential for teletype interaction. User-defined files are used to transfer data from one system to another or to store data in formatted ASCII.

Reserved permanent files belong to a special storage zone pre-allocated on disc. Each reserved file can be made to correspond to a square on the filing area on the menu. Files linked to the menu can be assigned directly by the user. The present configuration allows a maximum of 100 files to be accessed by user and the remainder can be accesssed by program.They are normally referred by number but external labels, which have no reference within the system, may be appended. The size of these files are fully adjustable depending on the amount of data.

Each file may contain one or more picture components. If the file is single it only contains one component, if it is multiple then each component is separated by a level delimiter. This is normally when pictures belonging to the same area on the frame but corresponding to different cel levels are stored in one file. Up to 60 levels are allowed in the system but seldom more than 5 are likely to be used.

Picture components stored in several files are concatenated to form a frame. A frame may contain one or more levels. Two files with the same number of levels in each when assembled give a picture with the same number of levels as in the individual files. Fig 5.7 shows two files with three components each belonging to three separate levels. When the pictures are assembled a frame is formed on three levels with two components in each level. The advantage of using multi-level planes allows zones of the picture to be displayed and edited individually.

As the amount of data increases a proper cataloguing system becomes essential. These can be divided into motifs-characters, lettering, three-dimensional or classified under users' names. Individual users may reserve their own disks, owning a disk in the same way as they would possess a video cassette, a film or a portfolio.

5.3.3 Macro facilities

Some drawings used for basic input often possess many features in common. For instance, many elements show some form of symmetry, or have recurring features like circles or squares and two key frames have only slight differences in detail between them. In most scientific simulation the basic model remains quite unchanged, and may be used again and again.

Picture elements that have common geometric properties or are used repetitively can be defined as macros. There is no restriction to the size or complexity of a macro. Two types of macros are available:

1 - System generated - these are basic geometric shapes like circles, squares, arcs which can be generated by mathematical equations. They are also known as symbols, but are stored in the same form as macros once they have been created. A symbol can be used to generate another symbol, e.g. the conformal mapping of a circle will give an ellipse.

2 - User-defined - this is a much more powerful facility which allows the user to define any picture as a macro. The picture is digitised and stored in a permanent file. Any data contained in a permanent file can be defined as a macro. Once defined a macro can always be recalled as required, scaled, rotated and accurately positioned.

When a pre-defined macro is required the user evokes the macro handling program which then asks for scaling, orientation and positioning parameters. It is then added to workspace, where it is stored in the same format as the other data, but with delimiters indicating start and end of each macro. Macros may be nested, i.e. large macros may contain point data plus one or more smaller macros, (see Fig 5.8).

The data contained in macros can be processed in the same way as the rest of the data or may be regarded as a single block of data. This enables editing facilities within macros or editing of macros. Therefore lines within macros can be individually changed or the whole macro can be deleted, rotated or translated.

For system generated macros there is an option for storing them in symbolic form, in terms of parameters and pointers referring to the symbol generating program. This represents considerable savings in storage but imposes restrictions in the manipulations of the data. System macros consist of basic geometric shapes, primarily squares, polygons, circles and arcs. Using geometric shapes can save a lot of time and effort but are restricting in style and by no means should entire pictures by made up of symbols.

ICODES		Contents of X	Contents of Y
0	end-of-file		
1	start line	X coord	Y coord
2	next line	X coord	Y coord
7	level	level no.	
9	dummy		
11	same as 1		
13	start symbol	symbol no.	
14	symbol data	XSYM	YSYM
15	symbol bounding rectangle	XLIM	YLIM
16	end of symbol		
19	same as 14		
20	pause		
21	line type	line type no.	
22	pen number	pen number	

Table 5.1. I-codes in data records

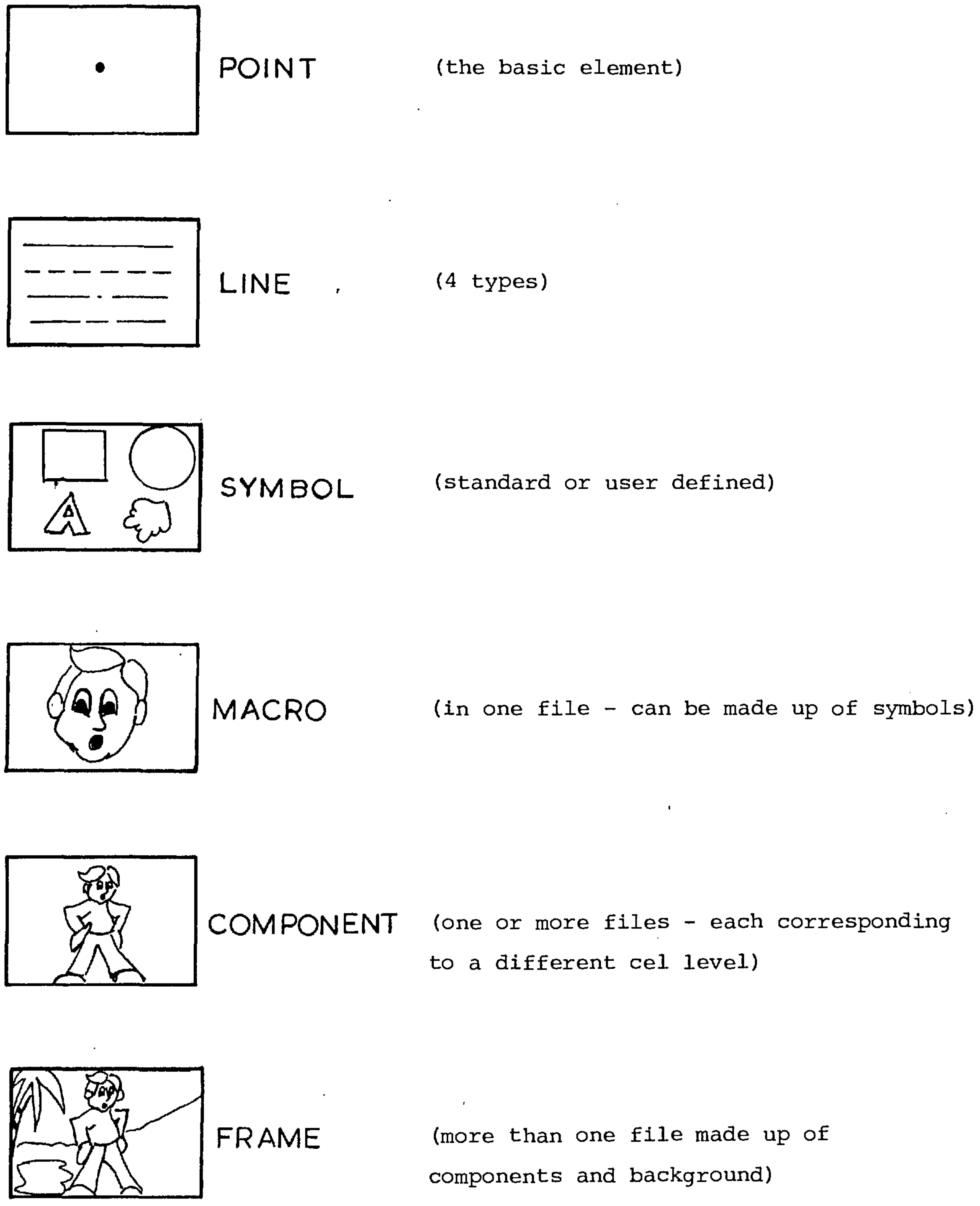

POINT
(the basic element)
LINE
(4 types)
SYMBOL
(standard or user defined)
A
MACRO
(in one file - can be made up of symbols)
COMPONENT
(one or more files - each corresponding to a different cel level)
FRAME
(more than one file made up of components and background)

Fig 5.5 Picture hierarchy

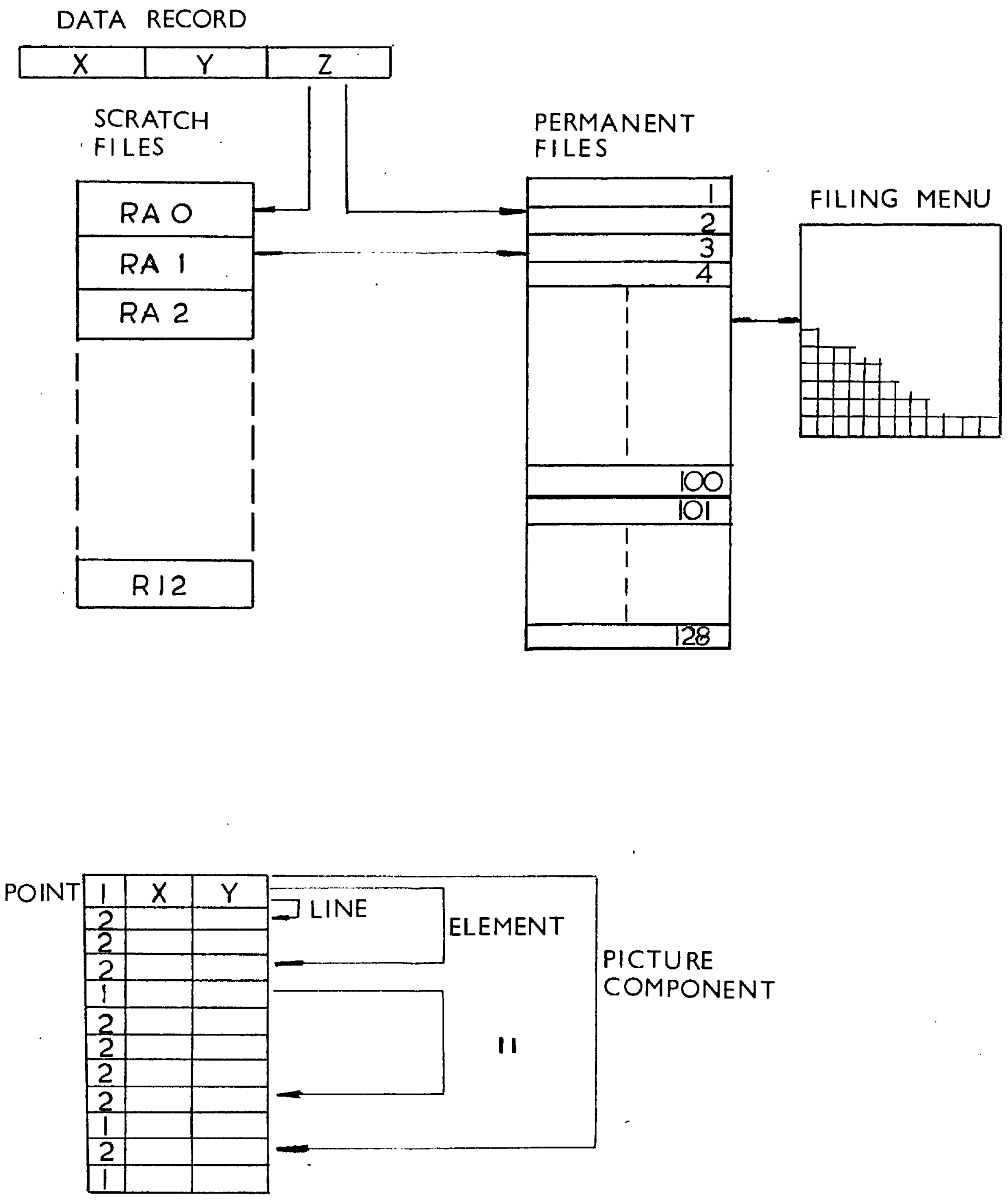


Fig 5.6 Data/file structure

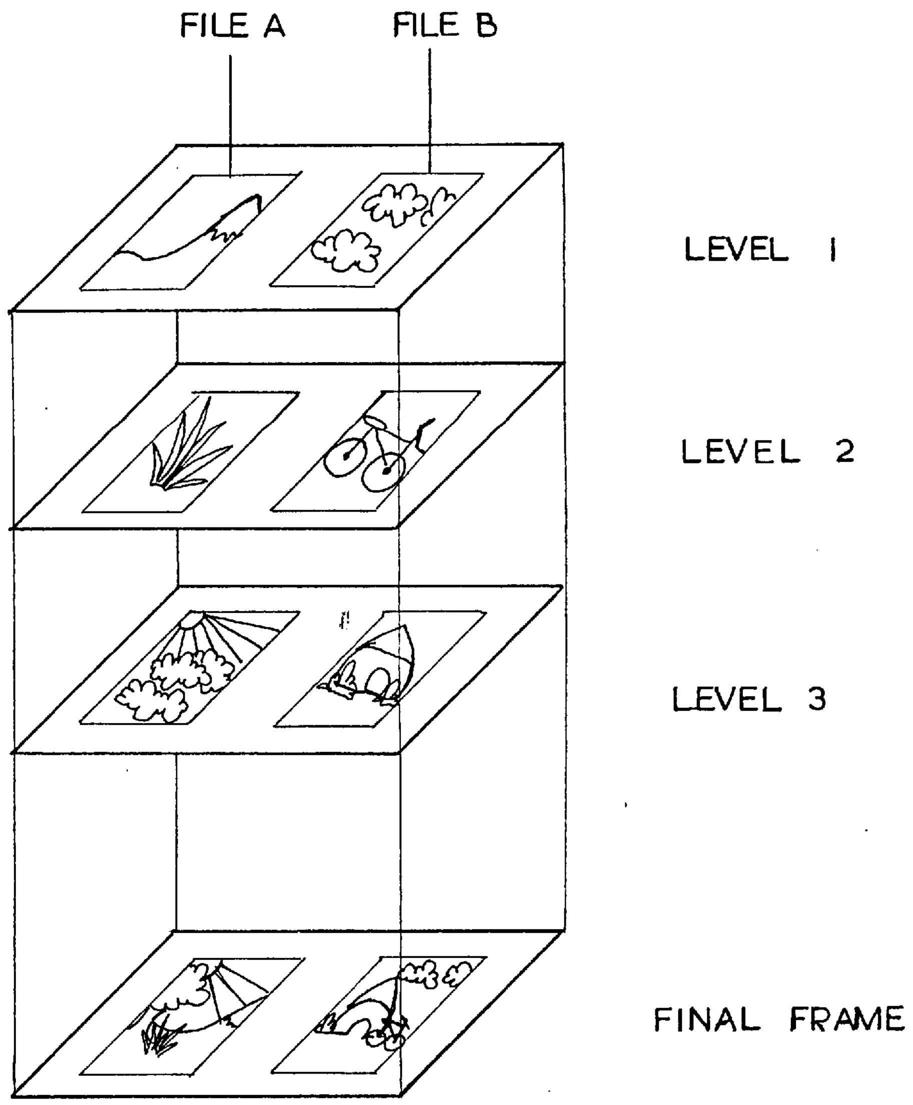


Fig 5.7 Components on three levels

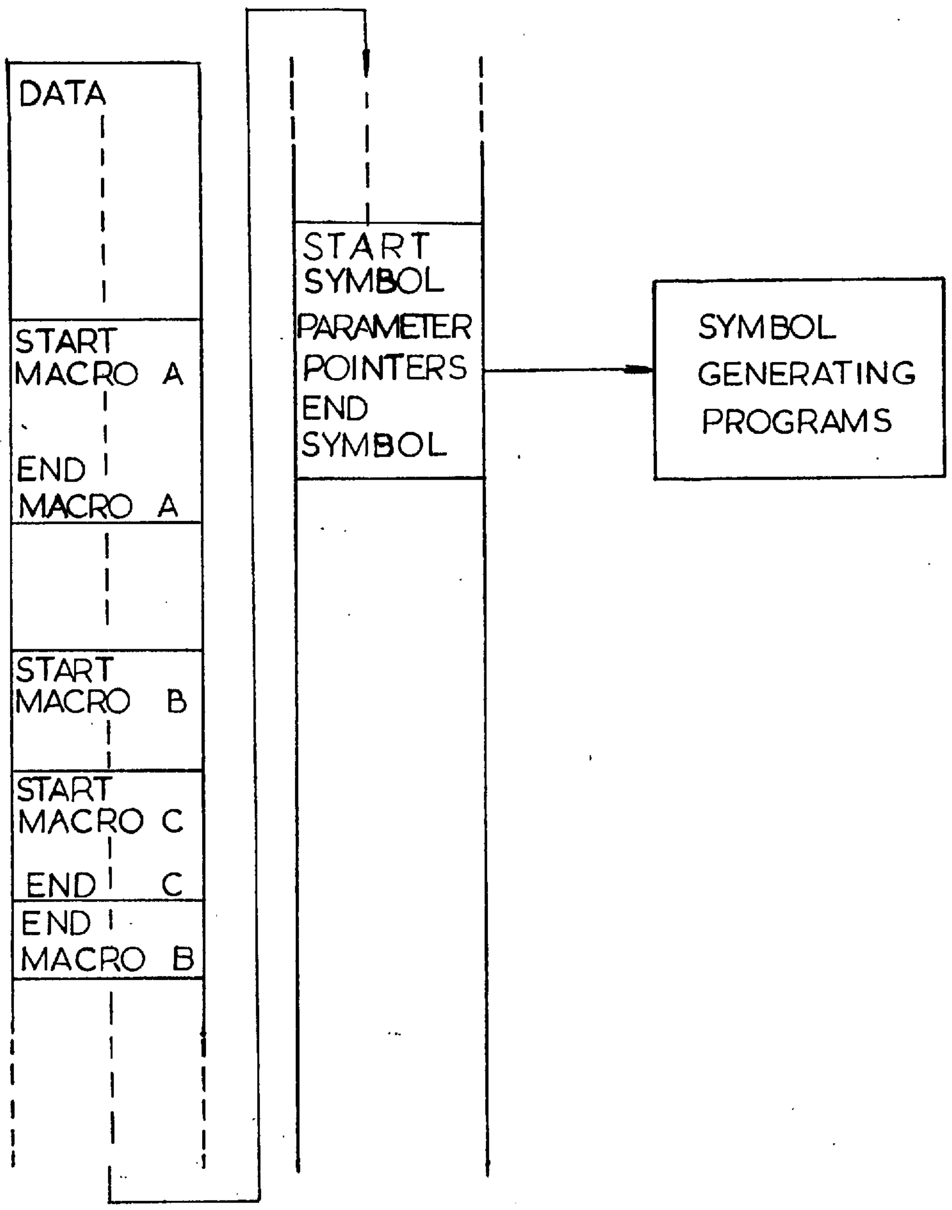


Fig 5.8 Example of file with macros

5.4 Picture Construction

As described earlier, although the point is the pictorial primitive it is more suitable to describe pictures in terms of lines,therefore convenient ways must be found to build pictures by connecting points defining lines. Commands like:

```
enter point A (X, Y)
enter point B (X, Y)
connect A to B
```

are to be avoided as this introduces a level of complexity in a manually trivial task. As points joined by lines are intended, more often than not, points are always assumed connected unless stated otherwise. The input uses linked point priority. Each point entered is assumed to be linked to the immediately preceeding one unless the user specifically acts to break the line to start a new sequence of linked points.

Pictures can be constructed from free hand sketches or from macros. Sketches are input by either digitising discrete points defining the outline of an object with the table pen or by tracing or drawing free hand on the sketch pad. As each line is entered the corresponding representation is displayed on the screen. The table is mapped on to the screen with the pen position indicated by a cross-hair cursor on the screen. The cursor is refreshed in non-store mode on the screen. The actual x-y coordinate values in millimetres are displayed on both the coordinate display unit on top of the table and on the screen. The screen display also gives the position in terms of the polar coordinates. The values of the coordinate can be with respect to an absolute origin or to the last point entered, i.e. a trailing origin.

A number of facilities exist to aid in the construction of pictures. These facilities are activated from either a menu square or pen button as described in section 5.2.

5.4.1 Control mode

This mode of operation is used to ensure that lines lie at fixed angles to the main axis. Either control 15 or 90 can be used. Control 90 is most useful when crudely digitised lines are straightened and adjusted to be at right angles to each other. Control mode is enabled and disabled by pressing button 6 on the table cursor. A message is displayed on the screen when control mode is entered.

5.4.2 Skew control

When pre-set this facility allows skewed data to be digitised with respect to any pre-defined base line. Thus data that is not perfectly horizontal can be rotated about the origin on the drawing. This makes the positioning of drawings quite flexible since they need not lie square on the table.

5.4.3 Drive mode

This facility allows vectors of exact dimensions to be input. It is particularly useful when engineering drawings are used as the source of input. It can operate on either cartesian or polar mode. When in cartesian mode the increments are in units of 1, 10, 100, 1000, multiples or combinations of these. In polar mode the units are 1, 5, 15 or 45^{o}. The mode is enabled by button 8. The positioning is controlled from a patch on the bottom left hand corner of the menu area. Each point digitised within a square adds the corresponding increment to a total which is entered by digitising the centre square.

5.4.4 Find mode

This facility allows input points to be accurately superimposed on existing data, when a point is digitised with button 7. The workspace is searched for points within an area of interest represented by a square of fixed dimensions with the digitised point at the centre. If there are points within the area of interest then the point nearest to the digitised point is stored, if not a message saying 'NO NEAR POINT' if flashed on the screen (Fig 5.9). This facility is useful to close loops by finding the start point and should always be used when digitising an outline for block-filling (section 5.4.8).

5.4.5 Continuous digitising

Digitised data consists usually of discrete points joined by straight lines resulting in pictures of very wooden appearance. One of the ways in which this can be avoided is by continuous digitising. In this mode the status of the pen is constantly sampled and as it is moved the coordinate positions are stored at regular intervals of time. This is the mode of operation normally used on the tablet. The sampling rate can be varied to be a function of time or distance. In either mode the cursor must first be moved a minimum prescribed distance before the coordinates of the new point are recorded. When sampling on time intervals the stored points not only represent positions in space but also moments in time. The use of this facility in describing motion paths of an object will be described later (section 5.6.4).

5.4.6 Curve fitting

The alternative to continuous digitising is to fit a smooth curve through a number of evenly spaced points stored in workspace. Curve fitting provides a more uniform and smoother curve than continuous digitising. It is also quicker since only a few points are required to define a complex curve. A number of techniques were tried. The first was based on a single pass projected quadratic equation:

$$y = a + bx + cx^2. \quad \text{for dominant } x$$

or $$x = a + by + cy^2 \quad \text{for dominant } y.$$

The coefficients a, b, c are calculated for each span. The initial condition is defined by the first three points. Subsequent span coefficients were determined from the starting slope and the two end points. This method presented two problems because of the equations in implicit form. The dominant term within the span must be determined in advance before one of the above equations can be selected because of infinite slopes. Also cases where the functions are double valued must be accounted for. The results become impractical for number of points greater than five because of cummulative ripple effect.

Another method was tried. First the equations were made parametric. This eliminates the problems of term dominancy and multivalued functions. The order of the equation was increased to a cubic. The technique is based on Chebyshev's theorem to determine good estimates of points for interpolating a low ripple poly-nomial (23)

Four points are required to fit a cubic equation. The Chebyshev points are obtained by dividing a semi-circle spanning the two extreme points by three (Fig 5.10). The perpendicular projection of the diameter of the points on the semi-circle give the interpolation points:

$$x = \frac{x_1 + x_2}{2} \pm \frac{x_2 - x_1}{4}$$

These correspond to points A, B, D and E in Fig 5.11.

Let cubic be

$$U = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

For symmetrical results normalise $-1<t<1$ so that

$a = -1$

$b = -\frac{1}{2}$

$c = 0$

$d = \frac{1}{2}$

$e = 1$

therefore $A = a_0 - a_1 + a_2 - a_3$

$B = a_0 - \frac{1}{2}a_1 + \frac{1}{4}a_2 - \frac{1}{8}a_3$
$C = a_0$
$D = a_0 + \frac{1}{2}a_1 + \frac{1}{4}a_2 + \frac{1}{8}a_3$

$E = a_0 \quad a_1 + a_2 + a_3$

give $a_0 = (-A + 4B + 4D - E)/6$

$a_1 = (A - 8B + 8D - E)/6$

$a_2 = (4A - 4B - 4D + 4E)/6$

$a_3 = (-4A + 8B - 8D + 4E)/6$

these are the coefficients of the cubic though A, B, D and E.

There is a misfit at the centre, the point C, when $t = 0$, $u = a_0$, given by:

$MAC = a_0 - C$

The cubic can be corrected by adding a quartic to pass through the centre. The quartic

$(1 - t^2)(1 - 4t^2)$ is zero at A, B, D, E, thus

$U = a_0 + a_1t + a_2t^2 + a_3t^3 - MAC\ (1 - t^2)(1 - 4t^2)$

So at $t = 0$

$U = a_0 - MAC = C$

This is equivalent to

$U = a'_0 + a'_1t + a'_2t^2 + a'_3t^3 + a'_4t^4$

solved for five points. However this is not Chebyshev's quartic because the points are equidistant.

The solution provides a smooth curve through every five points, but there is no continuity between the groups of five points. The method works well for path descriptions and well behaved curves but is not suitable for design purposes. For this one needs a much more flexible and powerful way of curve fitting.

5.4.7 Cubic spline

This package fits a smooth curve through any number of points with continuity of slope and curvature. The method is a simplified version of the one described in (20). The equations used are of parametric form so they can be generalised to any number of dimensions, e.g. for three dimensions:

$$X = A + Bt + Ct^2 + Dt^3$$

$$Y = E + Ft + Gt^2 + Ht^3$$

$$Z = I + Jt + Kt^2 + Lt^3$$

where all 12 coefficients are required for each span.

For a given span i the cubic can be represented by:

$$U_i = A_i + B_i t_i + C_i t_i^2 + D_i t_i^3$$

where the coefficients are given by:

$$A_i = U_i$$

$$B_i = (U_{i+1} - U_i) - (2U''_i + U''_{i+1})/6$$

$$C_i = U''_i/2$$

$$D_i = (U''_{i+1} - U''_i)/6$$

Appendix C contains the detailed mathematical derivation of these relations.

The package fits a smooth curve through points stored in workspace. In order to use the package the outline of a picture must be digitised with evenly spaced points. The more even the points the better the curve fits. Regions of great curvature require more points. Lines should be broken as in normal digitising procedure. The program is restricted to a maximum of 100 continuous lines at a time, more than needed in most typical applications, but can be expanded by increasing the buffer size. Three points define a parabola and two points a straight line.

5.4.8 Blockfilling

The system stores data in random x-y coordinate format but in some instances data in a sequencial raster scan format may be necessary. Data in raster format is valuable when output is on a raster scan device and when certain areas are to be cross-hatched. By scanning close enough, completely shaded areas can be generated and half-tone pictures may be generated by scanning at different intensity levels.

The system incorporates a facility to convert the random x-y data into a sequencial raster format. The program uses a single pass, single buffer scan-conversion algorithm.

Consider the boundary of a region as shown in Fig 5.12 defined by a point locus in a horizontal raster grid. The algorithm accepts any arbitrary scanning slope to x-y axes and any raster width. However for the sake of simplicity a horizontal scan is considered. For a diagonal scan the picture is assumed to be rotated. Boundaries should preferably be closed but they may intersect. If the former condition does not hold, the boundary will be closed by a scan line.

Lines making up the picture boundaries are differentiated as vectors from the scan lines. Two types of vectors may occur: sloping vectors, and horizontal vectors at inflexion or maxima-minima points.

Points on the sloping vectors are simply linearly interpolated and normalised to lie on the nearest scan line, but provision must be made for inflexion and extreme points. In this algorithm the inflexion lines are ignored as are all vectors parallel to the scan lines since they are implicitly contained in the end points of neighbouring sloping vectors. Maxima and minima are augmented so that tangential scanning lines always pass through a pair of boundary points at each extreme. This is accomplished during the boundary scanning process by repeating one of the boundary extreme points when the direction of y changes. The change in sign of Δy determines the boundary condition to be used. The y direction may be chosen positive upwards or downwards.

The three end conditions to be considered in the interpolation calculations, each requiring a different method are:

1 - Total span method: first and last scan points on vector are included.

2 - Forward step method: first scan point is discarded.

3 - Backward step method: last scan point is discarded.

The situations where these conditions or a combination of these conditions are used are depicted in Fig 5.12. The numbers correspond to the ones below:

1 - First vector in boundary loop: total span.

2 - Successive vectors:

(a) Same Δy sign: forward step method

(b) Change in Δy sign: total span method. (Maximum or minimum condition).

3 - Horizontal vector: ignored

4 - Last vector in boundary loop:

(a) Same Δy sign as first vector: backstep method

(b) Opposite Δy sign to first vector: forward step method.

Once calculated the points are sorted and partitioned into sets so that each set contains only points which have the same y coordinate. It is not necessary to provide storage for y coordinates in the data item because they are determined from knowledge of the y coordinate of the scan line being processed. The associated x coordinates of each set are sorted and ordered, increasing or decreasing in value, in pairs with odd-even subscript. To represent closed boundaries the number of x subscripts should always be even. Last odd subscripts are discarded.

The routine is used mainly to blockfill or cross-hatch bounded regions but the raster format data is also in a computer searchable form which possesses some useful algebraic and geometric properties, providing a rapid means of searching files for data associated with the geometric coordinates. It could be used in other applications such as:

1 - Compute the area contained within a close boundary. Provides a means of graphical integration.

2 - Select and erase part of pictures and generate the boundaries representing the union or intersection of several closed boundaries.

3 - Determine whether points lie within a closed boundary.

The last two properties are essential prerequisites when trying to determine visibility of faces of a solid in three dimensional space. The data is in a searchable format for use by a 'segment comparison technique' of hidden surface removal.

5.4.9 Alphanumerics

The system offers three types of alphanumerics each for a different purpose:

1 - Hardware generated characters on the Tektronix 611 storage tube. The characters are generated by a SEN Electronique CG 2018 CAMAC module. There is a selection of 2 character sizes and three intensity levels. There are 63 printing characters, as upper case letters only, space, carriage return and line feed. The code used is standard ASCII.

These characters,which can be generated at very high speed in any position on the screen,are suitable for displaying messages or text on the screen.

2 - Linear software character set - these characters are generated as symbols by a special program. The size is adjustable and set by the user. The characters can be positioned anywhere in workspace and in any orientation. The characters are defined by a 5 x 7 matrix with the corresponding matrix points stored in a special buffer and referred to by character numbers corresponding to their equivalent ASCII code. The current configuration offers 59 characters but may be expanded freely to include non-standard characters, like Greek alphabet, if required.

The random positioning and variable size make them suitable for incorporation as part of the picture, especially in the labelling and dimensioning of drawings. These characters can be stored either as symbols, so remaining unaffected by any transformation, or be made an integral part of the data.

3 - Bold type letters - these are a special type of user-defined symbols stored in a reserved file. The file is headed by an index which indicates the storage location of each letter. Each letter is scaled to fit within a box of fixed height and the lines making up the character are stored in the file separated by breakpoints. The breakpoint records also contain information on the size of the character width which is used to calculate the spacing between letters.

The alphabet is created by digitising each character and filing it in the right order in the special file. New type faces can be created by digitising new character sets to replace old ones. A number of fonts are already available but more can be added as required. To use the alphabet the size and position are digitised on the table and the characters generated by entering the desired character string from the keyboard.

These letters are suitable for titles and to produce solid logos and as they are stored in standard data format they can be manipulated in the same way as picture data. Fig 5.15 shows all three types of letters.

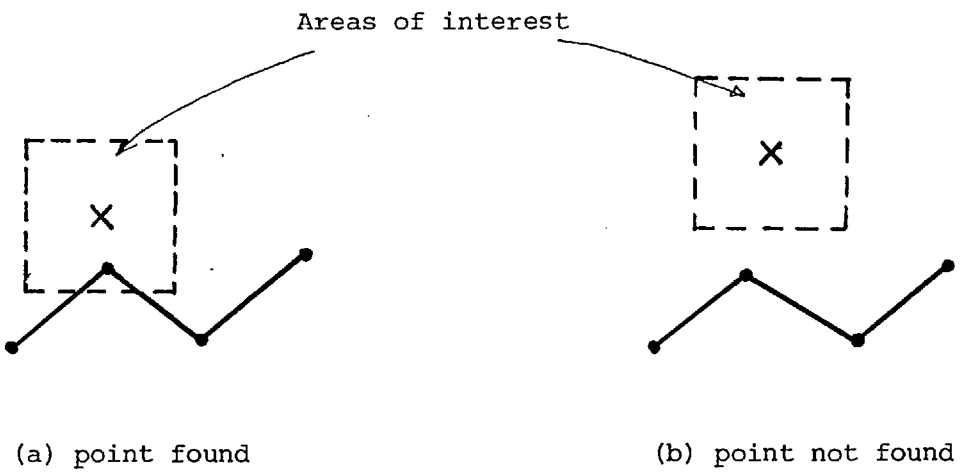


Fig 5.9 Find mode

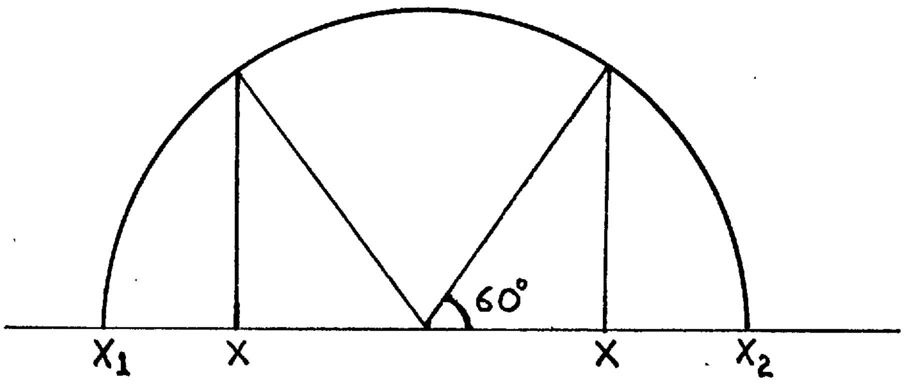


Fig 5.10 Chebyshev's points

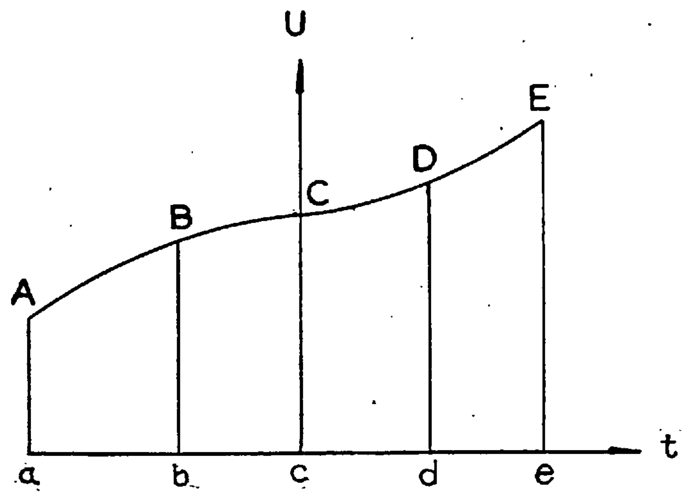


Fig 5.11 Cubic interpolation

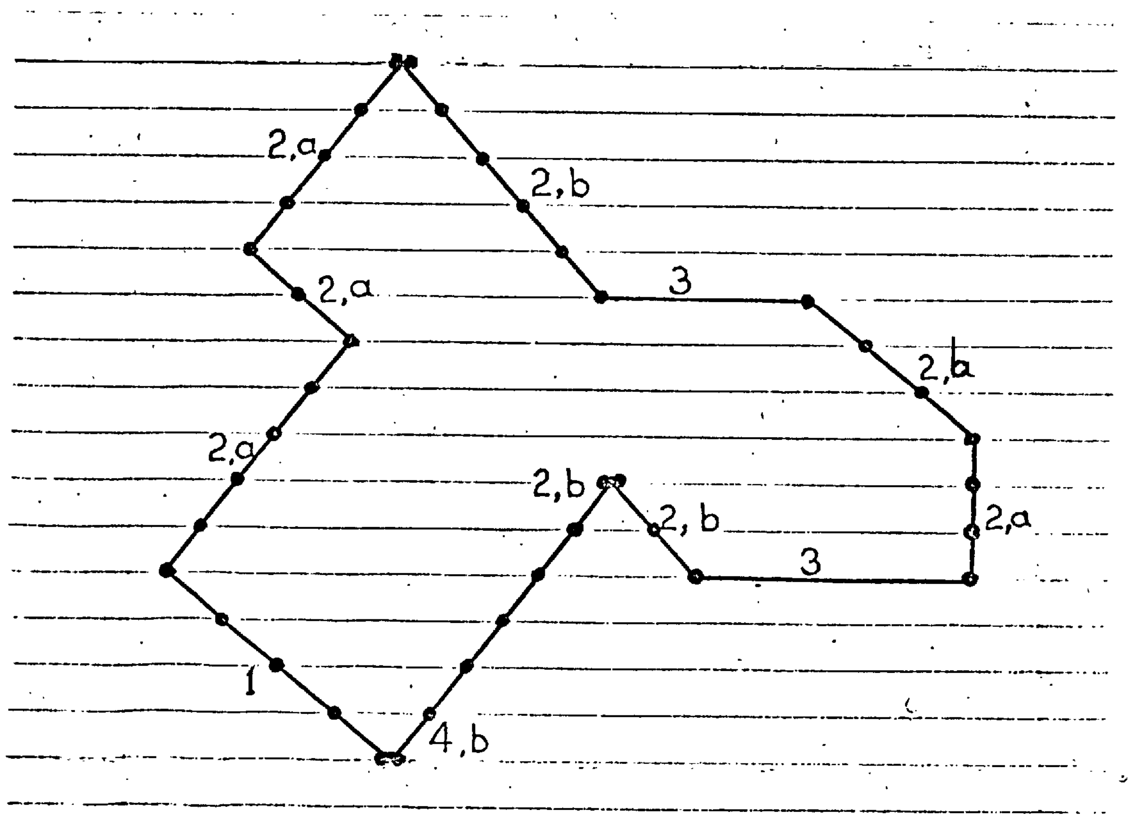


Fig 5.12 Example of scan-processing technique

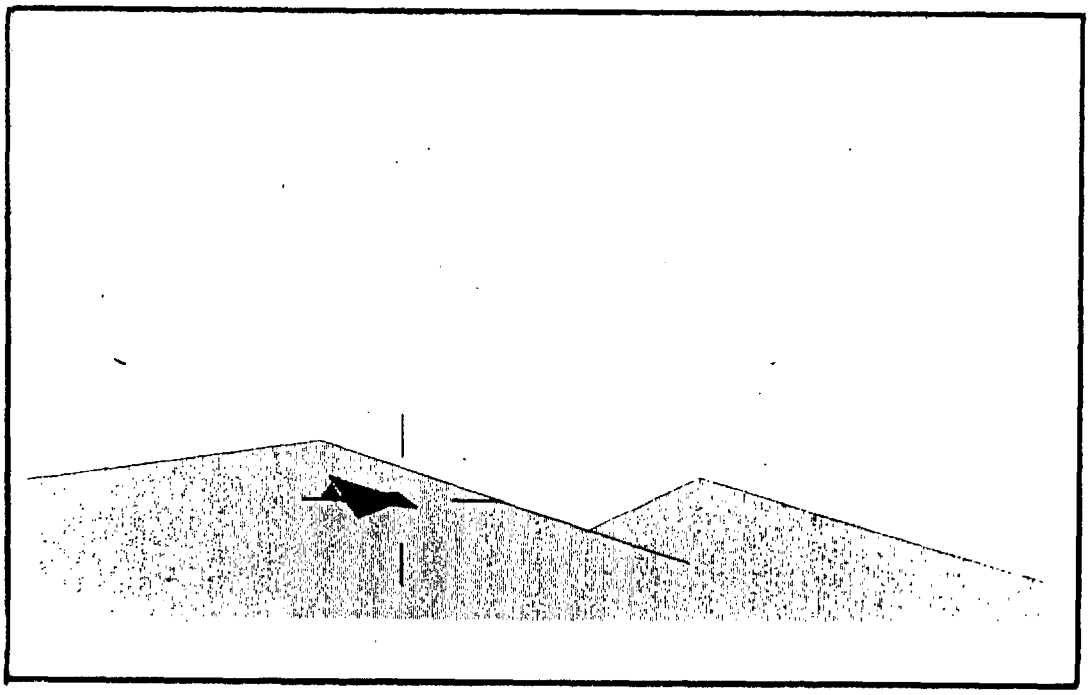


Fig 5.13 Example of shaded image

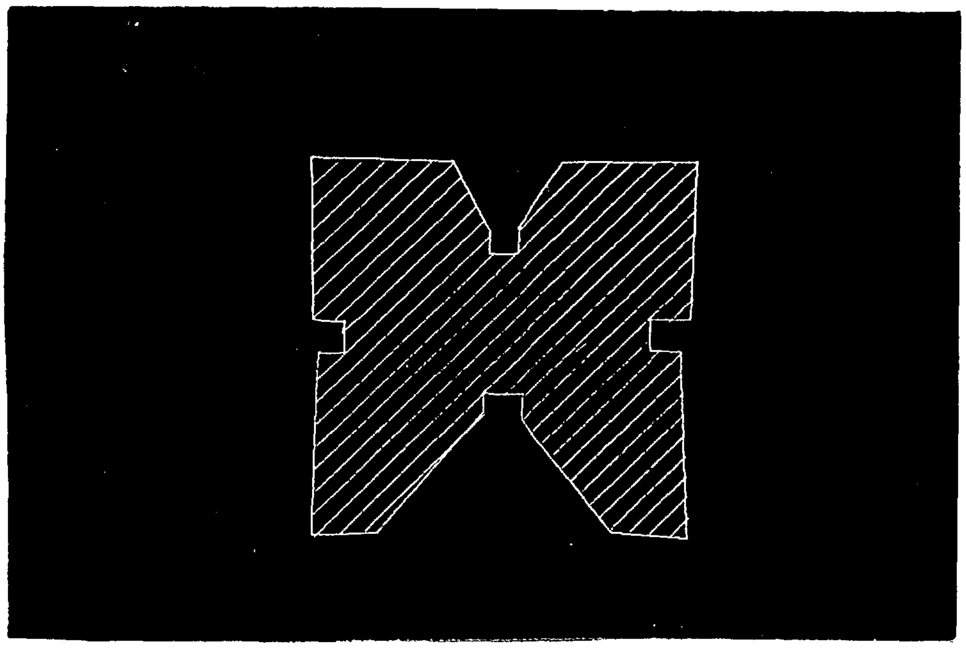


Fig 5.14 Cross-hatched cross-section

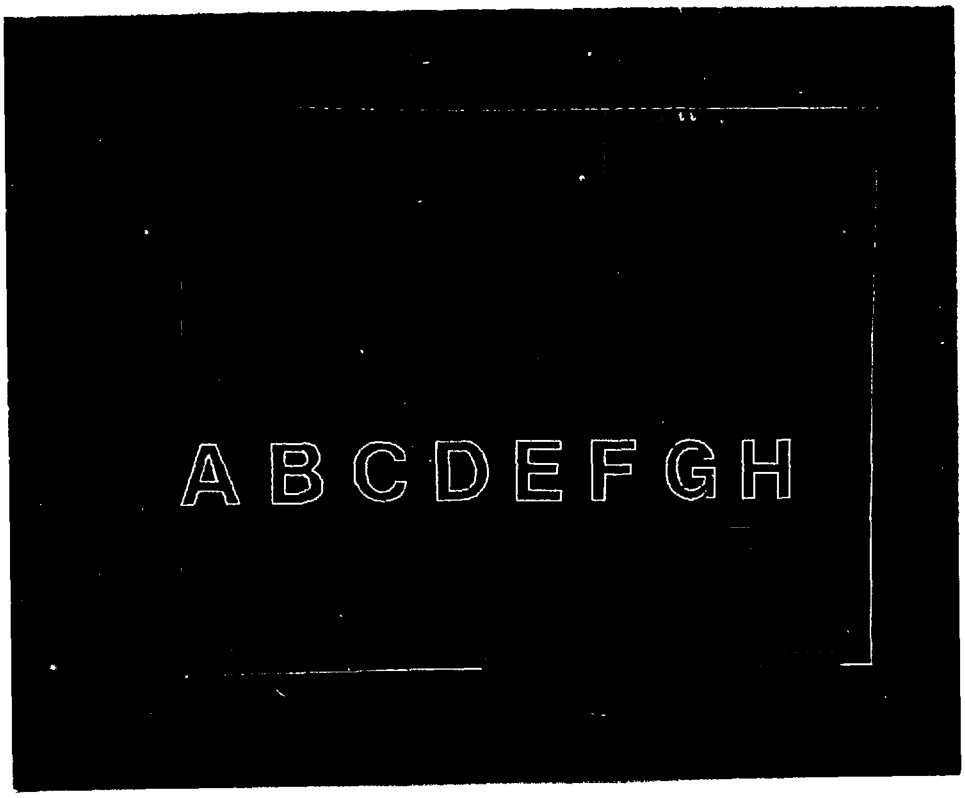


Fig 5.15 Three types of alpha-numerics

5.5 Editing

Data digitised from the table and constructed by viewing on the screen often contains mistakes which must be rectified, and this should preferably be done without the unnecessary burden of redigitising the entire picture. The essential function of an editor is to correct mistakes in data, but it can also be used for a much more powerful end - as a design tool. Once a data base has been constructed, it is often more economical to modify this data base to produce a new design rather than defining a new one. In most cases design consists of modifying and updating an old one. The computer data bank can have a file with all the old designs. As a modification is required the old one is recalled and edited to new specifications and a new updated drawing can be produced. The task of designing and drafting are combined in one.

One thing peculiar to animation is the use of large number of drawings which have many features in common, only differing in small detail. Therefore minor alterations to an initial drawing are sufficient to produce a number of pictures to create a reasonable sequence or to be used in a different application. This facility is particularly powerful when a picture is made up of macros. The interaction of a set of macros can result in a complete family of pictures for an entire sequence.

A comprehensive editing facility should incorporate the following features to deal with a specified data element: search, display, delete, insert, change and copy.

(a) Search - In order to be able to perform any of the other tasks the specified data item must first be found by using some sort of efficient search algorithm. The system uses a sequencial address search through the workspace in most instances. However block search is also possible. This is a more efficient technique if the data is known to be all contained in one particular block of data, as in cases where a cluster of lines or points are to be edited. Search time is also considerably reduced by the use of boxed data and minimax tests. Each block of data contains the maximum and minimum values of its contents, so the whole block can be tested to lie within the specified region before each element is tested. Search can be with respect to data points, distance from lines or within areas of interest, depending on the element of data required.

(b) Display - Display is mainly for identification purposes. When a data item has been found it can be continuously displayed (refreshed) on the screen to indicate that it is locked on to the item and expects some action from the user. There are several ways of indicating that an item has been found depending on whether the data is a point, line or macro-symbol.

(c) Delete - One of the possible actions of the user when a data item has been found is to delete it. The way this is done is by replacing meaningful data pointers in the data records with dummy data. Dummy data is ignored by any program and is equivalent to a no-operation instruction for a display compiler. A subsequent program can then be used to eliminate this redundant data and compress the file.

(d) Insert - A file can be expanded by adding data to it. Data can be inserted in any position on the screen, but it is merged to the end of the file only. This is satisfactory for display purposes but not for processes where the data is required in a given sequence. This problem does not arise in animation, but can be overcome by using the next facility.

(e) Change - This combines the above two functions. Thus instead of deleting an old item and inserting a new one, the old one is modified. The alternatives are normally repositioning or replacement of a data item.

(f) Copy - Facilities should exist to select sections of data file to be copied to another file. This is done by inserting two markers at the start and the end of the copied section. The insertion of markers follows the same search technique as in the other editing function.

The features described above apply to all editors in general, but there are specific differences related to individual data items. On the current system editing is permissible on three levels of identifiable hierarchies of data structure: point, line and macro.

(a) Point Editor - This only allows one feature - the repositioning of data points. Lines are repositioned by redefining the end points. The search technique is similar to the one used in FIND mode where an area of interest surrounding the cursor position is scanned. If a point is found the cursor locks in this position until a new point is defined, otherwise a message is flashed on the screen to indicate that the point has not been found.

(b) Line Editor - Used to delete lines. To determine the line to be deleted the program searches for the line with its mid-point nearest to the cursor. Only lines within an area of interest are tested. Once a line has been found it is continuously displayed on the screen awaiting a decision from the user: to delete the line or advance the search. The line editor can be used to eliminate large areas of data on the screen by continuously deleting lines. However, the macro editor is more suitable for this purpose.

(c) Macro Editor - This also operates on symbols as they are stored in identical fashion. It allows a number of manipulations to be performed on data defined as macros:

delete

translate

rotate

scale

mirror

replace.

Chains of unbroken lines may be defined as a macro in order to use the macro handling facilities. Since these macros represent a large proportion of the data in workspace, this allows a compact way of manipulating data within files. In an editing sequence, a macro is identified by a box enclosing its outline. The macro is found by positioning the cursor within this box. The user selects one of the above functions by pressing one of the pen buttons. A parameter may also be requested.

5.6 Picture Manipulation

The editing facilities described in the previous section are designed to handle only parts of the pictures: the picture manipulation facilities described in this section treat an entire picture as an entity.

In animated graphics movement is created by introducing changes from frame to frame. Animation is an event that takes place between frames. It is these differences between frames that give rise to movement when a sequence is viewed in continuous mode. Mathematically these changes may be regarded as transformations or mappings of one set of points from one plane on to another.

There are several ways in which these transformations can be defined, based on one or more pictures:

1 - By first defining two drawings representing two extremes of action. This is in-betweening and requires two key drawings per sequence.

2 - The path of the object is either drawn or described by a known equation, normally in the form of a parametric curve. The sequence is defined by an object-path pair and a set of parameters relating to the position of the object or the observer and the duration of the sequence.

3 - By means of special effects. These are usually non-linear transformations which cannot be accurately defined as movements. They consist of distortion in a two dimensional plane or a pseudo three dimensional effect with perspective superimposed.

4 - The final method is by means of pans and zooms over a still

picture. This is known as 'filmograph' and is a very economical form of animation, widely used in educational and industrial fields. The same drawing may be used for a number of sequences.

The first three methods will be described in the next sections. The filmograph will be described in the section 5.8, since it is directly related to windowing and display.

5.6.1 2-D Transformations

Every picture manipulation performed in the system can be represented by a matrix transformation:

$$X_T = XT$$

where X is a vector in the data file and X_T the transformed vector contained in a display file. X_T can be written back in the data file to represent a transformed new data source.

For two dimensional data X and X_T are represented by

$$X = [x \; y]$$

$$X_T = [x' \; y']$$

Thus T is a 2 x 2 matrix:

$$T = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

where the terms a_{ij} can be defined for each transformation

$$[x \; y] = [x' \; y'] \quad \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

T can be a single matrix or concatenated by multiplying two or more matrices. The coordinate values in X are normally relative to a reference point X_r. Therefore, in general:

$$X_T = T X + X_r$$

where $T = T_1 * T_2 * T_3 * \ldots.$

The order of the individual matrices T_i determine the resultant transformation. However using 2 x 2 matrices it is not always possible to concatenate the matrices. For example, to rotate a point $\begin{bmatrix} x & y \end{bmatrix}$ through an angle θ about an arbitrary point at a distance $\begin{bmatrix} R_x & R_y \end{bmatrix}$ from the origin, conventional operation in 2 x 2 matrices would give:

translate $\begin{bmatrix} x' & y' \end{bmatrix} = \begin{bmatrix} x & y \end{bmatrix} - \begin{bmatrix} R_x & R_y \end{bmatrix}$

so that $\begin{bmatrix} R_x & R_y \end{bmatrix}$ becomes the origin.

Rotate $\begin{bmatrix} x'' & y'' \end{bmatrix} = \begin{bmatrix} x' & y' \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$

translate again to old origin

$$\begin{bmatrix} x''' & y''' \end{bmatrix} = \begin{bmatrix} x'' & y'' \end{bmatrix} + \begin{bmatrix} R_x & R_y \end{bmatrix}$$

The transformation becomes:

$$X''' = (X - T)\ R + T$$

The matrices cannot be combined into one single transformation. This is possible by using homogeneous matrix representation (24), where each transformation in two dimensions is represented by a 3 x 3 matrix.

Although there is no practical advantage in storing data as homogeneous coordinates it is useful to perform transformations with 3 x 3 matrices. Add a third element

$$\begin{bmatrix} x' & y' & 1 \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & 1 \end{bmatrix}$$

The extra dimensional element represents a scaling factor and is always kept as unity because a single common scale is used which does not require to be constantly redefined and stored as data. This also reduces the data storage requirement by 25 per cent.

The multiplication of the last column can be eliminated, reducing the computation of the matrix from 9 multiplications and 6 additions to 6 multiplications and 4 additions:

$$[x' \; y'] = [x \; y \; 1] \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix}.$$

The above rotation becomes:

translation $T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ R_x & R_y & 1 \end{bmatrix}$

and rotation $R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

and $X = [x \; y \; 1]$

The matrices can be combined as

$X''' = X(-T)\ RT$

$$(-T)\ RT = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ -R_x \cos\theta - R_y \sin\theta + R_x & R_x \sin\theta - R_y \cos\theta + R_y & 1 \end{bmatrix}$$

Again the last column can be eliminated from the final matrix computation. It is however necessary for the three matrix operations.

5.6.2 Input parameters

In order to control and specify a movement parameter which defines the start and end conditions and any intermediate stages are required. These are automatically requested by the system by means of messages either on the screen or the teletype.

Typical parameters for an animation sequence are:

1 - Positioning vectors - usually in x-y cartesian coordinates.

2 - Speed - normally defined in terms of number of frames.

3 - Number of cycles - if process is cyclic.

4 - Step increments - define change between frames.

Some of the above are complementary and often one parameter can be obtained from the others. Appendix A describes the use of the parameters requested by the manipulation programs.

5.6.3 In-betweening

This is the most common method of generating a smooth flow of animation when using conventional techniques. It is also one of the most tedious parts of any animation process.

The movement of the character or object to be animated is characterised by a series of key positions known as extremes. A series of intermediate drawings, 'in-betweens', are then created to depict the progressive action between these extremes. How many drawings are needed in-between any two extremes is determined by the speed of the movement, the distance between the two extremes and whether the action is being photographed on ones, twos, or threes.

Most action is planned to be photographed on twos, as this is the most economical without excessive jitter, and requires 12 drawings per second of screen time.

The characteristics of a movement created by in-betweens is intrinsically contained in the extremes. These drawings are enough to define and characterise the whole movement. The speed of action is then simply determined by the number of in-betweens used. The animation is controlled and generated by the pictures themselves and this is called picture driven animation.

In the computer in-betweens are produced by first digitising two extreme positions, then the intermediate drawings are generated, by interpolating between those key drawings which have been stored in separate files. If the interpolation were to be linear, the effect yielded would be very wooden. An ease-in and ease-out should be embodied at the two extremes, resulting in an acceleration at start and a deceleration at the finish. This is called 'fairing'. The rate of fairing is dependent on the type of action required. It may be restricted to the two extremes or to cover the whole cycle. The user defines these rates before creating the in-betweens.

Several forms of interpolation can be used to give the fairing. Cosine interpolation has been shown to give good results.

$$X = (XE - XS) \cdot (1 - \cos(T))/2 + XS$$

gives the position of a point X at line T with

$T = 0$ when $X = XS$

and $T = \pi$ when $X = XE$

A graphical representation of the equation is shown if Fig 5.16, which is used to produce a spacing guide. There are 16 frames in the sequence with 4 fair-in's and 5 fair-out's. The positions on the base line are obtained by dividing a hypothetical quarter of a circle into the number of fairing frames required and dropping the normals on to the radius. These are in fact Chebyshev points (see section 5.4.6). The size of the quadrant is made to give the same step length between frames 4-5 and frames 11-12.

The two drawings used in an in-betweening sequence can be similar to represent the same object or character, or be completely different. In order to preserve the characteristics of a character, the two drawings should have as many common features as possible. However, they need not be identical nor has the user to concern himself with the length or number of lines. The program will automatically equalise the data in the two files. The program uses a two pass, single buffer algorithm. On the first pass the program searches and compares the data. Any discrepancies between files are removed by equalising the data. Segments and lines in the files are subdivided or duplicated to make the length of the files equal. On the second pass the program determines its present state in the sequence and uses the current parameters to interpolate the in-betweens. For entirely unsimilar extreme drawings the effect obtained is called MERGE. This simply gradually dissociates one picture and assembles into another. It provides an effect similar to fade-in-fade-out but it is achieved by the picture itself, rather than variations in the exposure intensity.

A variant of the above is provided by the EXPLOSION routine consisting of the scattering explosion of lines making up one picture, followed by an implosion to form another picture. The program can also be used on half-cycles, i.e. to destroy a picture exploding to obliteration (an erasing effect) or to create a picture by converging scattered data (a pop-on). The program allows a certain degree of control but the effects are mostly random.

Fig 5.17 shows the in-betweening sequence of a cartoon character.

5.6.4 Parametric representation of movement

This is a one path-one object means of defining a movement. Each object is associated with a path. Several path-object pairs can appear on the final screen and made to interact. The same object or path definition can be combined with other paths or objects to form different pairs.

The parametric curve describing the exact path of an object is defined by storing the position of a reference point on the object at fixed intervals in time. Therefore the curve contains both position and timing information.

There are two ways in which the paths can be generated:

1 - By digitising from a graphical path description. If the position vectors are known the points can be discretely digitised. If not, the continuous digitising facility described in section 5.4.5 can be used to make the path time dependent. This facility is enabled by setting it to time mode. The sampling rate can be varied but is normally set to 24 samples per second or fractions of it. So, as the path is traced, the position of each point at exact time intervals

is read and stored. This provides a natural means of inputing the rhythm and pace of a movement which is much easier to feel than to rationalize. The storage of points is not actually activated until a button on the pen is pressed and actually moving, to prevent accumulation at points when the pen is stationary.

2 - The other way of generating paths, more powerful and more accurate, is by use of parametric equations of the form:

$$x = f(t)$$

$$y = g(t)$$

where the reference coordinates (x, y) are generated at specific intervals of time. The equations produce a string of position vectors which are stored as a path description.

The equations can be of any complexity, including numerical solutions to differential equations. The results provide a graphical solution to the problem represented by the equations. With the solution presented dynamically, the phenomena can be studied in real time which is the basic requirement for dynamic simulation. For instance, in the study of motion of fluid particles a point or a circle is used to represent a particle. The numerical solution provides the position of the particle at each instant in time. The path can be displayed in real-time to show the progression of the particle or in one frame to represent a stream line.

This method has the disadvantage of requiring programming knowledge on the part of the user to incorporate the equations in a program. However this is a minor problem, since most users requiring a very complex solution can be assumed to have extensive programming knowledge. Usually the solution is first obtained on a large computer and the results then fed as data into the system.

The defined path is stored in the same data format as pictures, therefore it can be edited and manipulated in the same fashion. It is also stored in a standard permanent file like a picture description so there is no physical difference between a path description and a picture except in the way they are used.

To use a path it is first recalled from the permanent file into workspace. Then an object file is selected from one of the permanent files and a reference point defined, which can be defaulted to the mid-point on the object. The program is accessed by digitising the appropriate square on the menu. The object will precess along the path until stopping at the end or being interrupted by the user.

The parametric curve facility allows two modes of operation: with rotation, without rotation. The first one allows the orientation of the object to be changed with respect to the slope of the path in a motion similar to a vehicle on a track. The second mode restricts the object to a fixed orientation with movements restricted to translation as in the motion of a rigid structure.

Fig 5.18 shows examples of the two modes of operation.

The facility described here is two dimensional;path description really shows its potential when used in three-dimensional applications as will be described in Chapter 6.

5.6.5 Special effects

This is a generic term applied to all transformations that do not fall within any well-defined specifications. The effects are simple to achieve by the computer and are visually very impressive. It is a digital means of producing effects similar to the ones obtained from television but allows more precise control and since it is based on well-defined parameters, can be repeated. There is considerable demand in the industry for producing these effects, especially for titling and commercial logos on television.

Special effects are achieved by the mapping of an original picture on to different planes. These mappings can be represented by matrix transformations of the form

$$X_T = XT$$

as described in section 5.6.1, where T is a concatenated matrix. Non-linear transformations can also be used to introduce distortions in the pictures. The order of the terms in the matrix T determines the effect created. Each transformation is contained in a program stored in a library of functions. The functions are requested by calling from the corresponding menu squares (Fig 5.19). The functions can also be called by program in any order to create additional effects by combining the library ones.

The library of functions is being constantly expanded and the present configuration contains the following transformations:(Fig 5.20)

(a) SPIN/ZOOM - the data is rotated about an axis normal to the viewing plane. An optional scale factor can be superimposed to give a zoom effect. The transformation is represented by

$$X_T = XT_{scale}\, T_r + X_r$$

T_r is the rotation matrix

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

θ being the angle of rotation.

T_{scale} is the zoom factor which depends on the number of frames and current frame count.

(b) FLIP - The data is rotated about an axis in the viewing plane and θ is defined in a plane normal to the x-y plane. The FLIP plane may be at any angle of inclination α to the x axis. The data must first be de-skewed by α, thus for FLIP:

$$T = T_{deskew}\, T_r\, T_p\, T_{skew}$$

T_{skew} and T_{deskew} take the same form as T_r with θ replaced by $-\alpha$ and α respectively. T_p is the perspective transformation to be described later. The inclusion of perspective gives a three-dimensional effect.

(c) SQUASH/STRETCH - These are linear scaling effects used to expand or contract a picture along any direction. Distortion endows a flat image with vitality and dynamism.

$$T = T_{deskew}\, T_s\, T_{skew}$$

T_s is the scaling matrix.

$$\begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$$

(d) TUBE - This program wraps a flat image on to a cylindrical surface which is then revolved about its centre axis. The cylinder may be of any radius size and axial orientation. For a horizontal cylinder only the y coordinate is transformed

$$y_T = R \sin (y/R)$$

where R is the radius of the cylinder. TUBE is an example of obtaining three dimensional effects from two dimensional data. This is achieved by transferring the data from two dimensions to a three-dimensional space. The transformation is then performed in three dimensions and the result projected back on to the two dimensional plane through a perspective transformation.

(e) SPHERE - Similar to tube, also creating three dimensional effects using two dimensional data. The image is wrapped on to a revolving sphere. The transformation is represented by

$$X_T = X\, T_{deskew} \begin{bmatrix} (R/L).\sin\theta & 0 & 0 \\ 0 & (R/L).\sin\theta & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ R\cos\theta \end{bmatrix} T_p\, T_{skew} + X_r$$

where R = radius of sphere

$$L = (x^2 + y^2)^{\frac{1}{2}}$$

$$\theta = L/R$$

(f) SINE - This routine creates a ripple effect by wrapping the picture on to a corrugated surface. The wavefront may be in any direction in the x-y plane. For a horizontal wavefront:

$$x = R \quad 2.N + 1 - \cos (x - R.N)$$

where R is ¼ of the wavelength

N is half the integral number of wavelengths

$$= \left| \frac{x}{2R} \right|$$

(g) SKEW - Shears data about a horizontal plane. For skew angle θ and shear plane given by $y = y_s$

$$x' = (y - y_s) \sin \theta + x$$

$$y' = (y - y_s) \cos \theta + y_s$$

(h) CIRCLE - Wraps the points in the workspace round the periphery of a circle. For a circle of radius R and centre at (x_c, y_c)

$$\theta = (x - x_c)/R$$

$$x' = (x - x_c) \sin \theta + x_c$$

$$y' = (y - y_c) \cos \theta + y_c$$

As it can be seen from the equations above CIRCLE is similar to SKEW except it is progressive, proportional to the value of x.

(i) TWIST - Twist is a gradual FLIP with the flip angle varying with the x position.

i.e.

$$\theta \alpha x$$

$$SCL = 1 + y \sin \theta$$

$$x' = x/SCL$$

$$y' = y' \cos \theta/SCL$$

with perspective factor included.

(f) TORQUE - Is spinning gradually about a point in the x-y plane. The angle of rotation varies with the distance from the centre of twist and decreases towards the edge of the screen.

$$\theta = 1 - \frac{R}{R_1} \quad \theta_O$$

where θ is the twist angle at distance R from the centre of twist. R_1 is the distance of the farthest point from the centre and θ_O a nominal angular increment at the centre.

(k) ELASTIC - The mapping of the data on an elastic surface being stretched by pulling from a point. The surface corresponding to the screen is assumed to be held by a frame round the edge as it is distorted.

The movement of a point $\left[x, y \right]$ is given by

$$\frac{x}{x_1} \cdot \frac{y}{y_1} \quad dx_1$$

where $\left[x_1, y_1 \right]$ is the point being pulled by distance dx_1.

The library can be expanded by including more functions as required, but by judiciously combining the ones available, it is possible to obtain very complex effects.

However, underlying all these purely visual effects is a much more valuable application. The distortions obtained represent a mapping of one system on to another one. Phenomena involving distortions of a system can also be represented as a function. A simple example is a beam under load. As it bends each point on the beam will have a new position defined by the local strain. The strain distribution represents a new system on to which the original beam can be mapped. This provides a means of visually studying distortion of planes, shells or any mechanism under stress or vibration. The actual movements can be on a microscopic scale but these can be magnified for the sake of clarity. Again, as with parametric curves, to use this facility considerable programming experience is required, apart from knowledge of the solution to the problem.

Apart from distortion, the transformations can also be regarded as forms of movements. Thus a point projected on to a cylinder or sphere does not result in any distortion but when the solid is rotated the point can be regarded as tracing a path, defined by the object in three-dimensional space.

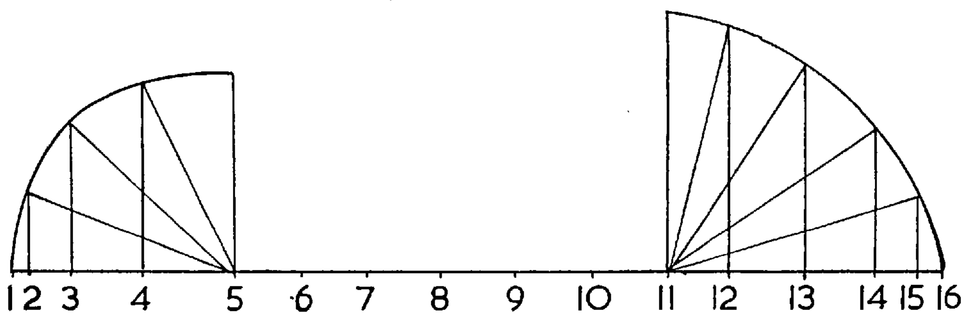
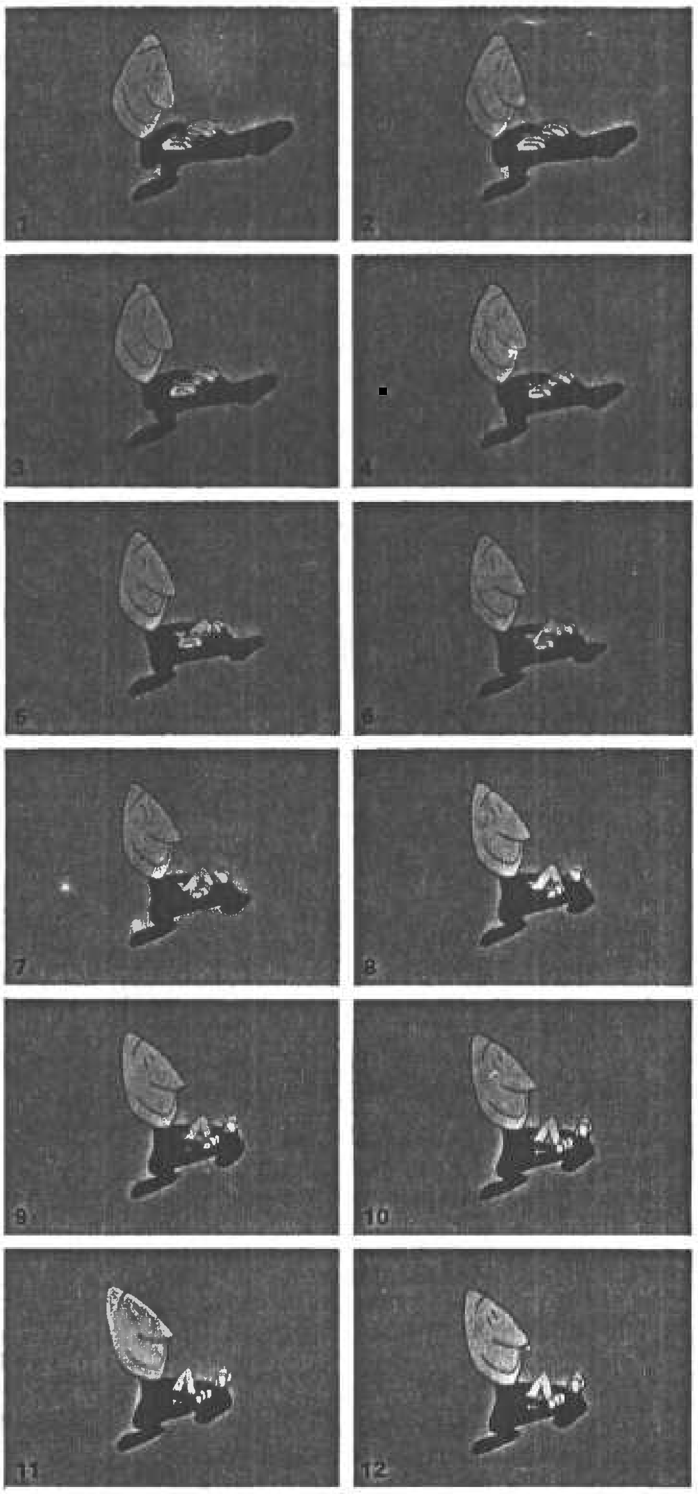
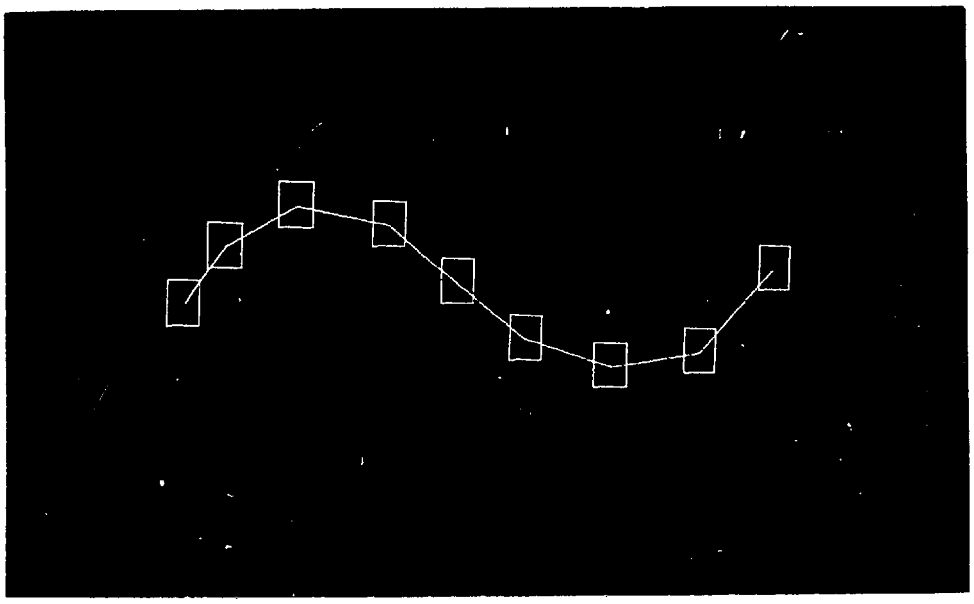


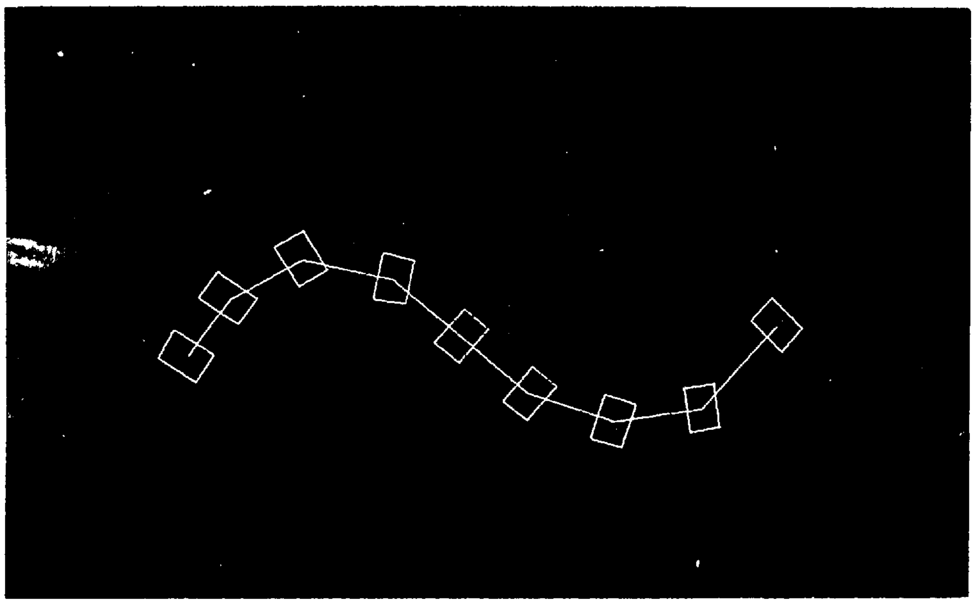
Fig 5.16 In-betweening fairing



5.17 In-betweening of a Cartoon Character



(a) fixed



(b) with rotation

5.18 Parametric path representation

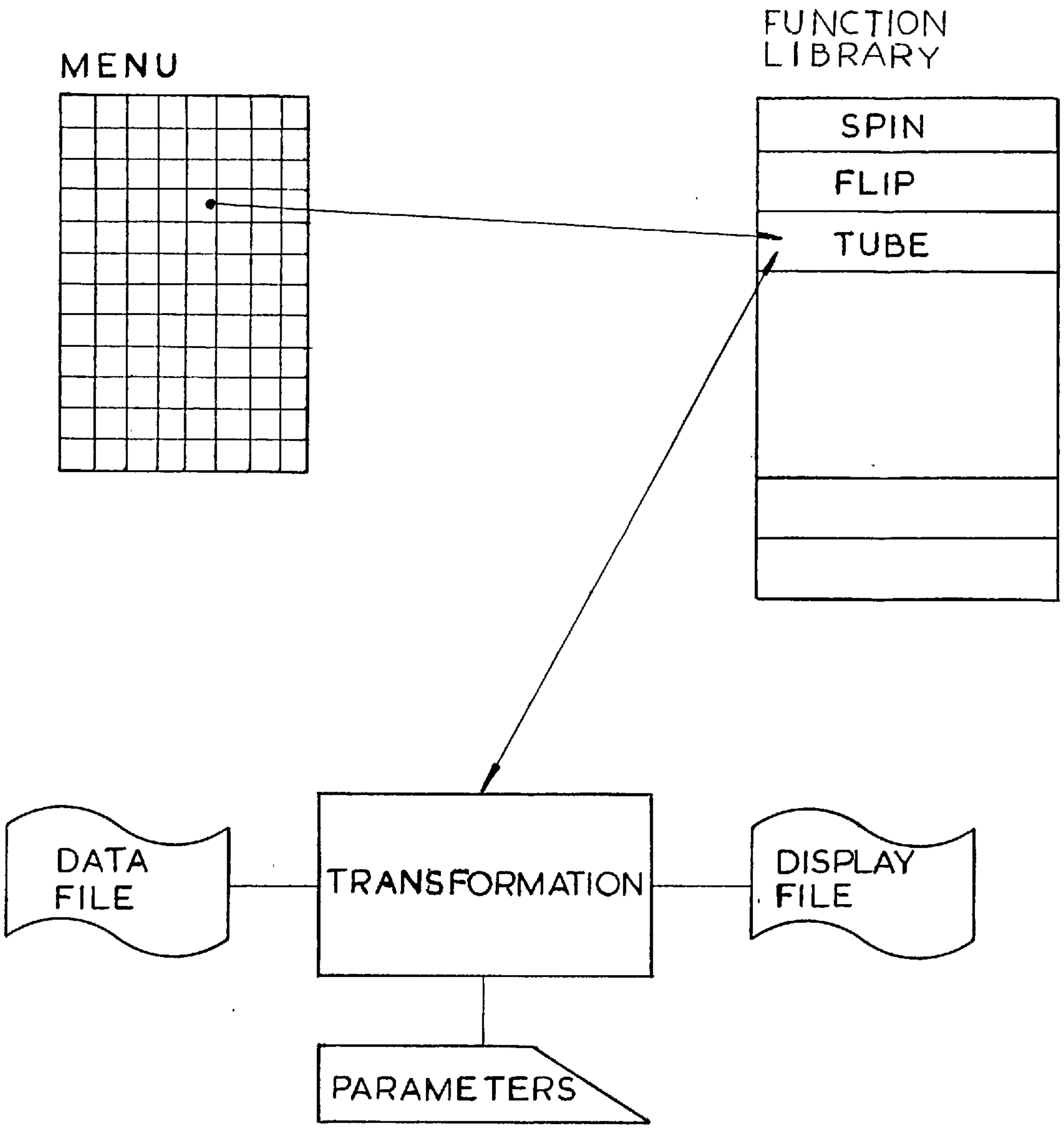


Fig 5.19 Operation of function library

	PLOT WORKSPACE									
Manipulation Routines	PAPER	CEL	SET PLOTTER	PLOT FILES	DEBUG	CONT. DIGITISE	MOVE PICTURE	CURVE FITTING	MINCE	BLOCK FILL
	CAMERA MOVES	FRAME	3-D	LETTER				P CURVE	IN-BETWEEN	RECOVER
	TUBE	SPHERE	SINE	DISC						
	XPLODE	SPIN	FLIP	SQUASH - STRETCH	SKEW	ELASTIC	TORQUE	TWIST	ERODE	
	GLOBE	RADAR								

Fig 5.20 Commands menu

5.7 Output

As mentioned earlier, the system offers two major forms of output: display on the Tektronix 611 storage tube and plots on standard animation cels. The output in both cases are directly related; as the area viewed on the screen corresponds to the area plotted.

5.7.1 Display

In animation, with the large number of displays involved, a lot of time is consumed in converting structured data into display signals. The display file may be regarded as a table of instructions to be executed by the display processor. The display file in this case is the same as the workspace since with a storage tube there is no need for a conventional display memory (section 6.2.1 describes the use of intermediate display files in three-dimensional display). The display processor consists of a program and a number of subroutines built into a library which is loaded to execute the instructions contained in the display file (Fig 5.21).

The instructions are stored in [I, X, Y] records with I representing the command code as to how the display is to be affected and X, Y the data. Compiling of the graphics language is simple, for instructions are always generated sequentially. The instruction code I has similar meaning to those depicted in Table 5.1.

for I = 1	drive beam from last position to [X, Y] with beam off
I = 2	same as 1 but with beam on
I = 0	stop display
I = 9	is a NO-OP (no-operation) command

I = 13 activate symbol program for symbol contained in X

I = 20 pause.

For fast display it is essential to devise efficient algorithms to convert the display instructions to signals driving the CRT. Two of the most time-consuming stages are the scaling of data to a window of fixed size and the elimination of data that lies outside the window.

5.7.2 Windowing and clipping

When coordinates are to be translated into a view to be presented on the VDU face a window which represents in the coordinate systems the area to be displayed must be established. The window when scaled to scope coordinates usually represents an area not larger than the scope raster. Data that lies outside the scope raster must be eliminated or 'wrap-round' effects will occur. Some hardware devices have automatic scissoring in which the window and the display vectors may be larger than the scope raster. For some applications, as in this case, where hardware clipping is not available, software clipping must be employed.

Two areas must normally be defined. A viewport and a window. They may be of different proportions but the viewport normally lies within the screen area. In this application the viewport is made to coincide with the whole of the screen to take advantage of maximum screen area.

The limits of the window are determined by the coordinates on the bottom left-hand corner taken as (0, 0) and the dimensions of the two frames. The window may be given the proportions of the camera frame so that the display corresponds exactly to what is recorded on film. However, an aspect ratio of 50 x 36 corresponding to the standard academy size cel is more useful as will be seen later for plotting purposes with the full screen size corresponding to field 12.

The original display program written for BASYS (21) was found to leave a lot to be desired, It was slow and performed unnecessary transformations on invisible data. Clipping was performed on the transformed screen coordinates and the windowing scales were stored in a form that could not be used directly by the transformation. In GAGS the clipping is done before scaling to screen coordinates, the window defined in actual table coordinates, so that the data is considerably reduced before display signals are generated (Fig 5.22). For cases where, as in three dimensional work, an intermediate display file is required the size of this file is also reduced.

The screen has a range of 0 to 1024 with the origin at the bottom left-hand corner. For an aspect ratio equal to 36 x 50 and in order to use maximum screen area an aspect ratio of 1024 x 737 was chosen.

The windowing transformation is given by (Fig 5.23):

$$X_s = 1024 \frac{X_p - X_w}{DX}$$

$$Y_s = 737 \frac{Y_p - Y_w}{DY}$$

where (X_w, Y_w) are the coordinates of the bottom left-hand corner and DX, DY the frame length of the viewport defined on the table.

$$\frac{DX}{DY} = \frac{1024}{737}$$

The clipping algorithm tests for position of lines in relation to the window. First it tests for the trivial cases: whether the line lies entirely within or without the window. For the first case, the line is immediately accepted. For the second test it first tests for cases where the line is entirely above, below, to the right or left of the window (Fig 5.24). If the line does not satisfy both tests it is assumed to intersect the window. The intersecting lines are trimmed so that the external points lie on the edge of the window. In cases like line A (Fig 5.25) where the line intersects one extended horizontal edge and one extended vertical edge, it is assumed to intersect the window. The trimming program reduces such lines to a point outside the window which can then be trivially rejected.

5.7.3 Plotting

Plotting is not unlike display, with signals sent to a CRT deflection system replaced by signals used to drive D.C. motors to move the plotter-pen along the x and y axes. The plotting and display areas are identical because of the window size chosen for the display. The data shown on the screen is scaled and plotted on a cel registered by a peg-bar fixed to the table (Fig 5.26). The plot area corresponds to a cel size defined by the user before plot is started. To initialise the plotter, the pen must be moved to the centre of the cel with the motors switched off and then turned on.

The windowing and clipping algorithms are the same as those for display with plotting and display being in fact controlled by the same program. On entry the program tests for the output mode to determine which device to serve: if in plot mode it sends data to the table, if in display mode to the screen, using different scaling factors for different devices.

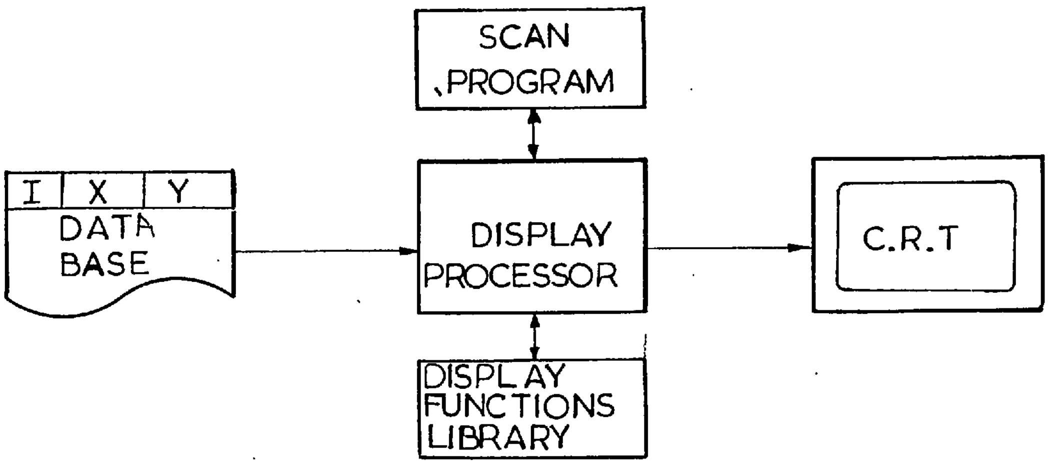
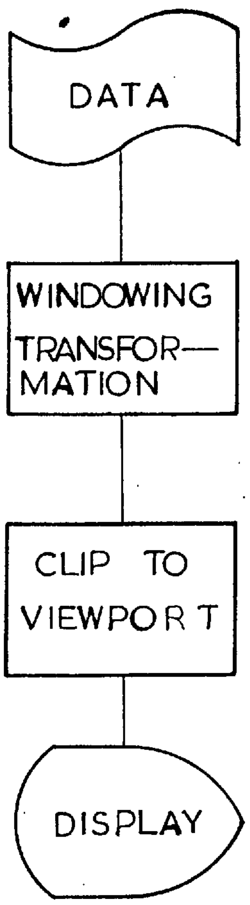
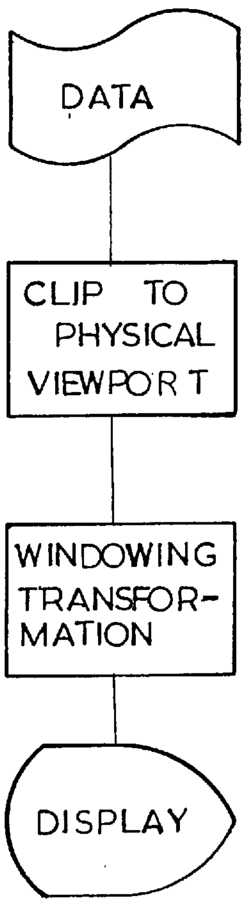


Fig 5.21 One pass display processor

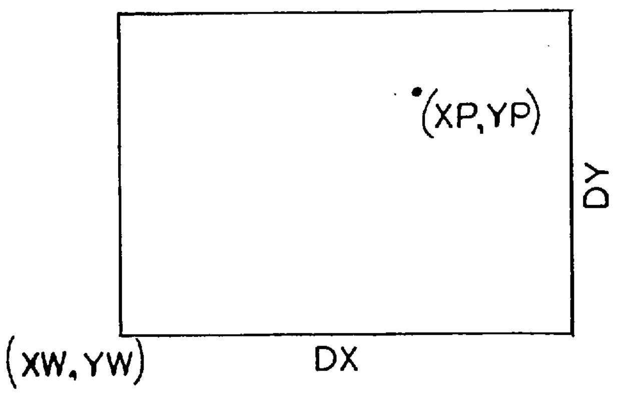


(a) BASYS

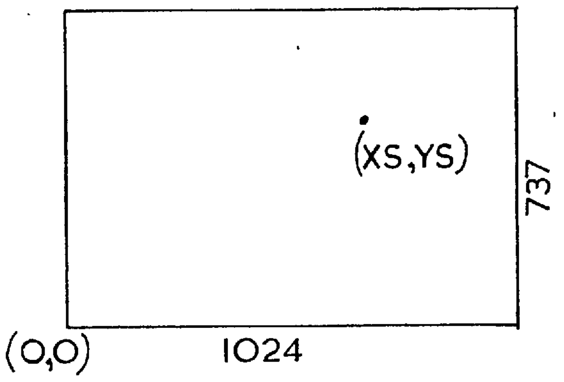


(b) GAGS

Fig 5.22 Processor sequence



(a) Table



(b) Screen

Fig 5.23 Windowing transformation

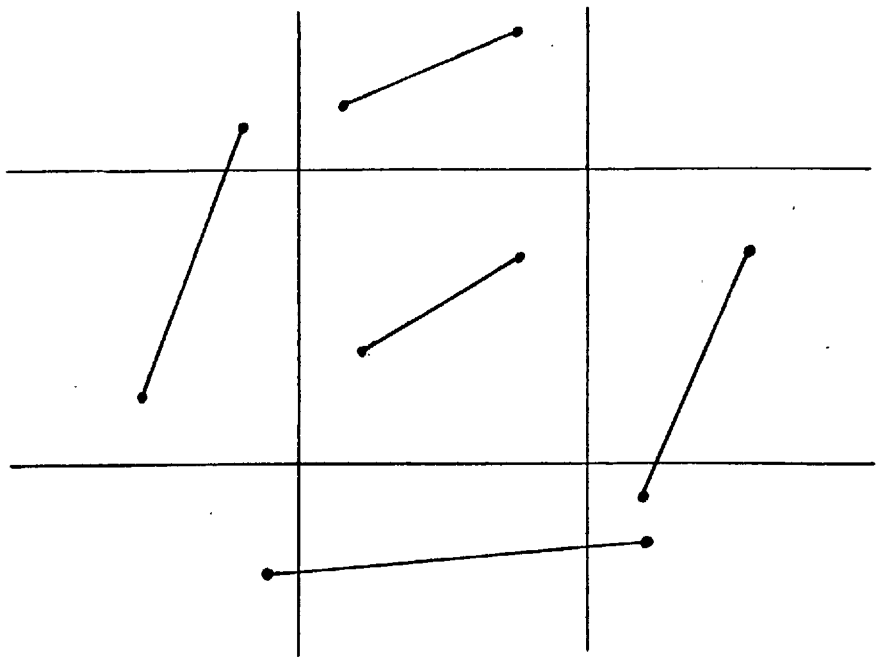


Fig 5.24 Trivial cases of clipping

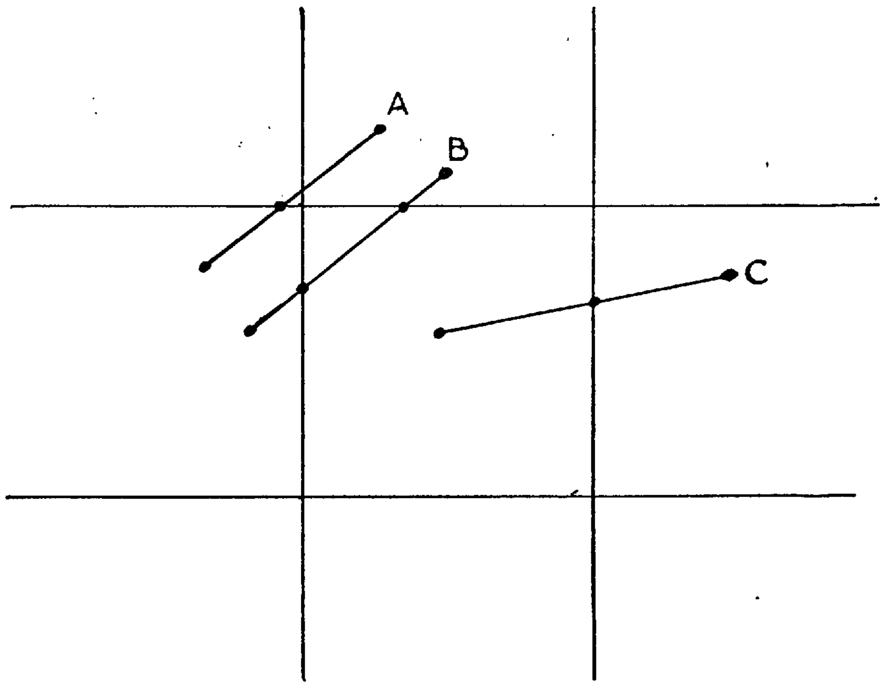


Fig 5.25 Intersecting lines on window

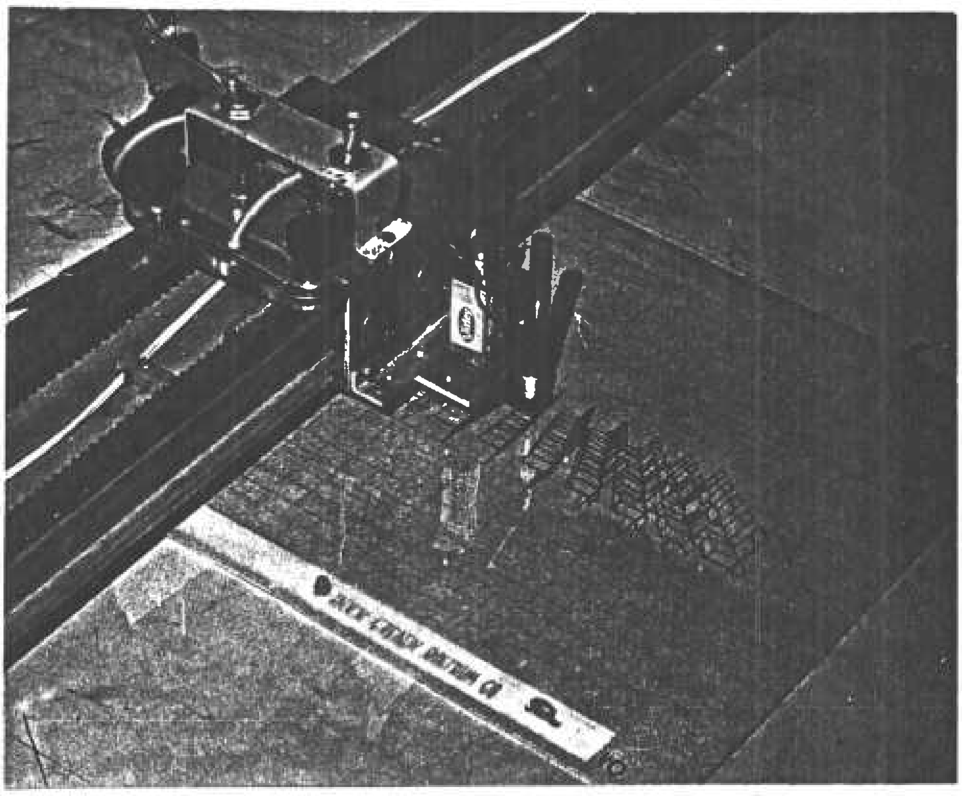


Fig 5.26 Plotter with cel on peg-bar

5.8 Camera-Compound Movements

Many of the movements achieved on animated films do not involve actual changes in the basic drawing. They are obtained by two basic movements from the standard rostrum camera compound: pan and zoom. By panning across a picture one can give the impression of movement across the screen. If two drawings are alternated, a reasonably long sequence can be obtained from just these two drawings. Zoom can be used to give the impression of an object approaching the viewer or vice-versa. These techniques are very powerful in producing animation from very simple drawings, and have been extensively used to produce low-cost educational films.

The windowing transformation described in the last chapter can be used to produce these movements on the screen. Pan corresponds to changing the position of the window and zoom to variations in the size of the window.

In normal animation pan and zoom are always well planned in advance, before committing the artwork to photography. These movements are determined from a field guide; representing the area that can be photographed on an animation stand, and plotted as N-S, E-W coordinates and field sizes on an exposure table. The field guide (Fig 3.4) consists of a transparent rectangular chart with a standard aspect ratio of 36 x 50 subdivided by horizontal, vertical and diagonal lines into divisions having the same proportions. There are usually twelve field sizes on a field guide, ranging from the largest 12 field representing the total area to the nearest close up in one field.

To be visually pleasing movements again should have an acceleration at start and a deceleration at the finish. The animator who has to plan all these non-linear movements well in advance can rarely visualise what the finished results will look like.

The camera movement programme simulates these triaxial movements with the display corresponding to the view seen through a camera viewfinder. The movement commands are input via the keyboard in the same format as the entries in a standard exposure sheet:

5 (6S - 4E) to 2 (10N - 10W)

represent a pan from bottom right quadrant to the top left-hand corner with a 2.5X zoom-in (Fig 5.27). For a field size FS centred at XC, YC the windowed coordinates (X', Y') on the display are given by:

$$X' = 12\,(X - XC + DX.FS/24)/FS$$

$$Y' = 12\,(Y - YC + DY.FS/24)/FS$$

where (X, Y) are the coordinates in 12 field and DX x DY is the size of the 12 field window which is pre-set by the user.

This program can be used to produce line tests and exposure tables. The line tests are used to check movements and to match sound track. The pan/zoom moves can be listed simultaneously as the sequence is being generated to produce a camera table. Fig 5.28 is an exposure table listing for the movement depicted in Fig 5.27 in 30 frames with 5 fair-ins and 7 fair-outs. All the necessary information from planning to the final shooting stage have been compiled, checked and tabulated. When the output is plotted on cels, the cels can be photographed in a fixed camera position as the plots are in 'subject zoom' format, i.e. the picture changes in size instead of the camera-object distance.

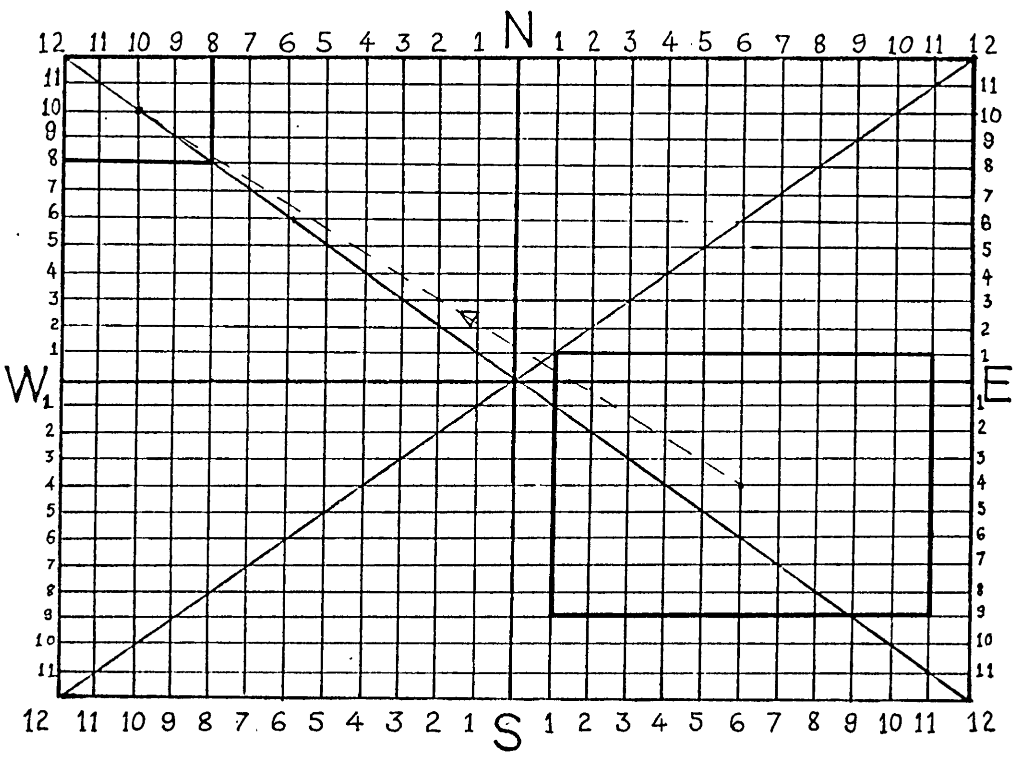


Fig 5.27 A 5(6S - 4E) - 2(10N - 10W) pan-zoom

```
COMPOUND MOVE:F1/XC1/YC1/F2/XC2/YC2/
5/-6/4/2/10/-10

NO. OF FRAMES/FAIR-IN/FAIR-OUT
30/5/7

FRAME              FIELD            E-W             N-S
 NO.                                PAN             PAN

   1               5. 000         -6. 000          4. 000
   2               4. 981         -5. 898          3. 910
   3               4. 925         -5. 600          3. 650
   4               4. 838         -5. 137          3. 245
   5               4. 729         -4. 553          2. 734
   6               4. 607         -3. 906          2. 168
   7               4. 486         -3. 259          1. 602
   8               4. 365         -2. 612          1. 036
   9               4. 244         -1. 965          0. 470
  10               4. 122         -1. 318         -0. 096
  11               4. 001         -0. 671         -0. 663
  12               3. 880         -0. 024         -1. 229
  13               3. 758          0. 623         -1. 795
  14               3. 637          1. 270         -2. 361
  15               3. 516          1. 917         -2. 927
  16               3. 394          2. 564         -3. 493
  17               3. 273          3. 211         -4. 059
  18               3 152           3. 858         -4. 625
  19               3. 030          4. 505         -5. 191
  20               2. 909          5. 152         -5. 758
  21               2. 788          5. 798         -6. 324
  22               2. 666          6. 445         -6. 890
  23               2. 545          7. 092         -7. 456
  24               2. 424          7. 739         -8. 022
  25               2. 309          8. 354         -8. 560
  26               2. 205          8. 905         -9. 042
  27               2. 119          9. 366         -9. 445
  28               2. 054          9. 712         -9. 748
  29               2. 014          9. 927         -9. 936
  30               2. 000         10. 000        -10. 000
```

5.28 Exposure table for rostrum

6 GAGS-3

This chapter describes GAGS-3, a three dimensional interactive graphics system. It has many features in common with GAGS-2, so only salient differences, specially related to data storage, display and three dimensional interactions are described here.

The system incorporates facilities for constructing three-dimensional data by digitising from two-dimensional representations of an object. The object can be manipulated in 3D but it is restricted to linear transformations. Display problems related to 3D are also discussed and some solutions presented. The systems can be used as an input processor for problems which require data in three-dimensional form such as pipe layout and routing, N.C. surface machining and finite element data generation.

Fig 6.1 shows the basic architecture of the system.

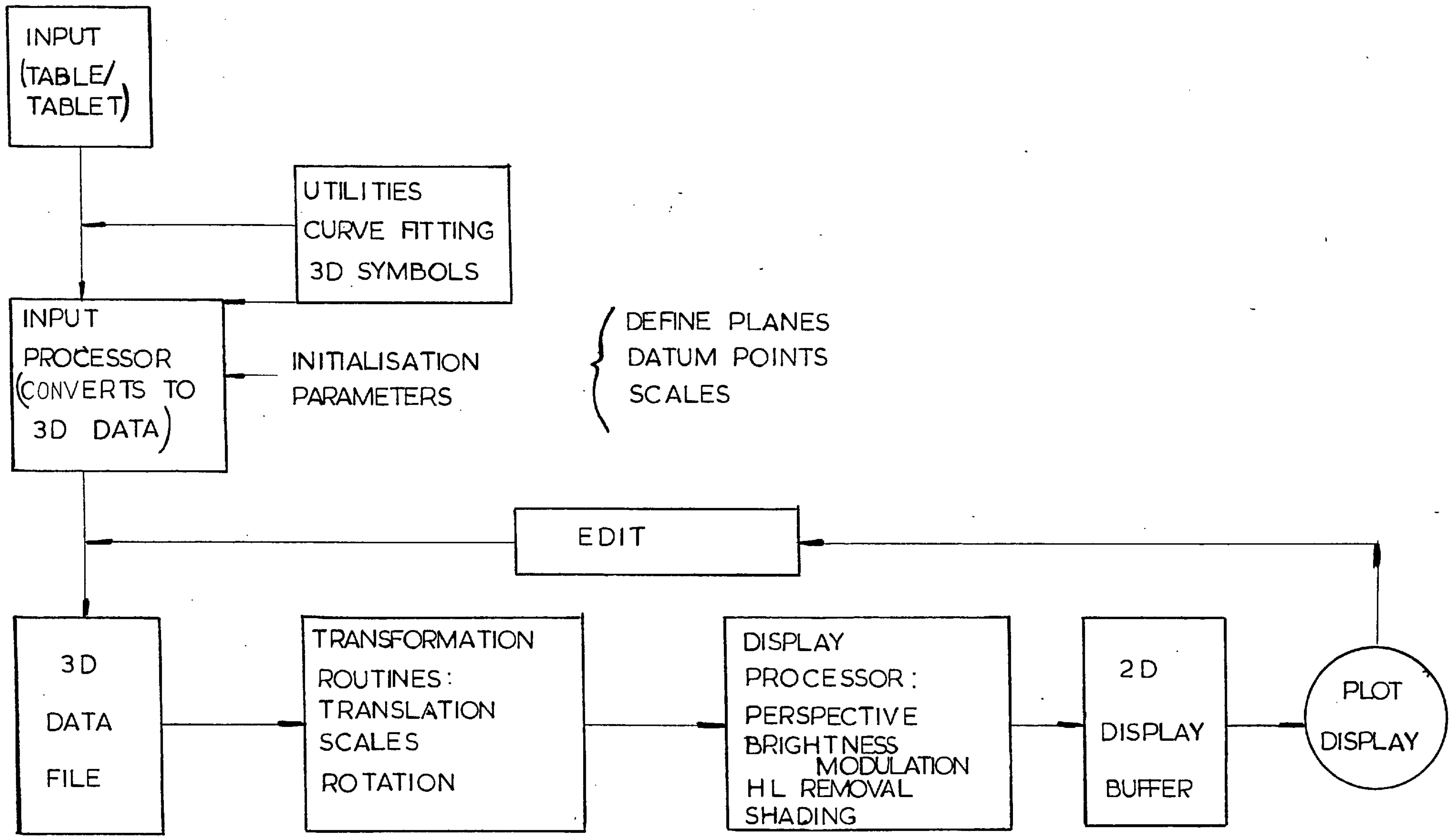


Fig 6.1 Basic system architecture

6.1 Data Structure

What is required is a data structure which links the displayed points and lines (both have to be referred to) with their exact dimensional values instead of their truncated screen coordinate values. In two dimensional work this link is provided by the direct mapping of the data base on to the screen, but this is not possible in three dimensions because the data must be transformed through some viewing algorithm. It must be possible to keep track of a full three dimensional description of the drawn part to enable design analysis of problems concerned with 3D geometry.

The essential property of these linked structures is to make it possible to break away from rigid sequential lists of coordinates and devise more natural computer structures to represent the geometry of the drawn part. The advantages such data structures have in the graphic processing of drawings are many. Of these, possibly the most significant is the way powerful editing algorithms can be devised which automatically update dependent data items that otherwise would have to be handled individually, normally a time-consuming procedure with interactive graphics. Another important attribute is the way structures can be used to overcome to some extent the difficulties of working in 3D when the display screen itself is 2D.

The file structure used in the system provides the links required between input data, displayed data and stored data. This enables the direct manipulation and editing of the three dimensional data base.

6.1.1 Data records

With a three dimensional data base the storage requirement is inherently increased by 50 per cent, with the corresponding increase in number of disc accesses to transfer data from disc to core and vice versa. The data should remain as compact as possible. The storage format used in GAGS-2 is wasteful because data is handled at word level, under the restrictions imposed by FORTRAN allocation of storage. By manipulating data at bit level it is possible to make it more compact by packing more information in the same amount of store.

Each line in the picture is represented by a string of vectors in $[X, Y, Z]$ cartesian coordinate form. The vectors define the position of the points describing the picture. A special code is also attached to each vector to qualify the data in $[X, Y, Z]$. Each record is made up of one integer and two or three real coordinates:

1 - $[I, X, Y, Z]$ for three dimensions

2 - $[I, X, Y]$ for two dimensions.

'I' is a packed single computer word and each coordinate stored as a real variable occupying two words of computer storage, therefore 5 to 7 words are required per record. A storage block in the DEC's RK-05 cartridge pack consists of 256 words, therefore there are 36 records of three dimensional data per block with 4 words to spare, and 51 two dimensional records per block with one word to spare (Fig 6.2).

Greater compactness can be achieved by storing the coordinates in single word integer format. The value of the coordinate data input from the table normally lies in the range of 0 to 7900 which is well within the range of the PDP-11 integer (-32760 to +32767). However this tends to impose restrictions on the value of the input scale, restricted to a maximum of 4, and prevents the use of realistic dimensions which is essential in some design problems, like in the design of ship hulls. Also there is a certain loss of precision in the use of integers to perform mathematical transformations, but this is relatively small and should not affect the results greatly. Although data is read in floating point format from the digitiser, the coordinate values are always represented by an integral number of units, thus no truncation errors take place at the input stage. Problems only arise with data varying widely in range being multiplied or divided by fractional numbers like sines and cosines. Single word integers reduce the storage requirements by almost 40 per cent, with the corresponding reduction in the number of disc accesses, but on the other hand, computations which must be performed in floating point mode are considerably slowed down.

These limitations can be partly overcome by the use of homogeneous coordinates, restricting the data values to the limits imposed by the hardware. However, the use of a fourth dimensional element increases storage requirements by 25 per cent in this particular case, so some of the immediate advantages of using homogeneous coordinates are lost. Homogeneous coordinates also introduce problems of their own. The use of multi-scale values can be confusing and if the data were to be normalised in order to be mapped on to any display, the values of the coordinates should lie in the range of

-1 to 1 or 0 to 1, thus precluding the use of integer storage.

It was mainly to enable normalisation of data, speed up mathematical operations and lift restrictions on the part of the user that the floating point storage of coordinates was chosen. The use of homogeneous coordinates for storage purposes does not result in any real advantage, but it can be used to simplify matrix operations. Also a single scale value was thought to be more convenient.

The 'I' code contains the parameters describing the status of each point (Table 6.1). In the PDP-11 implementation, the data parameters are packed in a single 16 bit word. Fig 6.3 shows the allocation of the bits in the word.

ICODE specifies the type of data contained in the $\left[X, Y, Z\right]$ locations, normally consisting of the coordinate positions of the data point. The code can have the meanings shown in Table 6.2, which are mainly 'pen-up', 'pen-down' indicators and macro delimiters.

LEV may have two meanings. It can be regarded as different cel levels in an individual frame or the display priority. When used as a display indicator it is compared with the value of a flag LEVSET to determine whether the data should be displayed. Only data with LEV below or equal to LEVSET is displayed. Therefore for LEV = 0 the data is always visible, for LEV = 3 only when LEVSET $\leq$ 3. This facility allows data of varying complexity to be contained in one file but only data up to a desired level need be displayed when the file is viewed.

INT represents the intensity level of a visible line. Up to 8 levels are permissible but with the Tektronix 611 storage tube only

3 are allowed and normally only 2 used: blank and full intensity.

LTYPE defines the type of line used. 4 line types are available: full, dotted, chain-dotted, dash-dotted.

PEN corresponds to the pen number on the plotting table. These can be pen with ink of different colour of different line thickness. Most plotters can cater for a maximum of four pens.

The positions for packing each individual parameter in 'I' are chosen so that the more frequently used ones are extracted more rapidly than the less often required ones. Appendix D describes the use of subroutine RECORD which is used to handle these parameters.

The parametric definitions can be modified or combined to give other types of information. For instance, only 8 intensity levels have been allocated, which are not sufficient for some modern displays, but enough for a matrix plotter with variations in grey-levels. For a display NPEN and INT can be combined to give up to 32 intensity levels since a display is unlikely to offer a choice of pen numbers. This is sufficient for most microfilm plotters which can cater for up to 30 intensity levels. For colour display INT may be regarded as a colour code. Some displays, like video monitors, allow a maximum choice of 8 colours including black and white. Combined colour-grey-scales systems can accept up to 256 grades. This can be arranged by combining any set of parameters to provide 8 bits of information.

Since the parameters defined in 'I' are directly dependent on the display hardware, they should be redefined for different configurations. The ones shown in Fig 6.3 pertain to the present system with Tektronix 611 display and the CADMAC plotter (no grey-scales, 4 pens).

Name	Range	Description
ICODE	0-15	Type of data (see Table 2)
LEV	0-7	Level/display priority
INT	0-7	Intensity level, colour, line-thickness
LTYPE	0-3	Line type
NPEN	0-3	Pen number

Table 6.1 - Data packed in I

ICODE	Meaning	Contents of X, Y, Z
0	end of data	
1	invisible vector	coordinate position
2	visible vector	coordinate position
3	start macro	X - macro number
4	end macro	X - macro number
14	parameter change	X = 1 LEV
		= 2 LTYPE
		= 3 NPEN
		= 4 INT
		= 5 all
15	null data	

Table 6.2 - ICODES

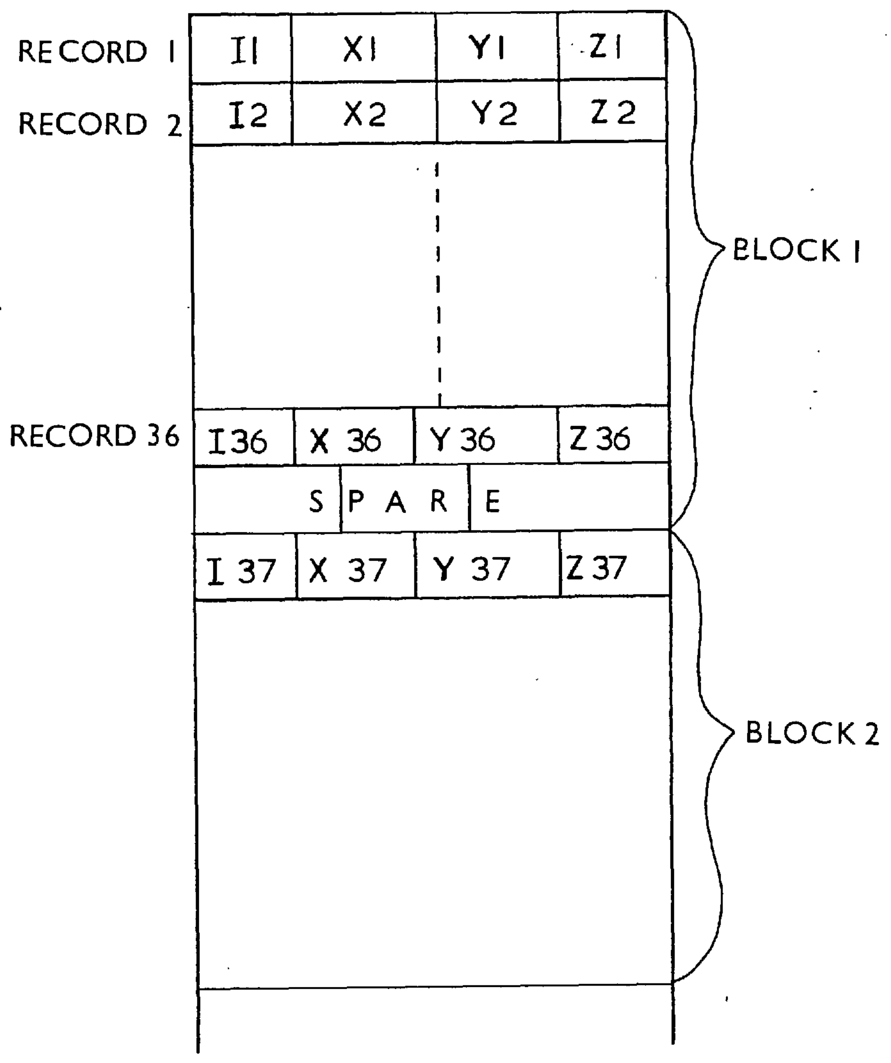


Fig 6.2 Storage format for data records

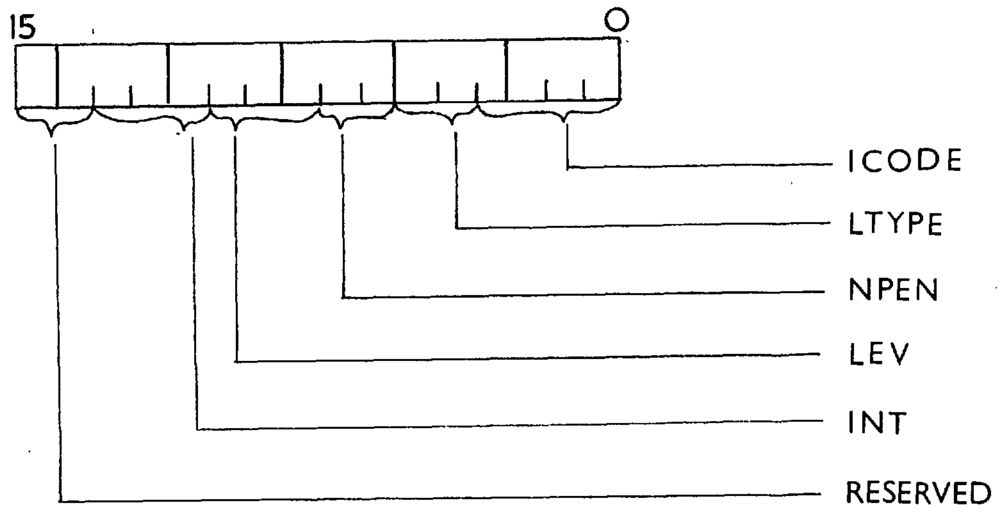


Fig 6.3 'I' storage format

6.2 File Management

The filing techniques used in GAGS-3 are similar to those described in section 5.3.2 for GAGS-2. The main differences are in the record access facilities and the existence of 3 special display/ data files.

Appendix D describes the subroutine used in the input/output of data records. Although the data is stored sequentially, the position of each record is indicated by a pointer corresponding to its address in the file. Random access of data is possible and the fact that each data record is self-contained in a qualitative sense removes the need to refer to previous qualifying data records. The usual mode of operation is auto-incremental mode. The I/O program automatically increments the pointer to the next record after each read/write operation. Random access is used mainly for editing operations.

Permanent files are manipulated in the same way as in GAGS-2 with 100 reserved files accessed from 100 squares on the menu, the remainder accessed by program only.

6.2.1 Display files

For display purposes, in three dimensional work, the data is arranged in three separate random access high-speed transfer files. These are known as FILE 1, FILE 2 and FILE 3.

FILE 1 - also known as the workspace, This file contains the true three-dimensional data describing the object. The file is made up of a sequence of $[\text{I, X, Y, Z}]$ data records as described in section 6.1.1. The coordinate records contain vectors indicating the position of points on the object relative to an absolute origin. The data can be viewed through a suitable projection or by direct viewing, represented by three orthogonal projections.

FILE 2 - This file contains the projected view of FILE 1. The data structure is identical to FILE 1, in the same sequence, but X, Y are in terms of screen coordinates and Z contains the actual spacial depth. The same sequence enables editing by a parallel search method of three dimensional data. A suitable routine converts data from FILE 1 to FILE 2 (e.g. perspective projection or contour mapping). This subroutine may be enabled at input stage with FILE 2 being created at the same time as FILE 1 or at some latter stage with revised parameters. The existence of this file should be transparent to the user.

The Z value is reserved for future use, related to visualisation aids (see section 6. 6.1).

FILE 3 - This file contains the actual blue-print of the digitised artwork. This is used mainly for input and checking purposes when the data is ambiguous in three dimensional viewing. Data is converted from FILE 3 to FILE 1 by an input routine. This can be invoked at a later stage but care must be taken to ensure that the same parameters are used. This file is made up of $\left[\ I, X, Y\ \right]$ records only.

The display of each file is controlled by the value contained in a switch ND:

ND = 0 no display
= 1 convert and display 1
= 2 display 2
= 3 display 3

Fig. 6.4 shows the three different display files containing a cube. FILE 1 and FILE 2 will give identical displays if the parameters are left unchanged.

Ideally views 1,2 and 3 should be shown simultaneously on the screen as in Fig. 6.5, with the top right-hand corner view made rotatable by a joy-stick or similar control. But due to the size of the screen this results in very small pictures, very difficult to visualise and lacking in detail. It is also difficult to manipulate data shown on part of the screen with a joy-stick when there is no selective erasure facilities. It is therefore preferable to display different views individually and benefit from improvements in display speed and clarity.

6.2.2 Input display modes

At the input stage 3 modes are possible, also determined by ND.

ND = 0 - No display at input (blank screen)

1 or 2 - 3D viewing with current parameters (3D cross in this mode). The coordinate display unit shows X, Y for positioning purposes only.

3 - Display blueprint - standard 2D mode.

All three files are created simultaneously irrespective of the display mode. The transformation sequence at input stage is shown in Fig 6.6. This is done at this stage because it represents a minimum time load, compared with the speed of reaction of the user.

When data is being digitised, a cursor or a cursor and a cross are displayed in non-store mode to indicate the relative position of the pen. When ND = 3 cursor indicates the position of the pen both on the table and screen. Only the cursor is shown and it allows exact positioning of data in FILE 3. When ND = 1 or 2 a cross is also displayed. The cross operates in three dimensional mode, i.e. its position corresponds to the actual position in three dimensional space as viewed through an appropriate projection. The position of the cursor is still two dimensional and relates to the table and the contents of FILE 2, as displayed on the screen. It allows careful matching when positioning points on a structure being displayed in 3D mode. This is possible because of the linked parallel file structure and is particularly useful when connecting already existing points using the find mode facility, or to search for points and lines in editing mode.

6.2.3 Direct viewing

This facility projects the data in the workspace on to three major planes which are then unfolded for presentation on the screen. The projection is equivalent to the transformation from FILE 3 into FILE 1 but in reverse. The three views occupy the same areas on the screen as in ND = 3 display mode, however they are complete orthogonal representation of the data. Reference axes indicating each plane of projection can also be included (Fig 6.5).

Each view is generated by taking the coordinate terms in FILE 1 corresponding to the plane being considered. Therefore from an [I, X, Y, Z] record only X, Y are used for the x-y plane and so on. The cross displayed on the screen represent the position of the datum point. Each view is centered within its own quadrant by shifting the data by an amount equal to $^1/2$ or $^3/2$ of the reference position.

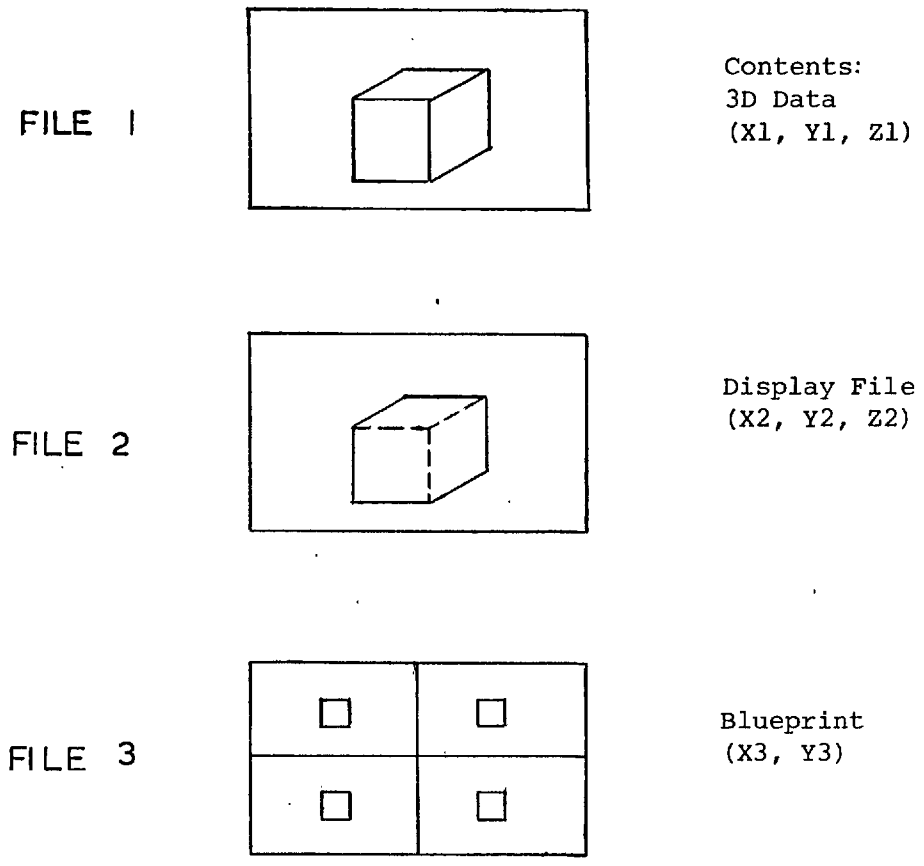
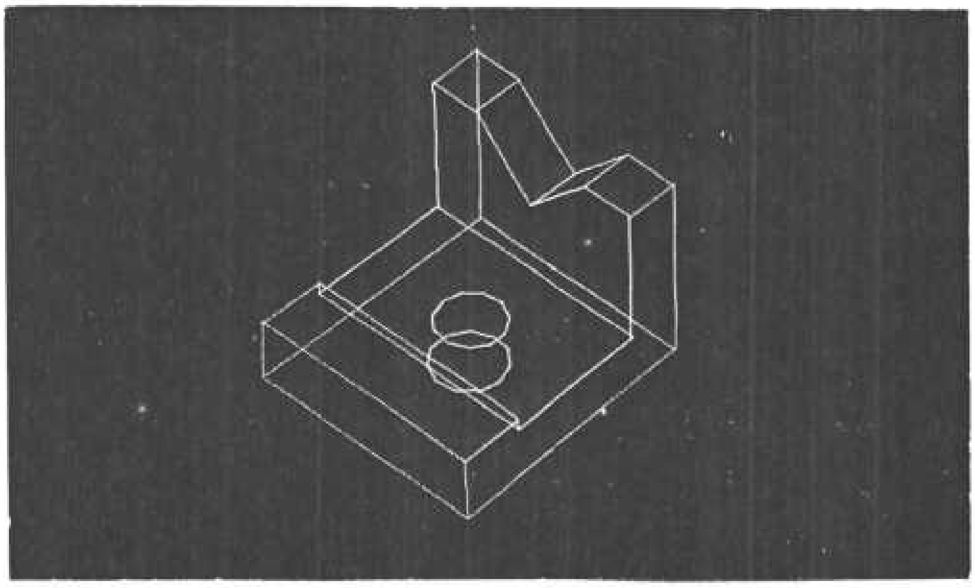
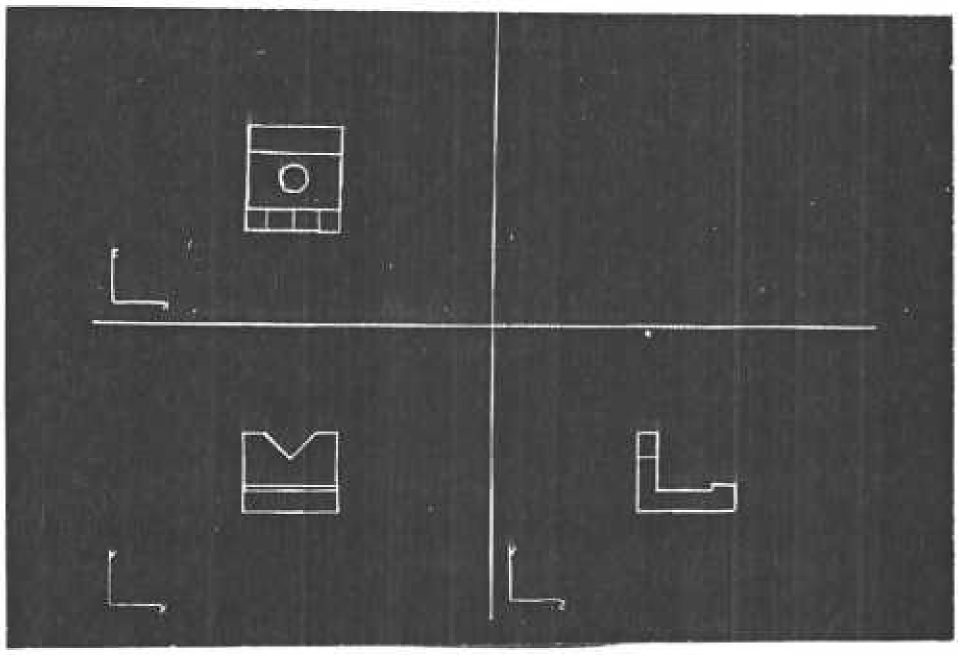


Fig 6.4 3D display files



(a) perspective projection



(b) orthogonal projection

Fig 6.5 Direct viewing

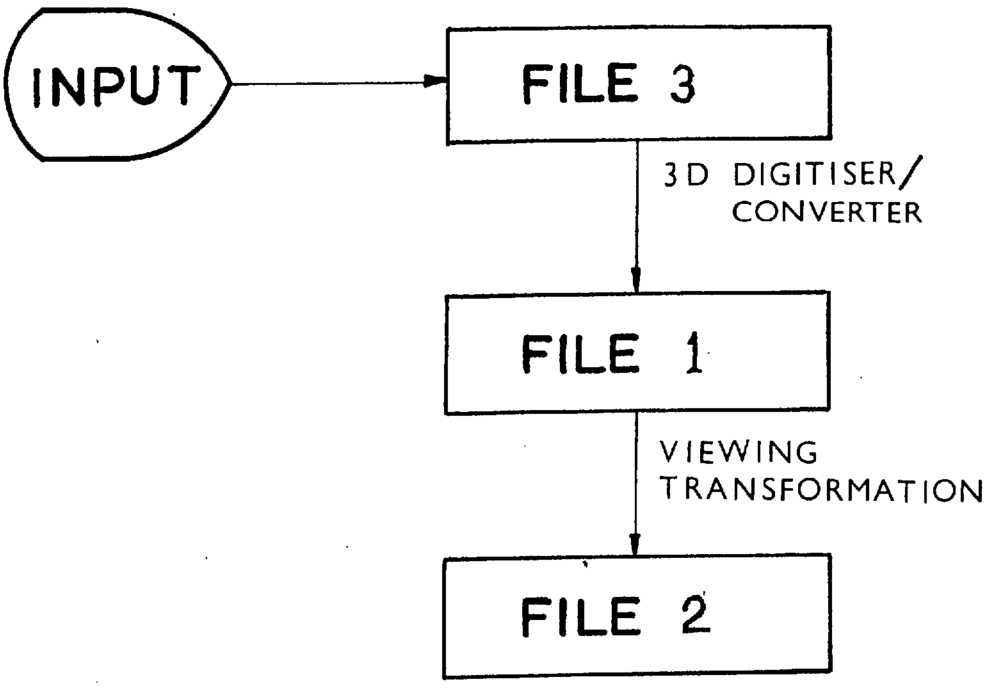


Fig 6.6 Input sequence

6.3 Input Techniques for Data Generation of Three-Dimensional Objects

Several techniques can be used to create a three dimensional data base from a three or two dimensional representation of an object.

Some are based on specialised tools, others on software techniques. Examples of hardware implementation are electronic probes on a physical model, laser beam or ultrasonic scan of the model (scan methods tend to produce excessive data, more than required to define the object), two pen digitisers, acoustic digitisers, etc.

When special hardware is not available software techniques must be used.

The system's standard form of input is based on orthographic views of an object. A facility also exists for input of data represented by contour maps. However there are other software techniques for inputing three dimensional data from plane pictures, but these were found to be unsatisfactory because of the nature of the artwork required.

Photogrammetry is a technique which consists in retrieving dimensional data from photographs (25, 26). It provides very accurate definition of three dimensional points but requires an already existing model in a form that can be suitably photographed. A more flexible transformation for inputing data from photographs is described by Parke (27) which consists of the solution of 6 equations for 6 unknowns. The equations can be represented by

$$\begin{bmatrix} x_1 & y_1 & z_1 & 0 & 0 & 0 \\ x_2 & y_2 & z_2 & 0 & 0 & 0 \\ x_3 & y_3 & z_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & z_1 \\ 0 & 0 & 0 & x_2 & y_2 & z_2 \\ 0 & 0 & 0 & x_3 & y_3 & z_3 \end{bmatrix} \begin{bmatrix} T_{11} \\ T_{21} \\ T_{31} \\ T_{12} \\ T_{22} \\ T_{32} \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

where (u_i, v_i) are the coordinate positions on the photograph corresponding to the points (x_i, y_i, z_i) in three dimensions. Therefore the exact position of at least three points must be known in order to solve the equations for the values of T_{ij}. Again, the data is required in photographic form and the technique does not apply to conceptual models. It is restricted to very special applications.

6.3.1 Orthographic representation of solids

There are several ways in which solids can be represented. For complicated objects if the shape is to be fully understood a number of pictorial views of a solid in different positions will be required. Orthogonal projections are a well-established method of representing solids. With this method the length, breadth and depth of a solid are fully represented by at least two views which are plane areas. In practice three or four main views will usually fully represent the most complicated solid.

Since many of the objects likely to be dealt with will have some sort of orthographic representation, it is useful to provide a means of input from such drawings. Because of the input facilities available many of these drawings can be in the form of rough sketches, which is the typical case with designers, when the object is usually just something in their imagination, at best there being a few sketches of the model on paper with many details missing. Either first or third angle projections may be used (Figs. 6.7 and 6.8), each with its own advantages and disadvantages. Only planes to the right of the basic x-y front elevation are considered.

Third angle projection has the advantage of having the front elevation situated on the bottom left-hand corner which is the most accessible area on the table. This is important because the front elevation is in most cases the most representative view of the object. Also two dimensional data can be positioned and digitised in this area without the need to redefine the position of the planes. The disadvantage of third angle projection is the confusion that may arise due to the conversion of the sign along the table X in the y-z plane.

First angle projection offers a more natural layout of the views, being easier to visualize. The main drawback is the very high position of the front elevation (plane A, Fig 6.7). For two dimensional data positioned low, the reference point needs to be redefined. The sign of z is negative in the plane view, therefore there is total inversion in the first plane which can be quite confusing.

The most natural layout for the drawings is a combined first and third angle projection system (Fig 6.9). The positions of the front elevation and plan are swapped to ensure that the x-y plane lies on the bottom left-hand corner of the table and that there are no sign inversions except in plane D, which is a rarely used view anyway.

To input data from orthogonal projections the table is partitioned into two or more areas, each corresponding to an x-y, y-z or z-x plane. Up to four views may be considered but normally only three will be available and in most cases two will suffice.

If a two pen device were available the two pens could be used to indicate a single point in two views simultaneously, thus defining the three dimensional position of such a point. With a single pen input device the same point must be digitised on two views. This can become a rather enormous task if each point must be repeated at least once. Means to simplify this have been devised.

When the system is operating in 3D mode the plane in which the cursor lies and the coordinates of the pen position are determined and converted automatically. To separate the planes a point known as the x-y datum (ZX, ZY) must be defined prior to any data input. This point corresponds to the common vertex of the planes. It is defaulted at the initialising stage but may be redefined at any time. A permanent cross on the screen shows the position of the datum point.

The cursor reads data in the two dimensional coordinates (X, Y) of the table so all input facilities described for GAGS-2 (section 5.4) are also applicable, though they now operate on several different orthogonal planes. The control mode facility is particularly useful as it allows the same point on different planes to be accurately positioned. In 3D mode only control 90 is available. This is the recommended mode of input when frequent changes from plane to plane are required.

The tests used in the input program to determine the spatial coordinates are very simple. The x-y datum determines the plane since it defines the intersection of any two planes.

Let (X, Y) be a point on the table

(x, y, z) the corresponding three dimensional coordinates of the point.

For $X < ZX$ and $Y < ZY$ the plane x-y is defined so

$x = X$

$y = Y$

If Y above y datum (ZY) then z is always given by Y-ZY.

x remains unchanged until X > x datum (ZX) when

for Y < XY   z = X-ZX

for Y > ZY   y = ZX-X

To input the first point or when a line is broken to change planes, two points on two different views are required to define the point in space. The first point sets the independent coordinate and the second determines the other two coordinates. From two points on the table four coordinates are always present, but for the repeated coordinate the second value overwrites the first one. Once the independent coordinate has been defined it is not necessary to reset it each time as it is automatically used as the trailing origin.

The analysis only dealt with the modified first angle projection, but similar results can be obtained by following the same reasoning with any other form of projection. The system can be easily modified to accept data in another layout by just changing a conversion subroutine in the data input program.

6.3.2 Contour maps

Another way of representing three dimensional data, particularly in land survey techniques, is contour mapping (Fig 6.10).

Because contours are a single surface two dimensional representation they do not represent a problem in terms of input. The facilities in GAGS-2 are sufficient to digitise contour maps. Appendix E contains a user guide to the contour mapping package.

Viewing of a surface requires a different presentation since it cannot be represented by line drawings as in the case of solids. Contour maps only contain the essential features of a surface. An interpolation technique is required to derive the information between the contour lines.

The surface generation program computes griddied data from the randomly spaced input data. The input data is sorted along the X and Y axes and made to lie on the nearest grid node. Because of the grid factor some nodes may contain more than one data point. Superimposed grid values are progressively averaged. Undefined grid points are obtained by linear interpolation of the set points. The interpolation is first performed round the edge of the plane defined by the grid points. This provides the edge profile of the surface and makes possible the second interpolation along the X direction. However, only points with different Z values are used to prevent flat plateaux. The interpolation is completed along Y for the remaining void nodes.

The grid with Z coordinates defined for each point is in a form suitable for surface representation. The program then generates a semi-oblique view of this data (Fig 6.11). Oblique views do not have the realism of perspective but have the advantage of straight edges so two or more views can be butted together to form one picture.

This facility combines two of the most common methods of representing functions of two variables. Whereas contour maps emphasize quantitative aspects and are useful for extracting numerical information, surface projection depicts the function in such a manner that its properties are most easily visualized.

6.3.3 Picture construction facilities

The facilities are essentially the same as those described in section 5.4 for GAGS-2, namely control mode, drive move, continuous digitising, gridding, curve fitting and symbols.

Because the table surface now represents more than one plane in space, the facilities also operate in more than one plane. They can be used to digitise data from any of the major planes defined on the table.

Curve fitting operates in three dimensions fitting through points in space. The lines are represented by parametric equations of the form

$$X = f(t)$$
$$Y = g(t)$$
$$Z = h(t)$$

The cubic spline technique described in Appendix C applies to both two and three dimensions, and for that matter, to any number of dimensions.

6.3.4 Symbols and macros

Symbols are created by a method similar to that described for two dimensional work. There are certain restrictions imposed by the limitations of interacting from plane surfaces in three dimensions. The symbols are represented by basic plane geometry shapes, such as circles, rectangles and polygons, but may be repositioned in space by using the manipulation facilities incorporated in the system. Appendix E contains recommendations on the creation of symbol overlays.

There are two useful properties in many solids which may be used to generate geometric representations of an object very quickly based on simple input. These properties are:

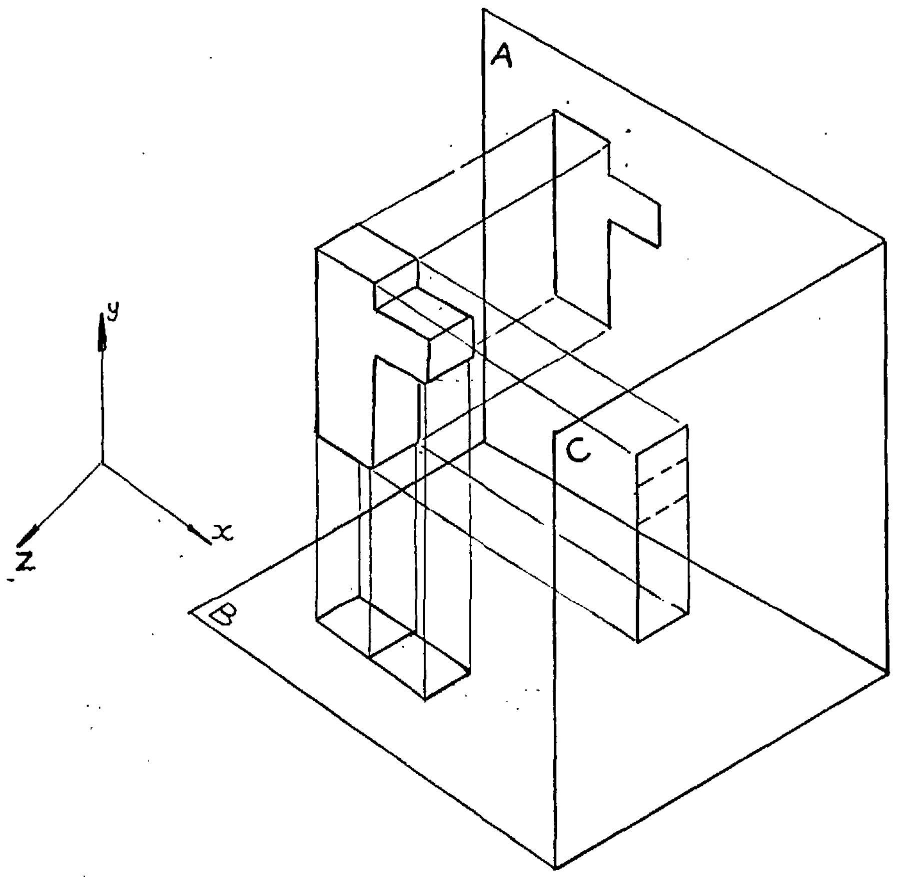
Uniform cross-section

Axi-symmetry (solids of revolution).

The input is two dimensional and may be regarded as a special macro to be manipulated by the solid geometry generation routines.

For objects with uniform cross-section, the cross-section defined in two dimensions and the length of the object are sufficient to give a complete description of the object. The cross-section is digitised in the x-y plane and the program automatically generates the solid (Fig 6.12).

Axi-symmetrical objects require the generatrix and the axis of revolution. The generatrix (Fig 6.13) defines the surface and is input by digitising on the x-y plane. The axis is given by a single point defined on the plane. For tilted axis the data is rotated first. The directrix which guides the revolution can also be displayed either as a circle or a polygon (Figs 6.13 b, c).



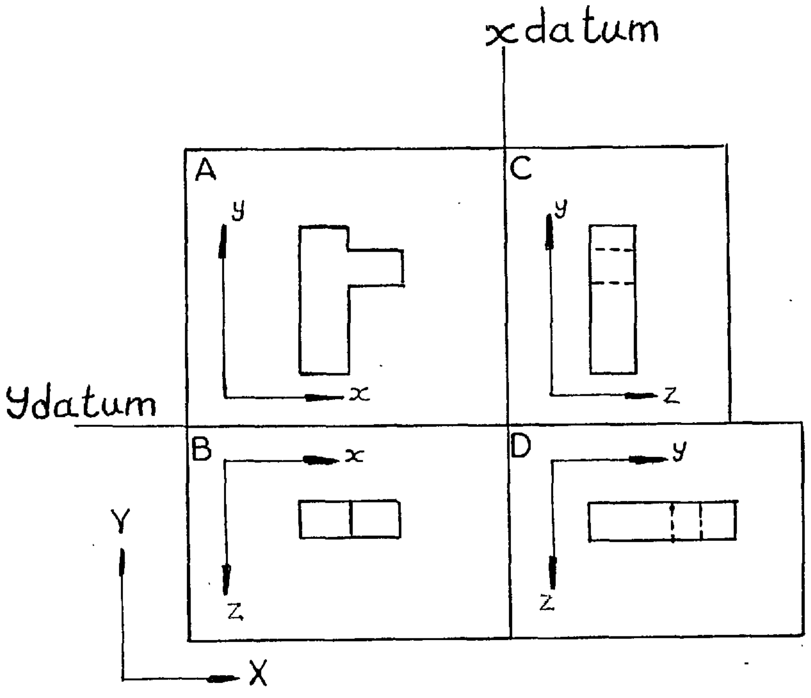
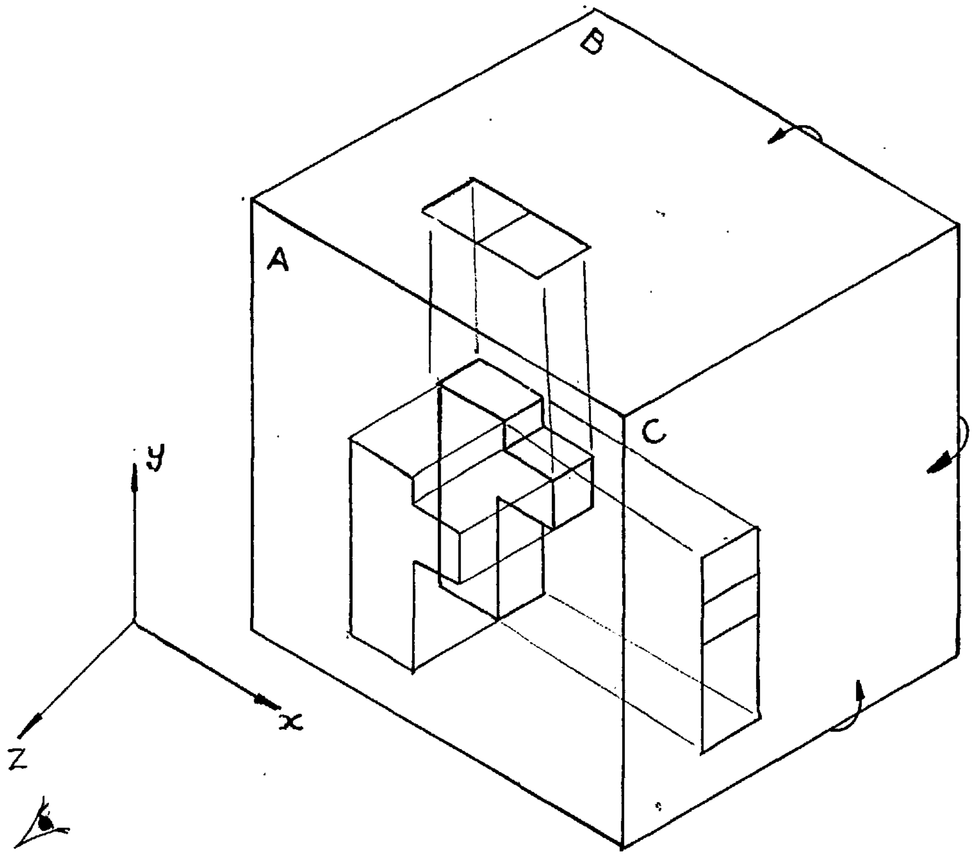


Fig 6.7 First angle projections



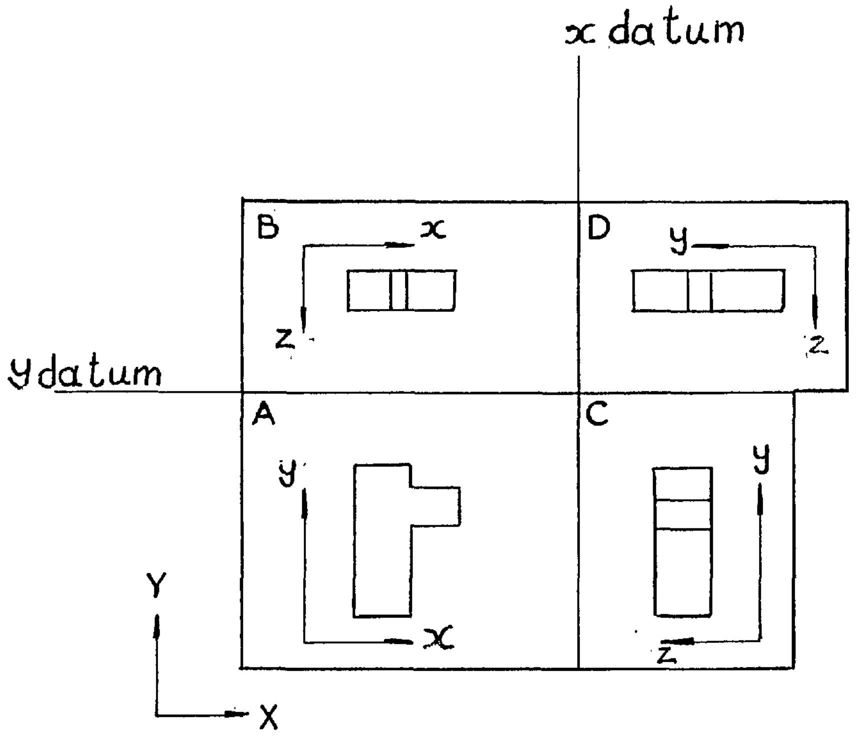
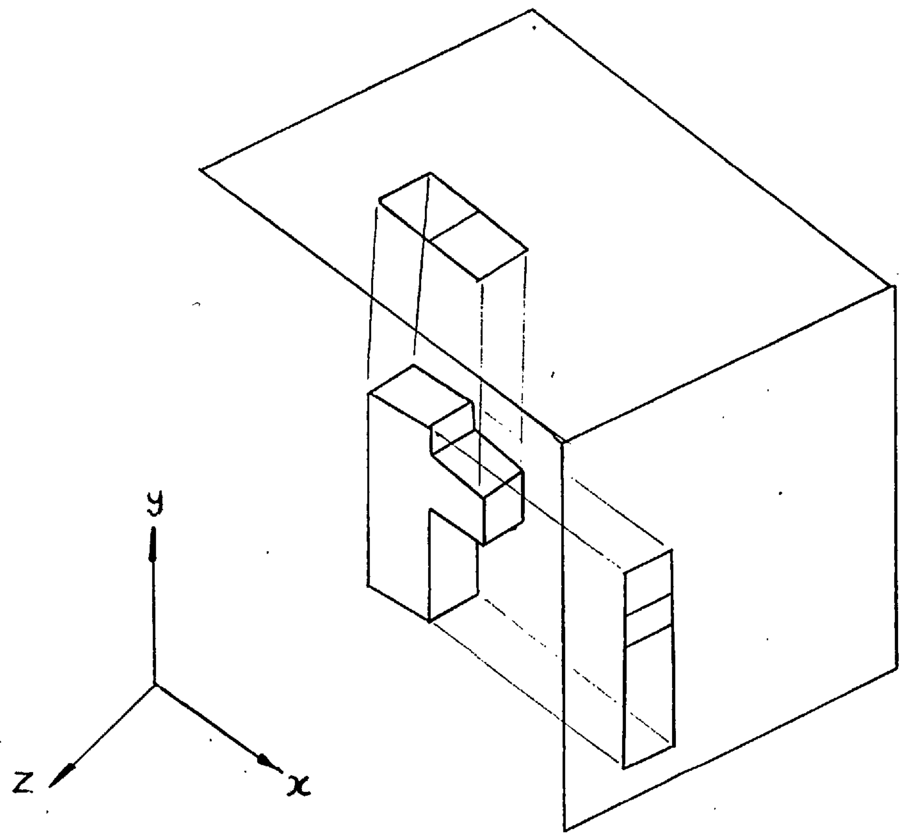


Fig 6.8 Third angle projections



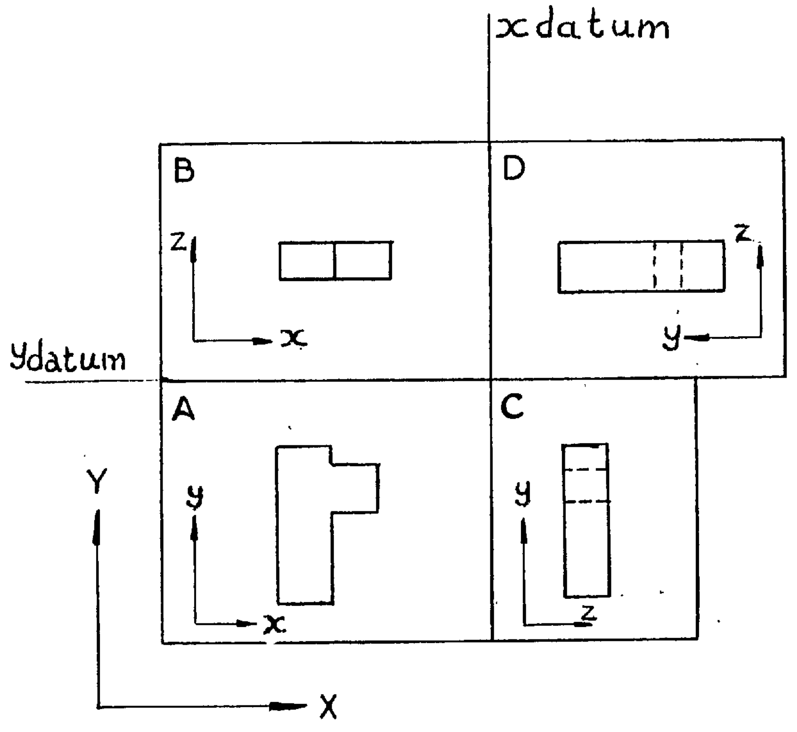


Fig 6.9 Modified first angle projection

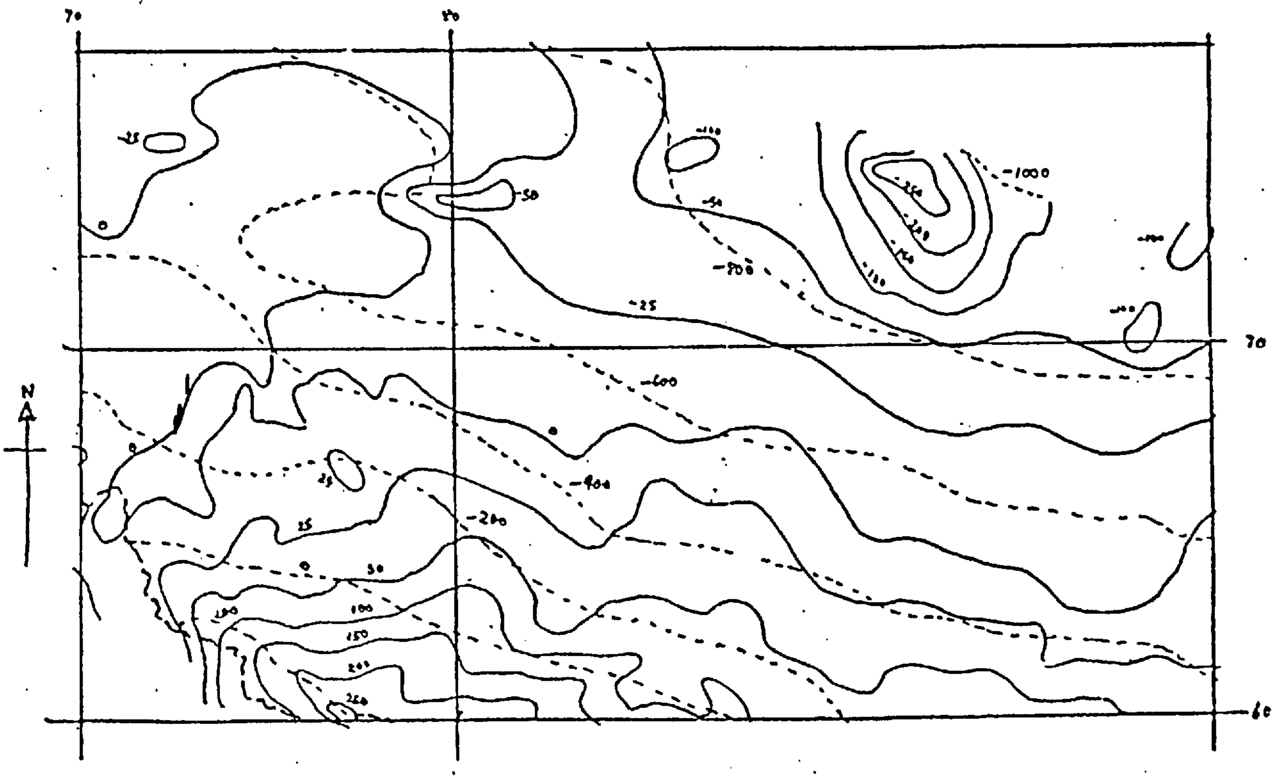


Fig 6.10 Contour Map

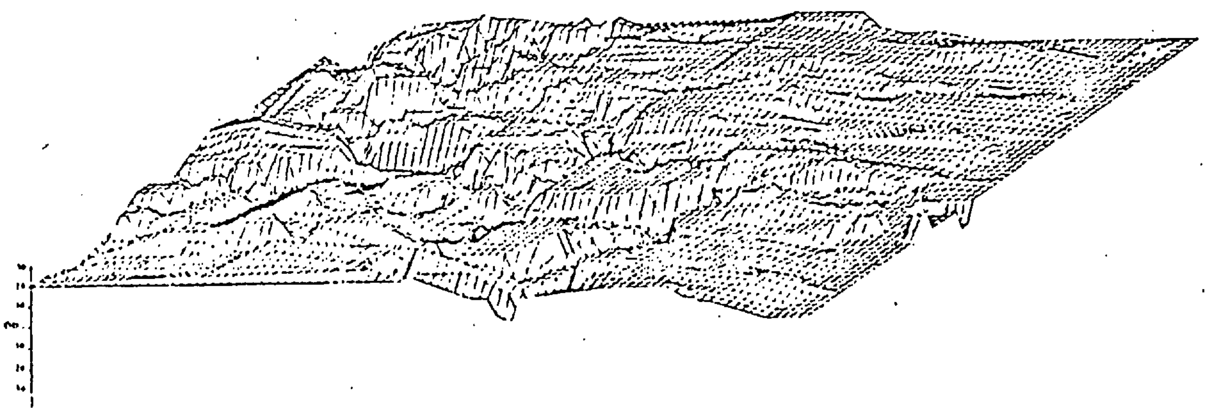
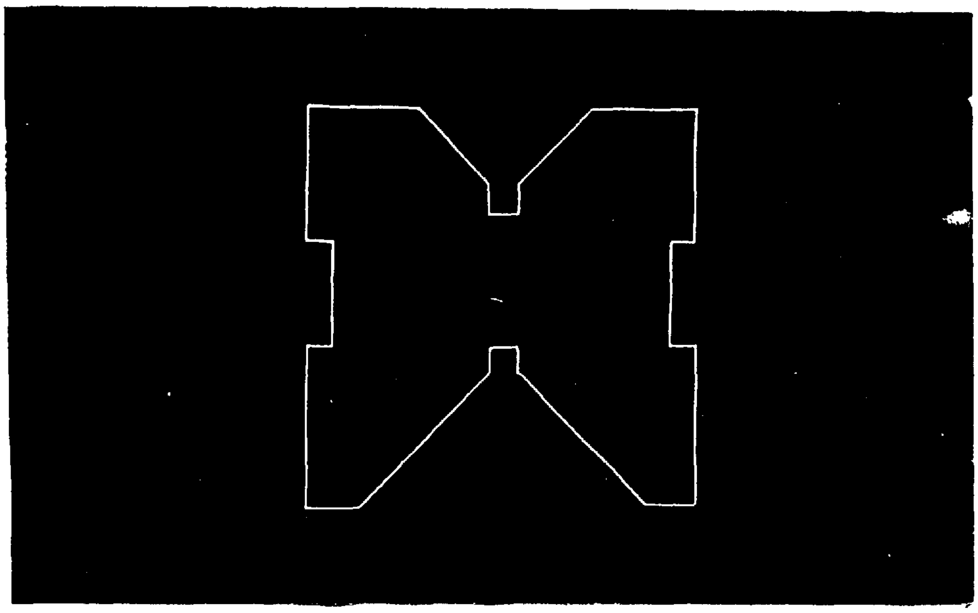


Fig 6.11 Oblique view of surface


(a) cross-section

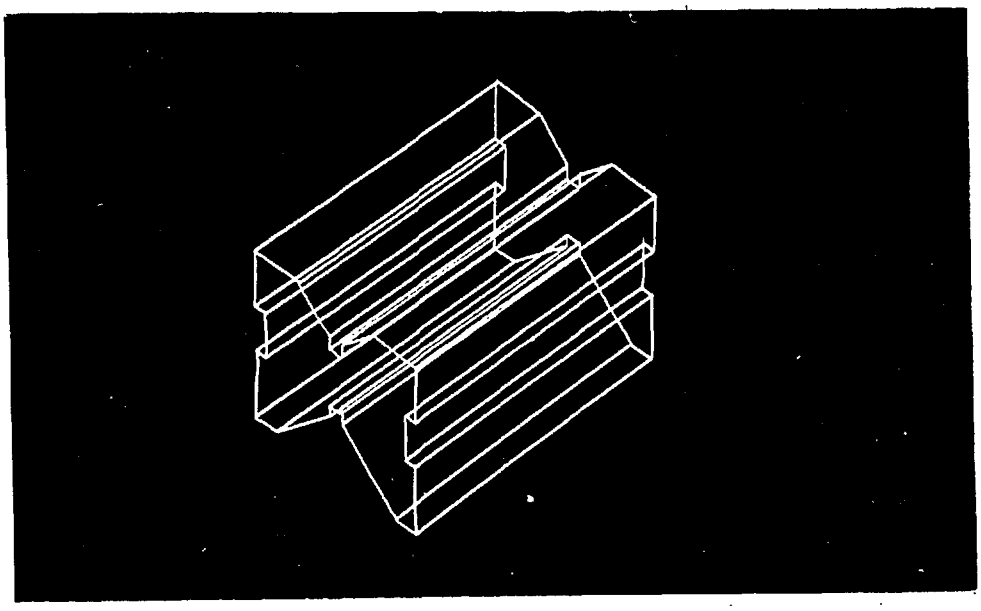
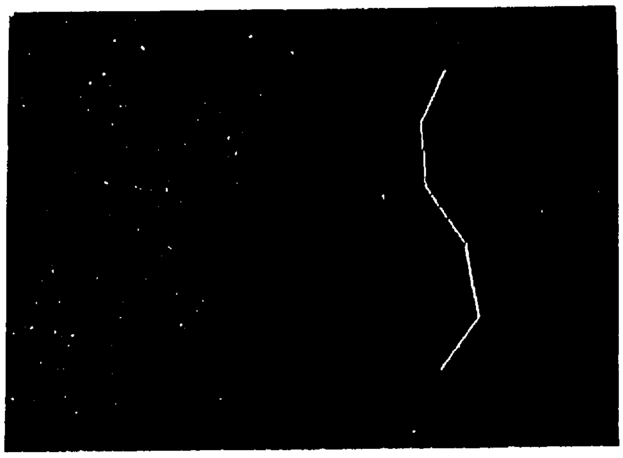
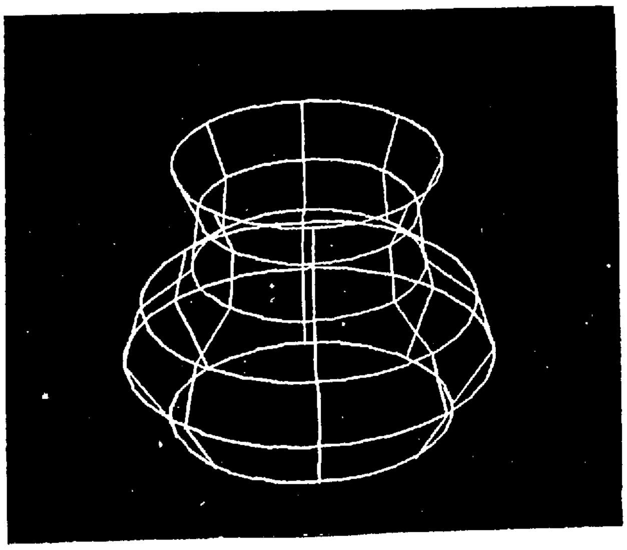

(b) object

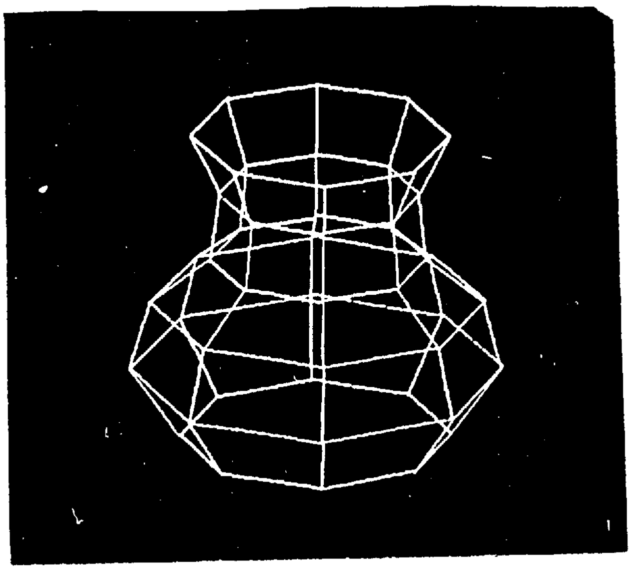
Fig 6.1 2 Uniform cross-section



(a) Generatrix



(b) Circular Directrix



(c) Polygonal Directrix

Fig 6.13 Solids of revolution

6.4 3D Editing

As mentioned earlier, one of the advantages of a data structure based on the different files is to be able to edit the data base on-line. The multiple file structure allows data displayed on the screen to be related to the dimensional values contained in the data base.

FILE 2 contains the data which is displayed on the screen. It can contain the actual screen coordinates but normally consists of scaled values of these coordinates. The editors use this file to search for the desired data element. Each editor uses a different algorithm to search and find the data depending on the editing mode in operation. The search can be carried out in parallel using both FILE 1 and FILE 2, but since there is a one to one correspondence between two files, it is normally done on FILE 2 using auto-sequence mode, i.e. the record position is automatically incremented as the file is scanned. When the correct record or block of data has been found the modification is performed on both files. The individual records can be randomly accessed, modified and written back in the file (Fig 6.14).

6.4.1 Graphical line editor

The line editor uses the distance of the points from a line to define the line. Two points are required to define a line. Let these be A and B from Fig 6.15.

The equation of a line is

$$ax + by + c = 0$$

therefore $y = -\frac{a}{b}x - \frac{c}{b}$

$\frac{a}{b}$ and $\frac{c}{b}$ can be defined as new constants.

let $b = -1$

$$y = ax + c$$

where a is the slope given by $\frac{y_a - y_b}{x_a - x_b}$

The unit vector normal to the line is $\bar{n}$

$$\bar{n} = \left[\frac{a}{\sqrt{a^2+b^2}} , \frac{b}{\sqrt{a^2+b^2}} \right]$$

Vector $\overline{PA} = \left[x_p - x_a, y_p - y_a \right]$

Therefore distance from P to line is $\overline{PA}.\bar{n}$

$$DIST = \left| \frac{a(x_p - x_a) + b(y_p - y_a)}{\sqrt{a^2+b^2}} \right|$$

from above $b = -1$

$$DIST = \left| \frac{a(x_p - x_a) - (y_p - y_a)}{\sqrt{1+a^2}} \right|$$

where $a = \frac{y_a - y_b}{x_a - x_b}$

Therefore given the position vectors of the points A, B, P the distance from any point to the line defined by the other two points can be obtained from the above relationships.

For line editing purposes, the points A and B are obtained from the display file. Two points are considered at a time, provided they are indicated to be linked to form a line. The point P is supplied by the user by positioning the cursor near the desired line. The algorithm scans the data and calculates the distance of the point from the lines contained in the data. Once the line near the point has been found it is trapped and continuously refreshed on the screen until the user either deletes it or advances the search.

6.4.2 Point editor

Because of the nature of the display, random point editing is only possible along the major orthogonal planes.

The point editor uses the direct viewing facility described in section 6.2.3. Since the table restricts the movement of the cursor to the major planes only, points can only be dynamically moved along those planes.

Point editing comprises of two main tasks: first find the point to be moved, second define a position where the point is to be moved to. These two tasks use the same facilities, essentially search the workspace for the position of a point being moved or a point to be moved to.

To find the point the user must define a point in space as near as possible to the desired point. In order to do this two points are required one on each plane to define a point in space. The first point is digitised in one of the planes and will represent the trailing point. The second point is digitised in another plane to give the three dimensional position by combining the four coordinates, in the same way as in input digitising. The program then searches the workspace for the data point nearest to the entered point within a given range. If no point exists within the range a message is flashed with 'NO NEAR POINT', otherwise the record position of the found point is stored for later updating.

The placing operation offers two options: the user can just define a point in space as above and it will be used as the new position of the data point or may search for another point in workspace and move the edited point to that position.

Each individual operation is controlled by a different button on the pen:

Button 1 is used to get and place the point.

Button 3 defines the trailing point. One trailing point is sufficient for the operation on the same plane.

Button 7 finds point. Has same effect as 1 at get point stage.

Position of the pen relative to the object is indicated by a cursor on the screen. By editing points along the main planes alternately, it is possible to move these points in any direction in space.

6.4.3 Find point editor

This editor operates on points but is not restricted to movements along one of the major planes. However points can only be moved to positions where a point has already been defined, i.e. the point is positioned by moving to a found point. For this facility the search is performed on the display FILE 2. The search algorithm is similar to the one above and the positioning operation is the same as the one above using button 7. Only one point needs to be digitised because the display view is a two dimensional perspective projection. For views containing ambiguities either the eye position or the orientation of the object can be changed to obtain a more unobstructed view. Search is performed on FILE 2 but both FILE 1 and FILE 2 are updated.

6.4.4 Symbol editor

The symbol editor allows system or user defined symbols to be selected and deleted from the workspace. The search algorithm is similar to the one used in GAGS-2 but the search is done with FILE 2 whilst the modifications are performed to both FILE 2 and FILE 1. The macro is found by digitising a point within a box containing the macro. When the macro has been found the box is refreshed continuously on the screen to indicate that the system awaits action from the user: delete or advance search.

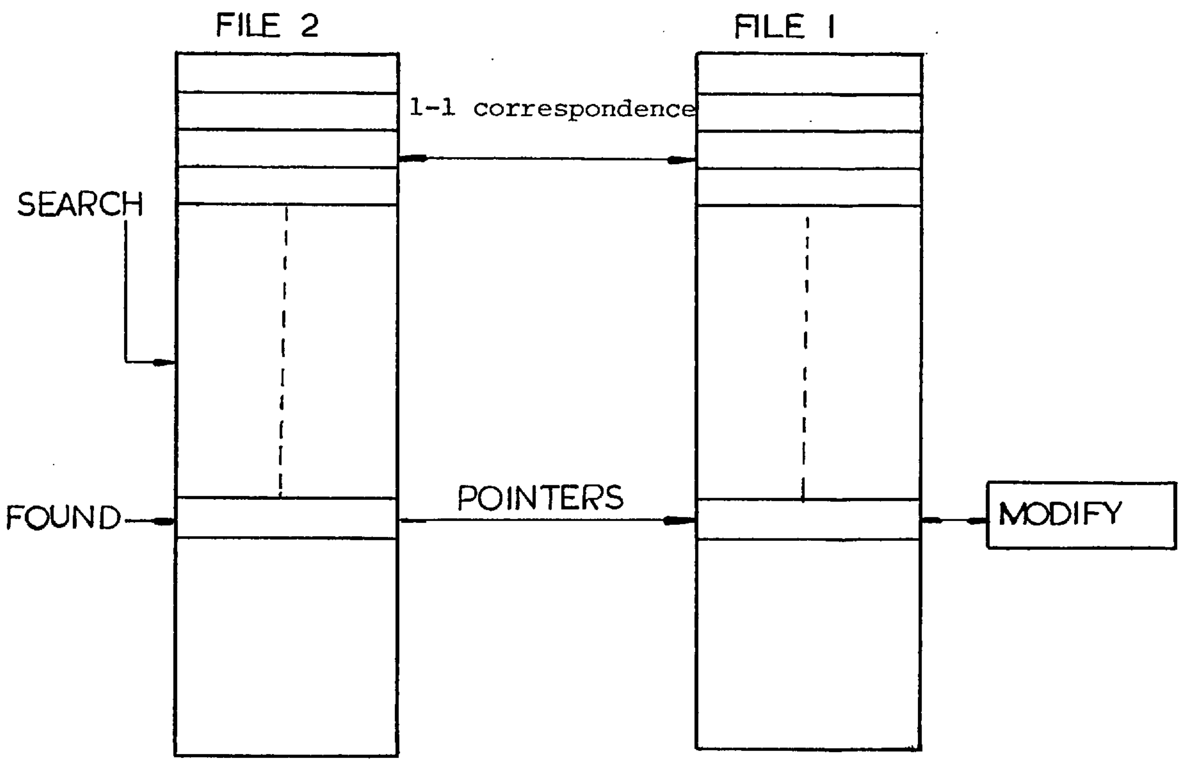


Fig 6.14 Parallel editing technique

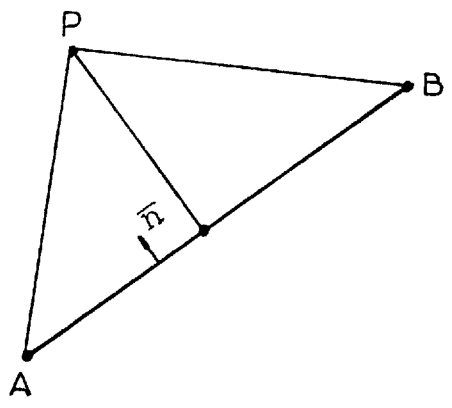


Fig 6.15 Distance of a point from a line

6.5 Three-dimensional Transformations

These transformations are performed on the data by matrix operations similar to the ones described in section 5.6. Either 3 x 3 or 4 x 4 matrices may be used, depending on the type of manipulation required.

6.5.1 System of reference axes

Before considering any transformations in three dimensions, it is necessary to define a system of reference axes and adopt a convention for the direction of rotation.

The conventional right-handed reference set of orthogonal axes, as shown in Fig 6.16 is used.

The x-y plane is chosen to correspond to any flat working surface, e.g. the table, the tablet or the viewing screen. The z direction is always forwards, towards the observer.

The angle of rotation θ about any axis is taken to be positive when the rotation is anti-clockwise and negative when clockwise, i.e. the rotation is said to be positive when in the sense of a right-handed corkscrew. (Fig 6.17).

The rotation is given by the matrix

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

with θ measured in the anti-clockwise direction.

The rotation refers to the set of axes on the plane of rotation. For the rotation of objects in a space with fixed set of reference axes the opposite applies and the matrix becomes

$$\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

6.5.2 Linear transformations

Contrary to two dimensional work, transformations in three dimensions are mainly linear in nature, consisting of:

Scaling

Translation

Rotation (about one or more axes).

For three dimensional data $[x, y, z]$ the transformation can be represented by

$X' = XT$

where $X = [x, y, z, 1]$ the original coordinates

$X' = [x', y', z', 1]$ the transformed coordinates

and T can be represented by a single or concatenated 4 x 4 matrix:

$$T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ t_1 & t_2 & t_3 & 1 \end{bmatrix}$$

where r_{ij} are the terms of the rotation matrices and t_j the translation offset.

The r_{ij} terms normally consist of a single or a combination of rotation matrices

(a) Rotation about z axis (Fig 6.17).

$$R_z = \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

(b) Rotation about y-axis (Fig 6.18)

$$R_y = \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{bmatrix}$$

(c) Rotation about x-axis (Fig 6.19)

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix}$$

R_x, R_y, R_z can be concatenated to give a general rotation matrix R, however, the order in which the individual matrices are combined will affect the results thereby achieved.

The scaling transformation is given by:

$$S = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & S_z \end{bmatrix}$$

For equal values of S_i the scaling is linear, otherwise distortions will be introduced into the system.

These transformations allow data to be repositioned anywhere and in any orientation in space. The translation, rotation and scaling effects can all be combined into a single transformation matrix by concatenating the individual matrices. However, the same effects can also be achieved by varying the viewing parameters used to define the object-observer geometry in the perspective transformation.

The equivalent effects: pan, rotation and zooming can be obtained by varying the observer's position and orientation relative to the object. This eye movement technique represents a more economical way, in terms of computing time, of obtaining movements from a static object. As the data base remains unaltered and only the viewing transformation, which is always required, irrespective of the parameter values, is performed. The joy-stick function described in section 6.8 uses this technique to provide very quick means of manipulating the display.

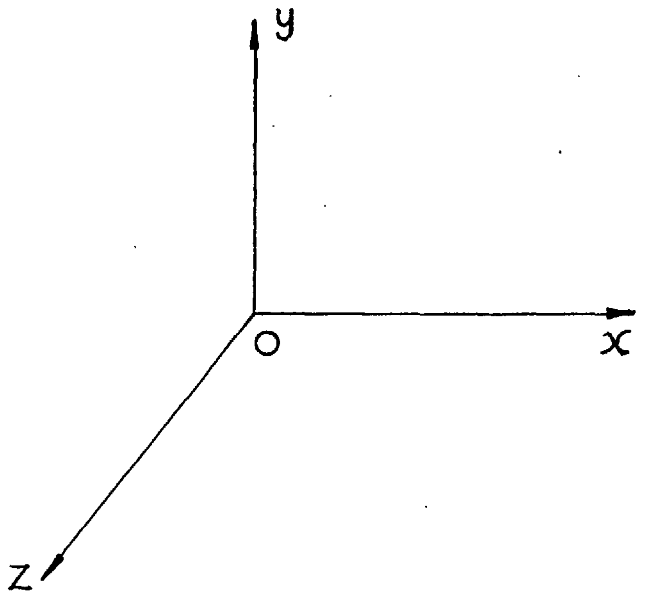


Fig 6.16 Right-hand reference set of axes

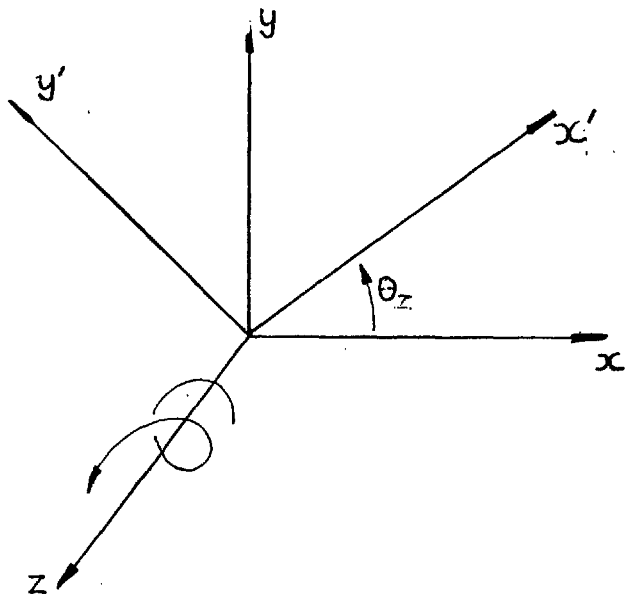


Fig 6.17 Rotation about z axis

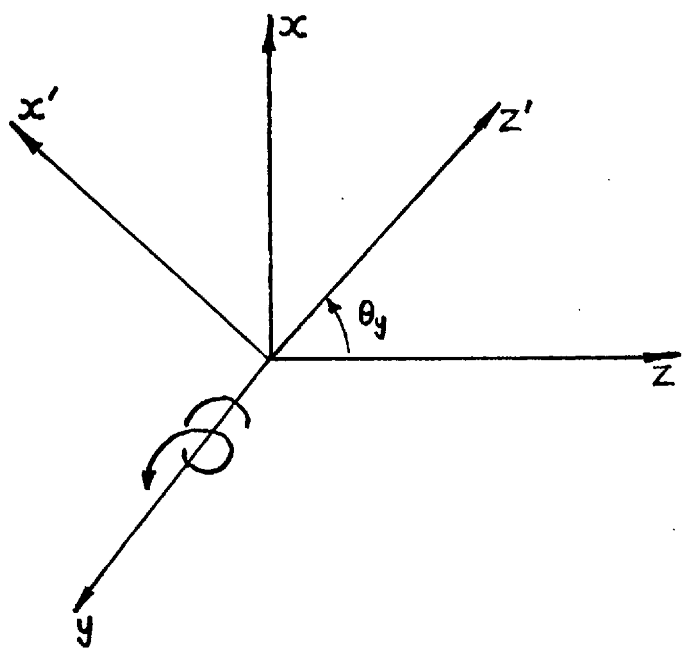


Fig 6.18 Rotation about y axis

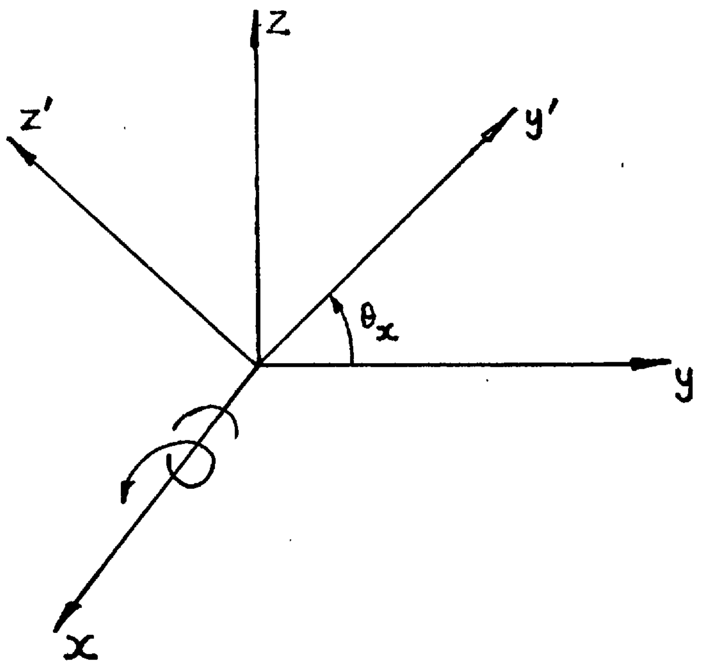


Fig 6.19 Rotation about x axis

6.6 Display

The introduction of a third dimension results in an increase in the number of processes involved in converting data from a basic structure to drive a display terminal. Two major stages are involved: compilation and processing (Fig 6.20).

In the compilation stage the data file is first scanned by a viewing algorithm, an output routine which generates function calls to the appropriate subroutines (scaling, perspective, windowing, clipping). Then, in response to these function calls, the display compiler interprets the stored data and creates a display file. The cumulative effect of these two processes is to generate a second data structure from the first.

In the second stage, the display file, which contains simple commands for the display, is scanned by the display processor, which then generates the drive signals for the CRT. The instructions are simple x-y drive signals, but the processor may incorporate some hardware character or symbol generation facilities.

These two stages are very similar, both involving a scanner and an interpreter, and can be combined into one, so that a set of data serves both purposes. With a storage tube a conventional memory for storing the display file is not essential, since the image is only displayed once. Because of the lack of selective erasure facilities the whole screen must be redisplayed to incorporate any changes. However, the use of a separate display file does have the advantage of enabling the use of the concept of segmented display files, which reduces the amount of compilation involved each time the picture is selectively modified. For example, a background is only compiled once and only the moved elements need be recompiled; in macro manipulation only the affected macro, or modified segment is recompiled.

In GAGS-3 the data and display file are stored in FILE 1 and FILE 2 respectively. FILE 2 does not fall within the exact definition of a display file, because it contains more than simple instructions for the display processor, and it also contains data that lies off the screen area. The main purpose of this file is to permit faster display at different window sizes, without the need of repeating viewing transformations, to enable interactive on-line editing of the data base by a parallel search technique (section 6.4) and to be used in visualization tests for three-dimensional viewing.

6.6.1 Visualization aids

Here one is talking about objects in a three dimensional world, a world to be observed through a person's sense of sight. Since people perceive objects in a certain way, they have become adapted to expect to see them that way and only with difficulty can other modes of perception be used to convey precise information about the objects.

Because one readily produces unfamiliar shapes with the computer, visual aids for perceiving depth are essential. When views are rotated, the succession of two dimensional projections shown create a good impression of a three dimensional object. However, in order that the work may proceed, the user needs help to interpret static two-dimensional projections of three-dimensional objects.

There are several ways in which the visualization of the objects can be improved:

(a) Perspective transformation

(b) Brightness modulation

(c) Hidden-line removal

(d) Shading

(e) Movement.

The first two techniques apply mainly to wire-frame models, whilst (c) and (d) are exclusive to solid objects. This section is mainly a survey of such techniques and how they can be incorporated in future work.

(a) Perspective transformation

In a real three-dimensional world, the most helpful phenomena for depth perception is, perhaps, perspective. Based on elementary laws of optics, it can be represented mathematically, and is therefore most suitable for computer manipulation.

Perspective is used extensively throughout the system and is regarded as the standard way of viewing graphical output. An advantage of perspective is that, it is defined by only a few parameters, related to the position of the observer. A new perspective view can be generated by simply redefining these parameters, leaving the object unchanged.

Once the data concerning location and direction of the observer's eye is defined, a program sets up and positions a viewing plane, at a given distance from the observer, on which the perspective view of an object can be projected.

For an observer with the eye position at $\overline{E}\left[E_x, E_y, E_z\right]$, direction of line of sight at angles $\left[\alpha, \beta, \gamma\right]$ to the major axes, the point $\overline{P}\left[P_x, P_y, P_z\right]$ in space is mapped on to a projection plane, at distance D from observer, by the following relations:

$$X = (S_x - R_x) \cos\gamma - (S_z - R_z) \cos\alpha \;/\sin\beta$$

$$Y = (S_y - R_y)/\sin\beta$$

where $\left[X, Y\right]$ are the coordinates of the projected position of the point $\overline{P}$. $\overline{S}\left[S_x, S_y, S_z\right]$ is the point of intersection of the line of sight on the projection plane and $\overline{R}\left[R_x, R_y, R_z\right]$ the foot of the normal from the observer eye to the plane.

Appendix F contains the detailed mathematical derivations of the above equations.

There are other forms of projections which can be used instead of perspective. True perspective drawings are useful in increasing understanding, but for work where measurements are required, isometric or orthogonal projections (see section 6.2.3), which enable easy measurement, might be preferred.

(b) Brightness modulation

With this technique parts of the picture near to the observer are bright while those away are dim.

When this is required on a view an extra routine is entered just before the vector generating routine and this deposits any brightness change into the display file being built. At each picture the maximum and minimum values of the z coordinate of the points on the screen are noted. The last measured z range is split into n regions, where n are the visible brightness levels available in the machine. Then, when a line is to be drawn from a different region to the previous line, the brightness of the display is set to that of the present region. In this way all brightness levels of a machine are used in any modulated view.

In the present system 8 intensity levels are permitted, stored in the parameter INT for each record in the display file. The value of INT in a particular record corresponds to the last drawn line, and is calculated from the z value of the midpoint in the eye coordinate system (see section 6.7).

Brightness modulation is easy to implement and very effective, offering a much cheaper alternative to hidden-line removal.

(c) Hidden-line (surface) removal

Visibility tests are a desirable feature in picture processing, particularly in complex drawings where the large number of lines eventually render the picture impossible to perceive. However, a considerable amount of computing time is required. Computation goes up roughly as the square of the number of edges. Therefore for moderately complex situations computation can become prohibitive on a small machine.

It is clearly very difficult to establish reliable algorithms to identify the lines to be removed. This problem becomes magnified when they are to be removed from an animated sequence. In general, the geometric calculations are quite straightforward if the objects are convex polyedra, but if the three dimensional bodies are not rectilinear the problem can be very difficult.

A number of ingeneous algorithms have already been devised (28). Of these, the most suitable one, especially for display on a raster scan device such as a television monitor, is perhaps Watkin's algorithm

(29), based on the scan-line technique, in which visibility is tested on each plane parallel to the scan line. Data in raster format is also suitable for producing shaded surfaces and since the tests required for shading are essentially the same as those for visibility, the two processes can be combined with very little penalty on cost.

(d) Shading

This is a technique which has been developed extensively in work related to hidden-line or hidden-surface (more applicable here) removal, mainly at the University of Utah (27, 30) and the CAD centre at Cambridge (31). The technique is based on the recognition of distance and shape as a function of illumination.

The technique has great similarity with finite elements. The surface of the solid is divided into small patches and in regions of greater curvature smaller patches are used and vice-versa. Each individual element is then tested for visibility and the degree of shading required. The visibility test is the same as that required in hidden-surface removal. Hidden-line removal is an essential pre-requisite for any shading algorithm. The amount of shading is determined by calculating the angle between the normal to the plane of the element and the vector direction of the propagation of the light. The normal vector can be calculated from the cross product of two vectors on the plane or from the equation of the plane. The angle of incidence is given by the dot product of the normal and the line of incidence. Brightness (and visibility) increases as the angle increases from 0 to $\pi/2$. For visibility test only the sign of the dot product is required to determine whether the plane is facing the light or not. This is a preliminary test to eliminate all planes facing in the wrong direction. Gouraud (30) used a method where the intensity at the point where the elements meet is calculated. The intensity is then interpolated to provide smooth shading of the surface. For illumination purposes either a point or parallel beam source of light can be used. The surface can also be given a reflective index to make it shiny, dull or even transparent.

The output is obtained on devices that can scan at different intensity levels, either by drawing a series of parallel lines or by overwriting, i.e. multiple exposure of individual dots. The denser the region the greater the number of exposures.

Results obtained by some researchers (27, 30, 31) were excellent, with a very photo-realistic appearance, but had to rely on powerful, special purpose hardware.

Fig 6.31 shows a number of frames from a film of a flying aircraft with partial hidden-line removal and uniform shading produced on a microfilm plotter.

(e) Movement

Movement is the basis of the whole concept of animation. In this case it is used as a means of improving recognition of the displayed object. As the object is rotated or translated, ambiguities that arise due to the superposition of points are eliminated and the geometrical properties of the object are revealed by the interaction of the points defining the object. The effect is particularly effective if perspective is superimposed on the display.

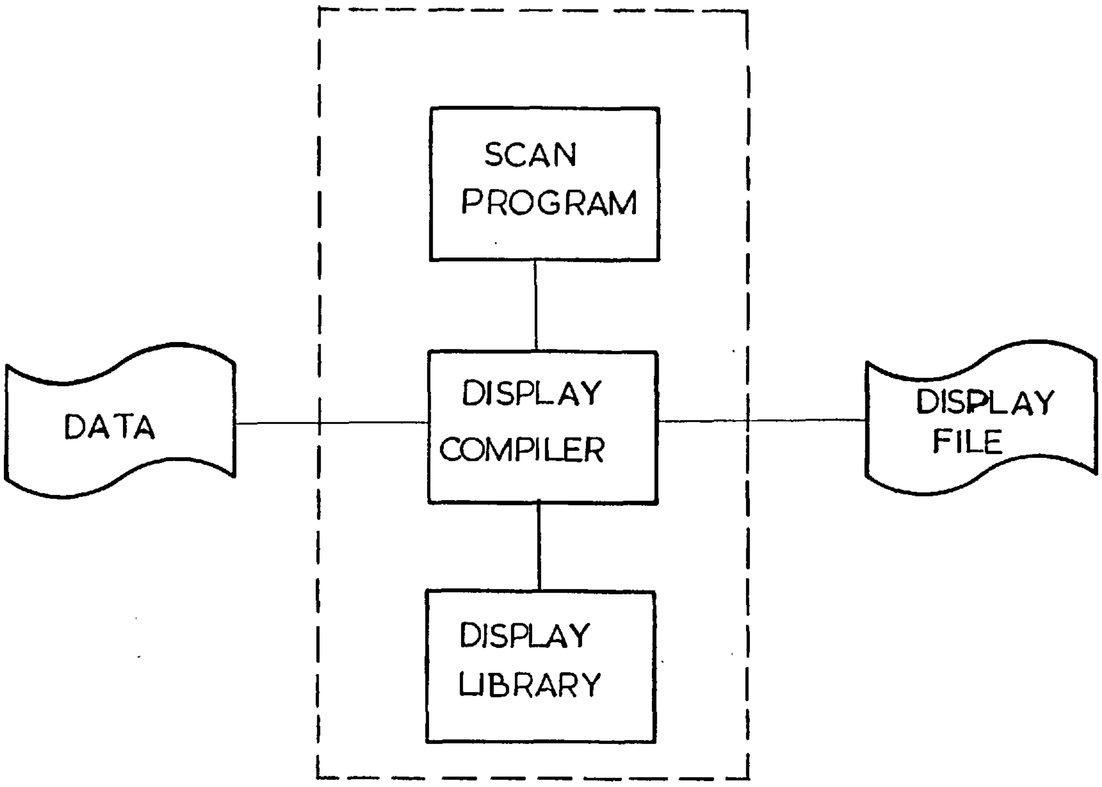
If large amounts of movement are not desirable, the object can be rocked on the display. This does not entail complicated movements and can be quite effective on a refreshed type of display.

Movements can be obtained by producing a line test filmed directly off the screen. For output recorded on film the movement is generated by the playback so no additional processing is required.

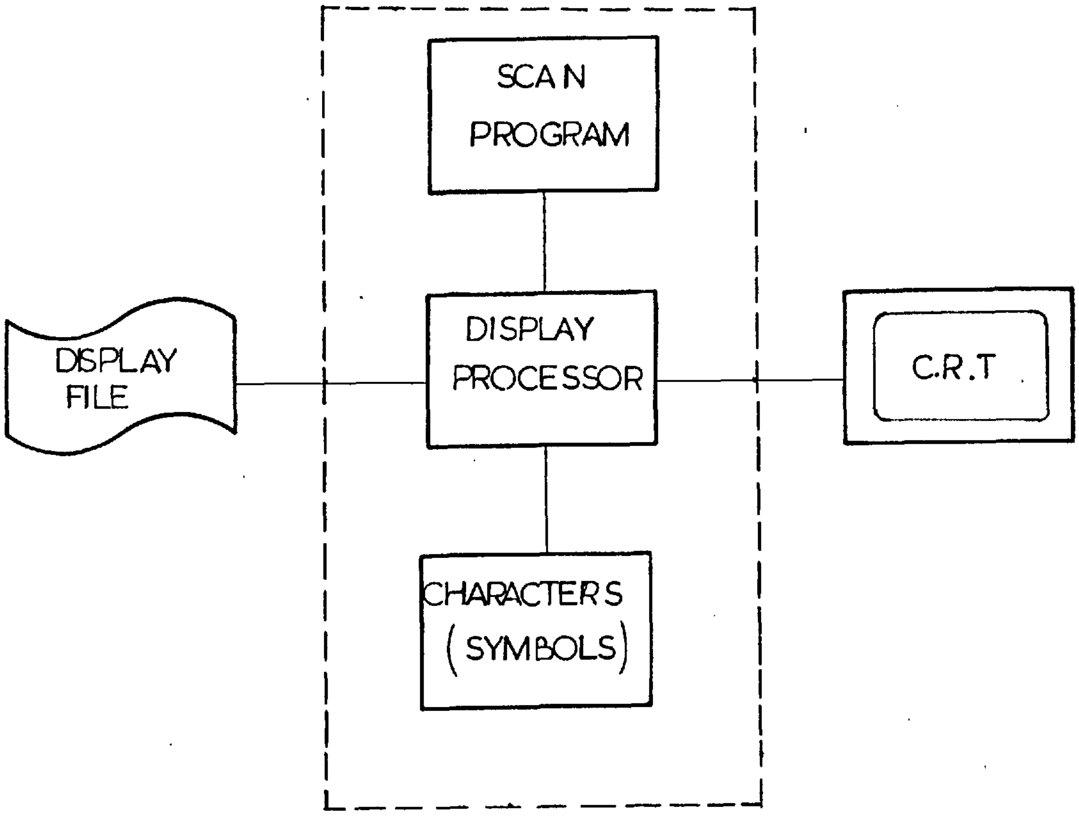
To sum up, the main drawback of these techniques seems to be in the requirement for special display hardware and massive computing power. Also, for realistic display times some of the mathematical transformation should be implemented in hardware.

The CADMAC-11 configuration is rather limited in this sense and only perspective can be implemented effectively at the present stage. Brightness modulation and shading are possible on microfilm (see Fig 6.31) but the process is time consuming and difficult to control. Hidden-line removal is a classical problem in picture processing. Many algorithms have been suggested but few are actually practical enough (28). Simpler for rectilinear convex bodies with flat surfaces, it becomes increasingly complicated with curved surfaces. Within interactive systems it is possible to devise algorithms which require direct intervention from the user. Many trivial decisions, such as entirely accepting or rejecting an item, are best made by the user based on the displayed information. The system would require very intricate sorting algorithms to arrive at the same decision.

The storage format of the transformed data in FILE 2 allows for the future implementation of visibility tests. The Z location in the record is reserved for storing the z coordinate of a point transformed to an eye coordinate system (see next section). The data is then in a form suitable for either direct perspective viewing or post-processing to remove the hidden line. This perhaps on a larger machine, driving a suitable type of display. The blockfilling facility described in section 5.4.8 provides a software technique for converting data from random x-y into a suitable raster form.



(a) compiler



(b) processor

Fig 6.20 3D display

6.7 Eye Coordinate System

In the last section a method whereby a perspective view of a three dimensional object could be generated was described. The final drawing was made up of $\left[X, Y \right]$ points on the projection plane. However, it is not always desirable to represent an object as a flat drawing. Sometimes it is essential to preserve the depth information in order to determine the spacial properties of the solid, which would have been lost with a plane representation.

What is needed is a transformation which converts a three dimensional object as viewed in perspective into another object which will give the same view when projected orthogonally (Fig 6.21). The transformation in effect moves a local observer to infinity and distorts the object appropriately so that it still looks the same on the viewing screen. The transformation preserves all the spatial qualities of the object so it is always possible to apply the three dimensional perspective transformation before doing any visualisation computation like brightness modulation and hidden-line removal. It is much easier to perform hidden-line removal computation from an orthogonal projection than a perspective projection.

Before the distortion transformation is applied it is necessary to define the object in terms of a local system of reference axes with a different orientation.

The new set of axes x', y', z' is taken with the origin at the eye position and with positive z' along the line of vision in the opposite direction (Fig 6.22). x' is in the plane parallel to the x-z plane and y' is upwards. The angles the line of sight makes with the x, y, z axes are given by their cosines cos α, cos β, cos γ respectively.

The transformation is a combination of a translation and a rotation.

$$X' = R(X - T)$$

$X = [x, y, z]$ point defined in the old coordinate system.

$X' = [x', y', z']$ point defined in the eye coordinate system.

T = Translation vector given by $[E_x, E_y, E_z.]$

Matrix R is given by

$$\begin{bmatrix} \lambda_{x'x}, & \lambda_{x'y}, & \lambda_{x'z} \\ \lambda_{y'x}, & \lambda_{y'y}, & \lambda_{y'z} \\ \lambda_{z'x}, & \lambda_{z'y}, & \lambda_{z'z} \end{bmatrix}$$

where $\lambda_{x'x}$ is the direction cosine between x' and x and so on.

From above

$$\lambda_{z'x} = \cos\alpha$$

$$\lambda_{z'y} = \cos\beta$$

$$\lambda_{z'z} = \cos\gamma$$

And x' is parallel to X

y' is parallel to Y

$[X, Y]$ are the axes on the projection plane.

Referring to equation (22) in Appendix F

$$\lambda_{x'x} = \cos\gamma/\sin\beta$$

$$\lambda_{x'y} = 0$$

$$\lambda_{x'z} = -\cos\alpha/\sin\beta$$

and from equation (23)

$$\lambda_{y'x} = -\cos\alpha\ \cos\beta/\sin\beta$$

$$\lambda_{y'y} = \sin\beta$$

$$\lambda_{y'z} = -\cos\beta\ \cos\gamma\,/\sin\beta$$

thus $R = \begin{bmatrix} \dfrac{\cos\gamma}{\sin\beta} & 0 & \dfrac{-\cos\alpha}{\sin\beta} \\ \dfrac{-\cos\alpha\ \cos\beta}{\sin\beta} & \sin\beta & \dfrac{-\cos\beta\ \cos\gamma}{\sin\beta} \\ \cos\alpha & \cos\beta & \cos\gamma \end{bmatrix}$

Alternatively, R can be derived by regarding the transformation as a combination of two rotations (Fig 6.23). The first R_θ about the y axis in the x-z plane through angle θ anticlockwise (object rotation).

Thus $R_\theta = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}$

from Fig 6.23

$$R_\theta = \begin{bmatrix} \dfrac{\cos\gamma}{\sin\beta} & 0 & \dfrac{-\cos\alpha}{\sin\beta} \\ 0 & 1 & 0 \\ \dfrac{\cos\alpha}{\sin\beta} & 0 & \dfrac{\cos\gamma}{\sin\beta} \end{bmatrix}$$

where $\sin\beta = (\cos^2\alpha + \cos^2\gamma)^{\frac{1}{2}}$

And $R_\phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \sin\beta & -\cos\beta \\ 0 & \cos\beta & \sin\beta \end{bmatrix}$$

therefore $R = R_{\phi} R_{\theta}$

as before

$$R = \begin{bmatrix} \frac{\cos\gamma}{\sin\beta} & 0 & \frac{-\cos\alpha}{\sin\beta} \\ \frac{-\cos\alpha\cos\beta}{\sin\beta} & \sin\beta & \frac{-\cos\beta\cos\gamma}{\sin\beta} \\ \cos\alpha & \cos\beta & \cos\gamma \end{bmatrix}$$

The above rotation matrix is valid for all directions of z' in Fig 6.23 except when it is vertical. In either of the cases shown in Fig 6.24 the direction cosines of the axis z' with respect to the reference axes can be determined by inspection and the rotation matrix becomes

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta & 0 \\ 0 & 0 & -\cos\beta \end{bmatrix}$$

The matrix is valid for both cases in Fig 6.24. The direction cosine cos β is +1 for the first case and -1 for the second.

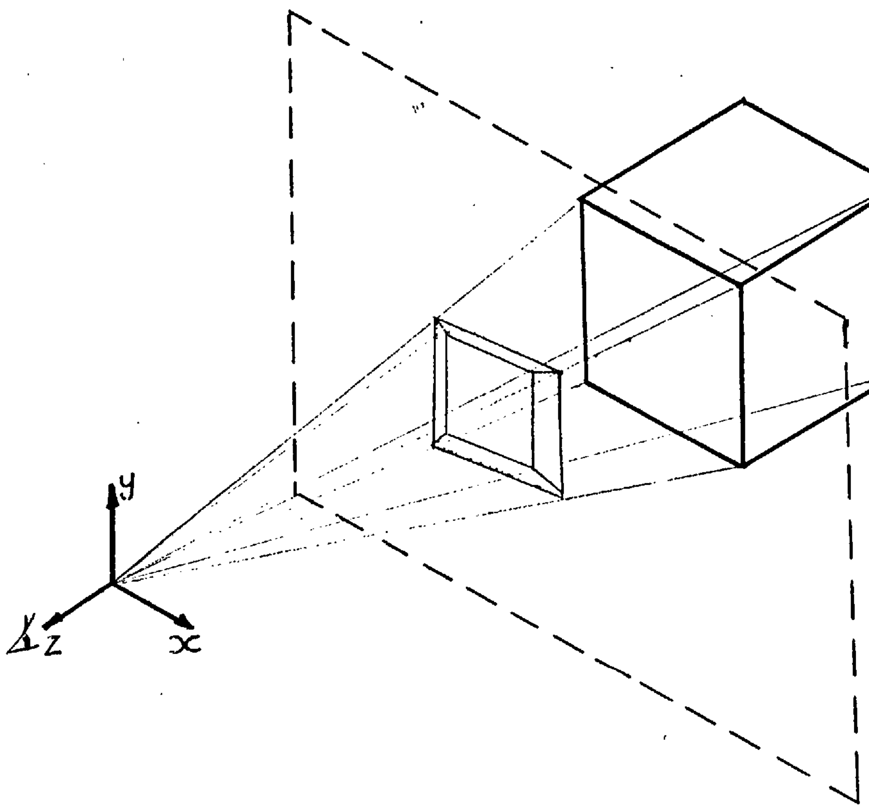
In the new coordinate system the perspective projection becomes a simple linear relationship because the x'-y' plane is parallel to the projection plane.

From Fig 6.25

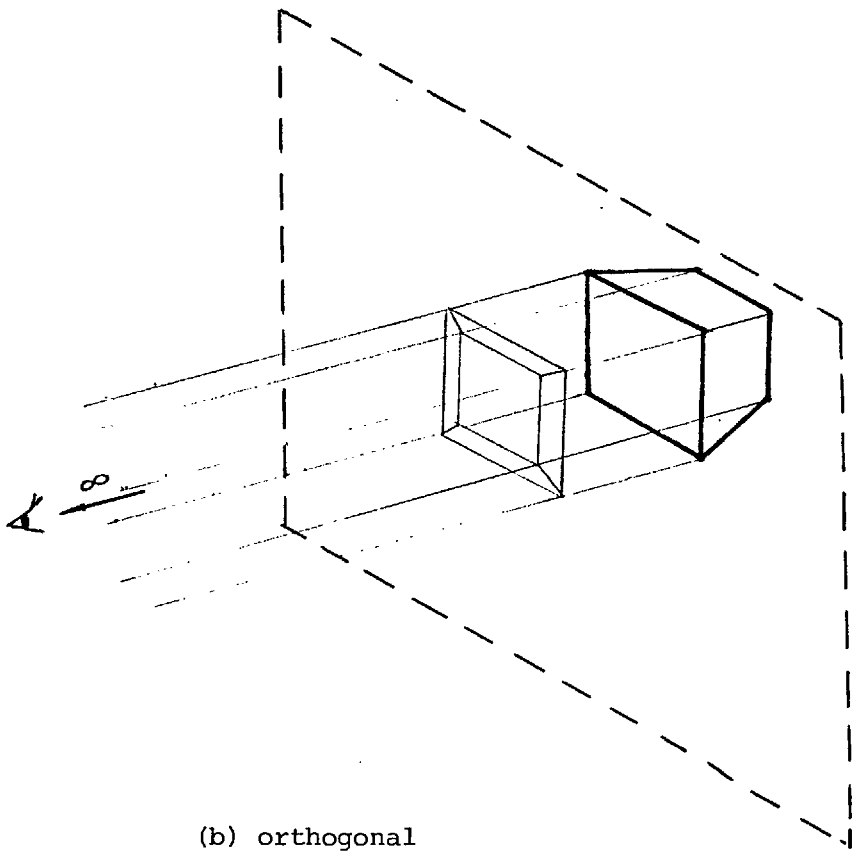
$$\begin{bmatrix} X & Y & Z \end{bmatrix} = \begin{bmatrix} x' & y' & z' \end{bmatrix} \begin{bmatrix} D/z' & 0 & 0 \\ 0 & D/z' & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where D is the distance from the eye position to the projection plane.

The Z term is the depth coordinate. This value is stored in the Z location of display FILE 2 described in section 6.2.1. The display features of FILE 2 remain unchanged and the same program is used to generate the perspective view, but in this case it will consist of the orthogonal projection of the transformed solid. This Z coordinate can be used in visualisation tests together with the parameters INT and LEV stored in the I location.



(a) perspective



(b) orthogonal

Fig 6.21 Perspective - orthogonal projections of a cube

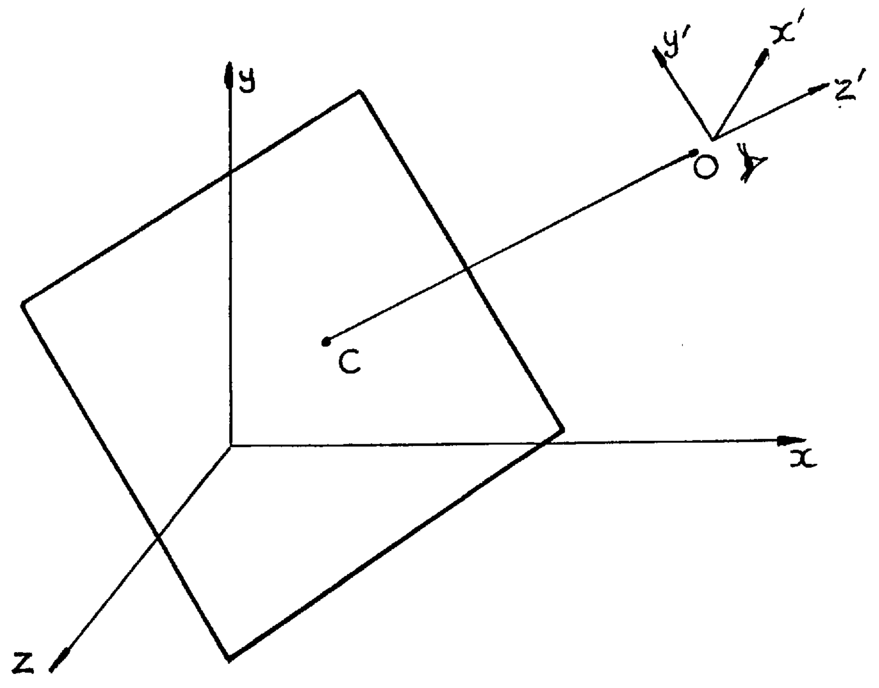


Fig 6.22 Eye coordinate system

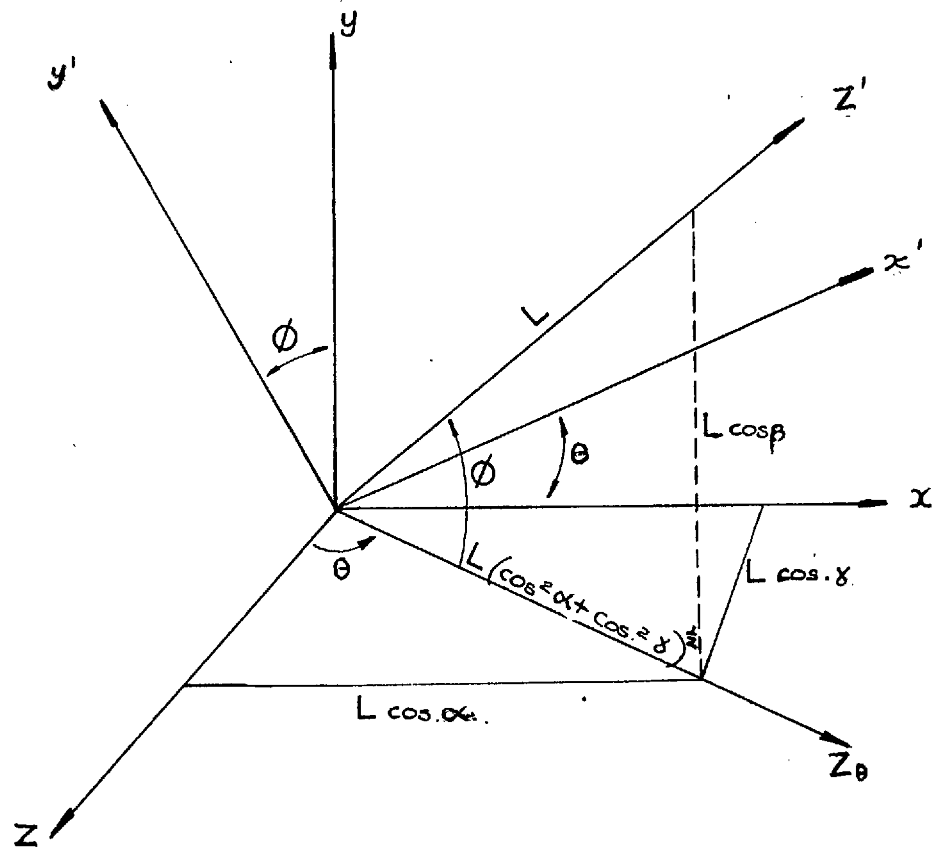


Fig 6.23 Rotation of axes in space

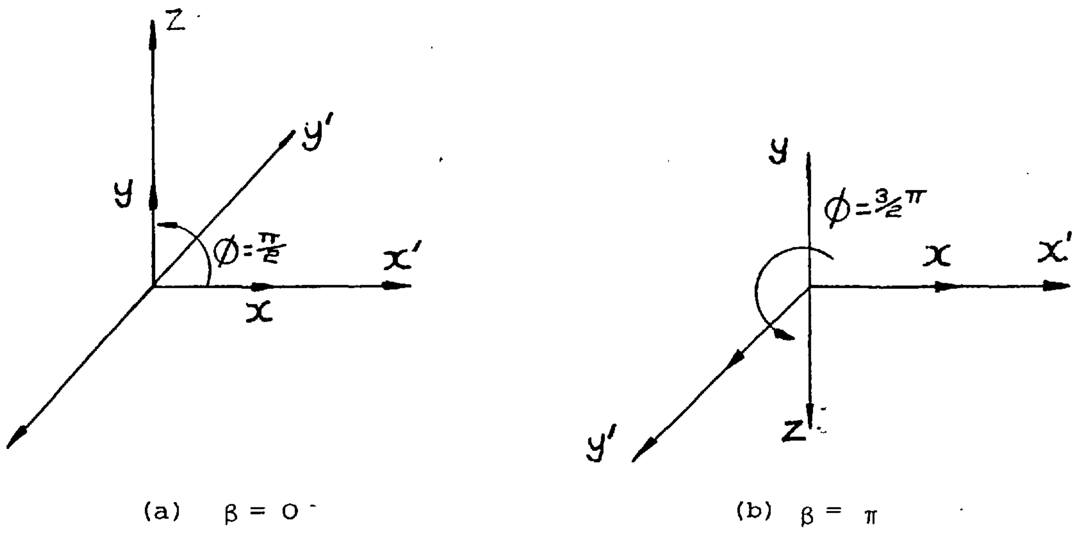


Fig 6.24 Orientation of axes for a vertical line of sight

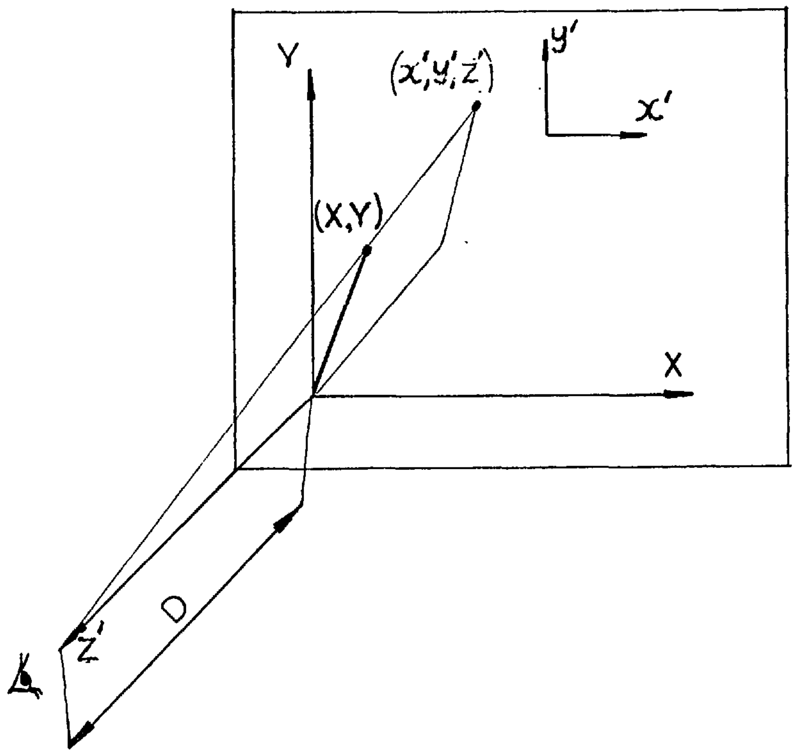


Fig 6.25 Linear perspective

6.8 Joy-Stick Function

The variations in perspective parameters, as described earlier, allow a very convenient solution to a joy-stick controlled display.

The joy-stick function allows the operation of a joy-stick to be emulated on a patch on the menu. By pressing the pen button on the menu the user can control two functions: rotation and zoom. Rotations are possible about two axes: y axis and an axis on the x-z plane determined by the pan and tilt angle respectively. (θ and ϕ in Fig 6.26).

The eye position is defined by $[\, E, \theta, \phi \,]$ where

E - distance from centre of attention

θ - pan angle

ϕ - tilt angle.

The coordinates of the eye $[\, E_x, E_y, E_z \,]$ with respect to the centre of attention are given by

$$E_x = E \cos\phi \sin\theta$$

$$E_y = E \sin\phi$$

$$E_z = E \cos\phi \cos\theta$$

and the direction cosines

$$\cos\alpha = E_x/E$$

$$\cos\beta = E_y/E$$

$$\cos\gamma = E_z/E$$

Conversely

$$E = (E_x^2 + E_y^2 + E_z^2)^{\frac{1}{2}}$$

$$\theta = \tan^{-1}(E_x/E_z)$$

$$\phi = \sin^{-1}(E_y/E)$$

The observer is assumed to be always facing the origin of the coordinate system. For cases where the object is positioned such that it falls out of range, it can be repositioned by using the centering function, which automatically places the object with the centre of volume at the origin.

It is easier to create the movement of-the object by varying the observer viewpoint than the actual object position. With the centre of attention inside the object, the object is always visible on the screen. The coordinates of the object in the data file remain unchanged, only the perspective parameters are updated. Since the perspective transformation is always requested before display and the time factor involved in the updating of parameters is relatively insignificant, the resultant increase in display time is minimal.

Rotation is obtained by varying the two angles θ and ϕ . The relationships between θ and ϕ and the direction cosines cos α, cos β and cos γ are given by (see Fig 6.23)

$$\theta = \tan^{-1} \left(\frac{\cos \alpha}{\cos \gamma}\right)$$

$$\phi = \tan^{-1} \left(\frac{\cos \beta}{\sqrt{\cos^2\alpha + \cos^2\gamma}}\right)$$

$$\cos \alpha = \cos \phi \cos \theta$$

$$\cos \beta = \sin \phi$$

$$\cos \gamma = \cos \phi \cos \theta$$

The zoom factor in the display is determined from the ratio between the distance of-the plane of projection and the observer distance (D/E).

It will be shown later that magnification is proportional to the above ratio, i.e.

$$\frac{H'_2}{H'_1} = \frac{R_2}{R_1}$$

where H'_i is the image size for ratio $R_i = \frac{D_i}{E_i}$ and $D_i = E_i - P_i$, P_i being the distance of the projection plane from the object. Any of the distances E, P and D may be varied independently, but the effects achieved are different.

Another ratio needs to be defined, the distortion, given by $\frac{H'}{H''}$ which is the ratio of the image sizes for two objects of the same size at a fixed distance apart.

(a) Lens zoom (constant P)

By fixing the plane position and varying the eye position we obtain a 'lens zoom' effect. The distortion varies with the magnification in the same way as the image obtained through a zoom lens, when changing the focal length of the lens, with the camera in a fixed position relative to the object.

Take Fig 6.27 a. From similar triangles

$$\frac{H}{E_1} = \frac{H'_1}{D_1}$$

$$\frac{H}{E_2} = \frac{H'_2}{D_2}$$

Magnification is given by

$$\frac{H'_2}{H'_1} = \frac{D_2}{D_1} \cdot \frac{E_1}{E_2}$$

therefore $\text{MAG} = \frac{R_2}{R_1} = \frac{D_2}{E_2} \cdot \frac{E_1}{D_1}$

From Fig 6.27 b

$$\frac{H}{E_1} = \frac{H'_1}{D}$$

$$\frac{H}{E_1+\ell} = \frac{H''_1}{D_1}$$

where H' and H" are the image sizes of two segments of same length H, distance ℓ apart. E_1 is the distance from the first object to the eye positions. The same relations apply for E_2, therefore distortion is given by

$$\frac{H'}{H''} = \frac{E+\ell}{E}$$

For a fixed P_1 given E_1 and D_1

$$P = E_1 - D_1$$

$$R_1 = D_1/E_1$$

to obtain a magnification $MAG = \frac{R_2}{R_1} = \frac{D_2}{D_1} \cdot \frac{E_1}{E_2}$

The new eye position is given by

$$E_2 = \frac{P}{1 - R_1.MAG}$$

and $D_2 = E_2 - P$

(b) Object zoom (constant D)

The second alternative is to move the eye and the plane together by keeping D constant. The size change is given by (Fig 6.28 a)

$$\frac{H'_2}{H'_1} = \frac{E_1}{E_2}.$$

$$\text{size} \; \alpha \; \frac{1}{E}$$

again MAG $= \frac{R_2}{R_1} = \frac{D}{E_2} \cdot \frac{E_1}{D} = \frac{E_1}{E_2}$

Therefore for a certain magnification MAG

$E_2 = E_1/\text{MAG}$

$D_2 = D_1$

The distortion given by $\frac{H_1'}{H_1''} = \frac{E_1 + \ell}{E_1}$ actually increases when the magnification increases. The zoom has the same effect as closing in with a lens of fixed focal length on a fixed object, or conversely move the object towards the observer. This is called an 'object zoom'.

(c) <u>Combined zoom</u> (constant E)

If the eye position is now fixed and the projection plane is moved we have a 'combined zoom'. This is equivalent to looking through a zoom lens and closing in as it is zoomed out. The distortion remains the same as the size increases.

$$\frac{H'}{H''} = \frac{E + \ell}{E}$$

is a constant now, because the eye position is fixed.

From Fig 6.29 a

MAG $= \frac{H_2'}{H_1'} = \frac{D_2}{D_1}$ since E is constant.

size α D

again magnification MAG $= \frac{R_2}{R_1} = \frac{D_2}{D_1} \cdot \frac{E_1}{E_2} = \frac{D_2}{D_1}$

For a combined zoom:

$D_2 = D_1 \cdot \text{MAG}$

$E_2 = E_1$

The effects of the three types of zoom on a cube are depicted in Figs 6.27 c, 6.28 c, 6.29 c. The three can be combined to give almost any type of zoom effect required.

For any eye-plane combination

$$\text{Magnification} \propto R \propto \frac{E - P}{E} \propto \frac{D}{E}$$

When performing a zoom effect it is essential to ensure that the projection plane P lies between the eye position and the object centre, i.e. D must always be positive or unpredictable results, such as inversion, partial inversion or very large scale magnification, will occur.

The control of the parameters θ, ϕ and R is via a patch on the menu (Fig 6.30). It is the same as the patch used for drive mode (section 5.4.3) consisting of a square 9 x 9cm in size divided into 81 smaller squares, and a rectangle 6 x 9cm divided into 9 rows. The square controls the tilt and pan angles and the rectangle the zoom magnification. The angles are in increments of 1, 5, 15 and 45 degrees, or combinations of these. Rotations can be accumulated before display, which is activated by the 'enter'switch. The zoom zone is divided into three parts, each controlling a different type of zoom. The upper rows are for zoom-ins and the bottom ones for zoom-outs.

A point within the small squares must be digitised to obtain the desired orientation and magnification. The values of these are displayed on the coordinate display unit.

The joy-stick control facility enables the user to select quickly the best orientation of an object for inspection. Because the display is linked point by point to the object data, it provides a convenient means of isolating data for construction and editing purposes.

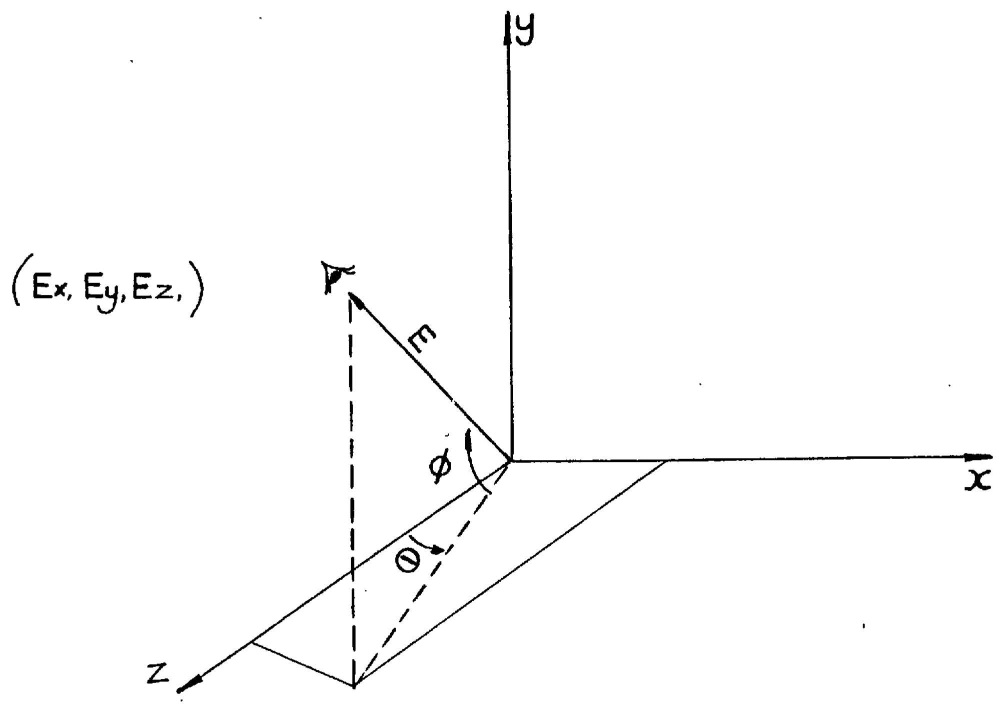
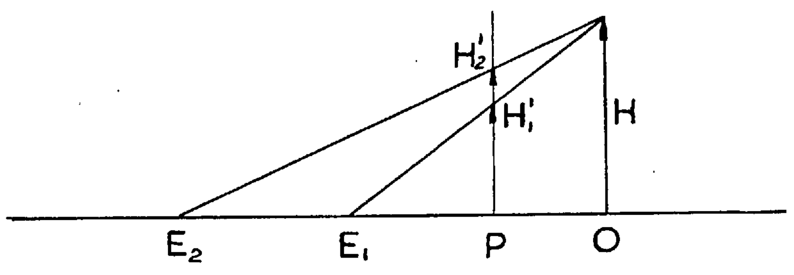
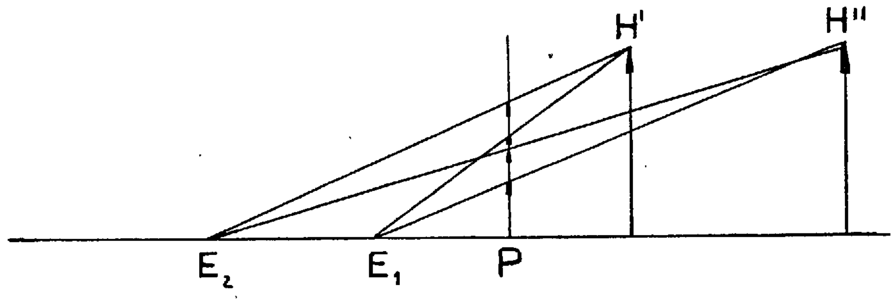


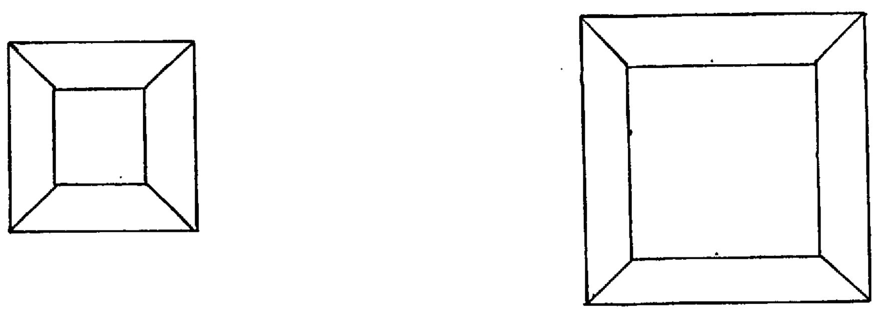
Fig 2.26 Eye position in terms of pan and tilt angles.



(a) constant P

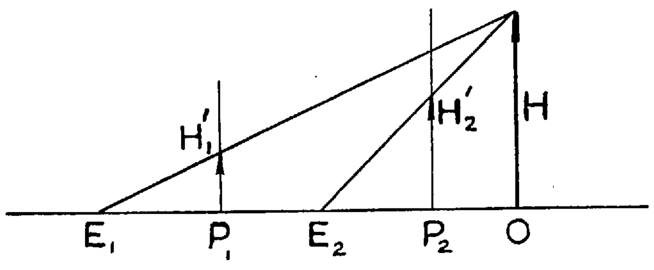


(b) distortion

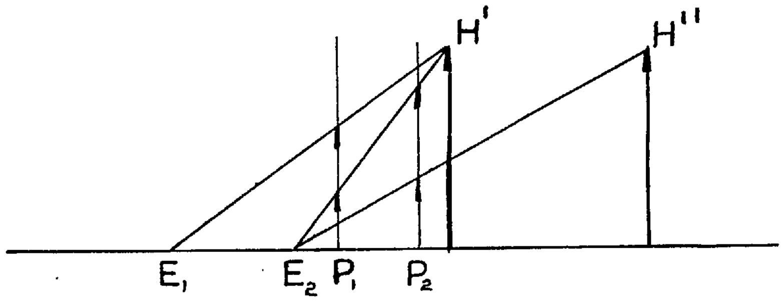


(c) less distortion

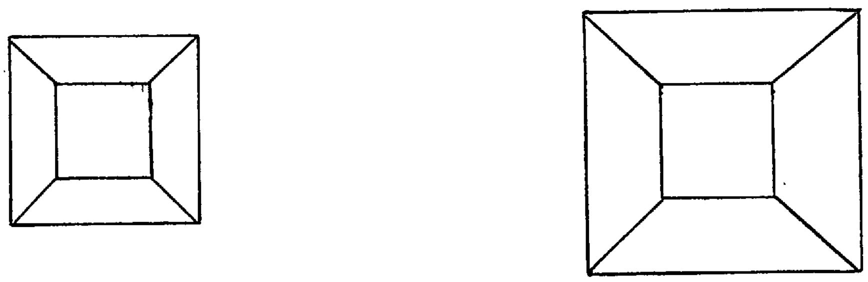
Fig 6.27 Lens zoom



(a) constant D

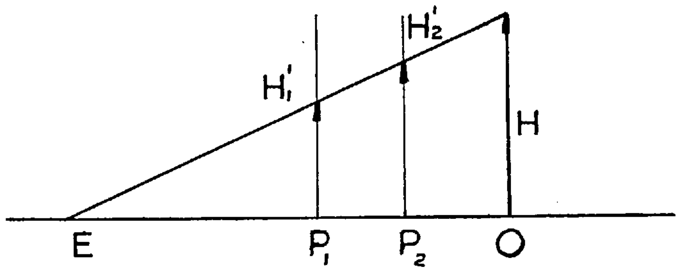


(b) distortion

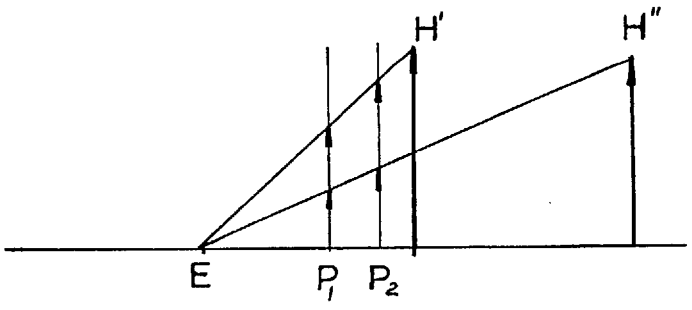


(c) more distortion

Fig 6.28 Object zoom



(a) constant E



(b) distortion

(c) same distortion

Fig 6.29 Combined zoom

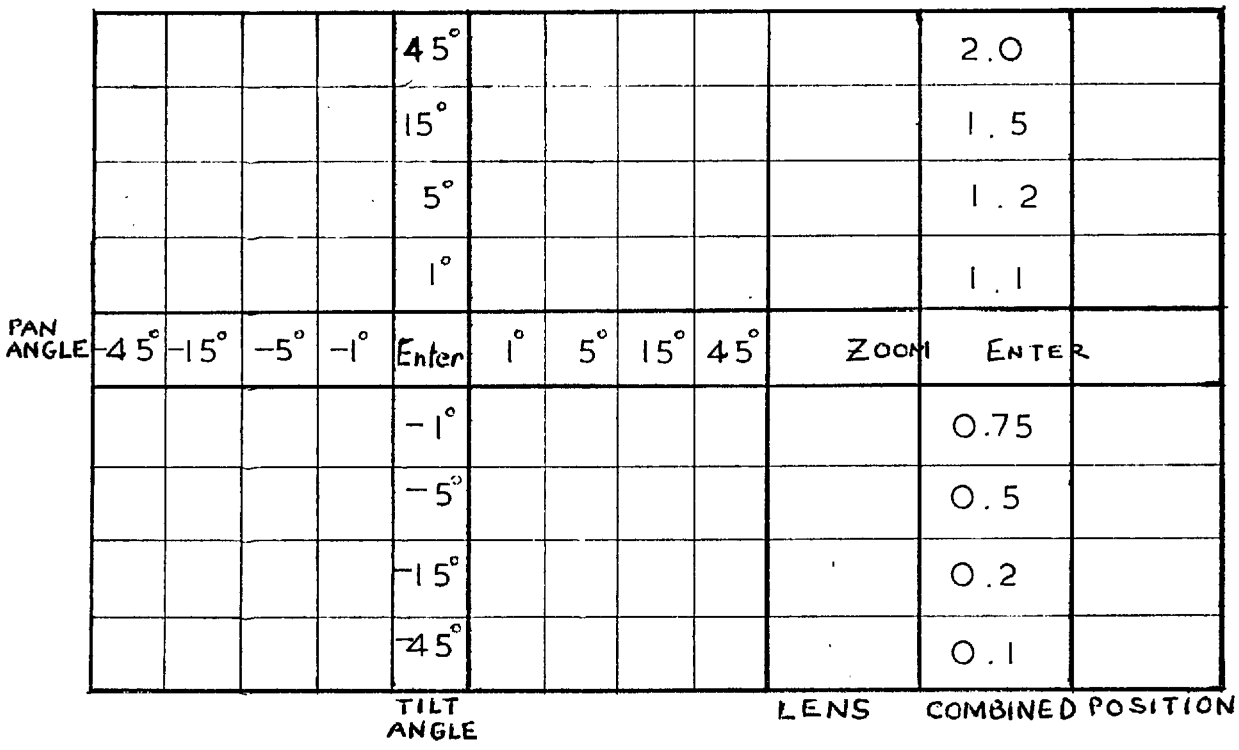


Fig 6.30 Joy-stick patch

6.9 Movement Definition

There are two ways in which the movement of an object on the screen can be controlled: in-betweening and path description.

6.9.1 In-betweening

In-betweening in three dimensions is similar to that for two dimensions with the interpolation being performed on three coordinate values instead of two. Two key positions are required to define a movement. These may be two different objects or the same one in two different positions, with the first being transformed into the second. Either linear or cosine interpolation can be used. To give a smooth action the movement should be faired at both extremes.

6.9.2. Paths

In three dimensions there are two ways in which paths may be defined, resulting in similar effects on the screen. The viewing of a three dimensional object depends on the object-observer geometry only, so either the object or the observer may be moved relative to the other.

The path of the item to be moved, either the object or the eye, are stored in a permanent file in standard data format. The usual way in which this data is generated is by either digitising discrete points which are then curve fitted or by generating the points from a program.

6.9.2.1 Object path

In order to move an object along a given path, a point of reference on the object must be defined. This point is chosen to be the middle of a box which contains every point making-up the object. The object is always positioned so that the reference point coincides with the origin.

The movement can consist of translation only or to include rotation to follow the path. The transformation is identical to that used in the conversion from absolute coordinates into the eye coordinate system (see section 6.7).

For fixed movement only, the translation is performed.

$$X' = X + \overline{P}$$

where $\overline{P} = [P_x, P_y, P_z]$ is the position of a point on the path and represents the new, moving origin.

For rotation the direction cosines in the rotation matrix are given by

$$\cos\alpha = (Q_x - P_x)/L$$

$$\cos\beta = (Q_y - P_y)/L$$

$$\cos\gamma = (Q_z - P_z)/L$$

where $\overline{Q}$ $[Q_x, Q_y, Q_z]$ is a second point on the path and $L = |\overline{PQ}|$, the length of the vector.

Fig 6.31 depicts a number of frames from a film showing the flight of an aircraft in three dimensions. The path is defined by four points in space only, which were curve fitted to give 1500 intermediate flight positions. The eye position remained fixed.

6.9.2.2 Observer path

The movements obtained by varying the eye position are similar to those from the joy-stick facility, except that the eye position is now stored as a series of points defining the observer path.

The centre of attention is taken to be the centre of the object, which coincides with the origin of the absolute coordinate system. This ensures that the object is always visible on the screen.

So for a position $\overline{P}$ $[P_x, P_y, P_z]$ in the path file, the viewing parameters are given, as in section 6.8, by

$$E = (P_x^2 + P_y^2 + P_z^2)^{\frac{1}{2}}$$

$$\cos \alpha = P_x/E$$

$$\cos \beta = P_y/E$$

$$\cos \gamma = P_z/E$$

The operation of these path facilities is identical to that described for two dimensions in section 5. 6.4.

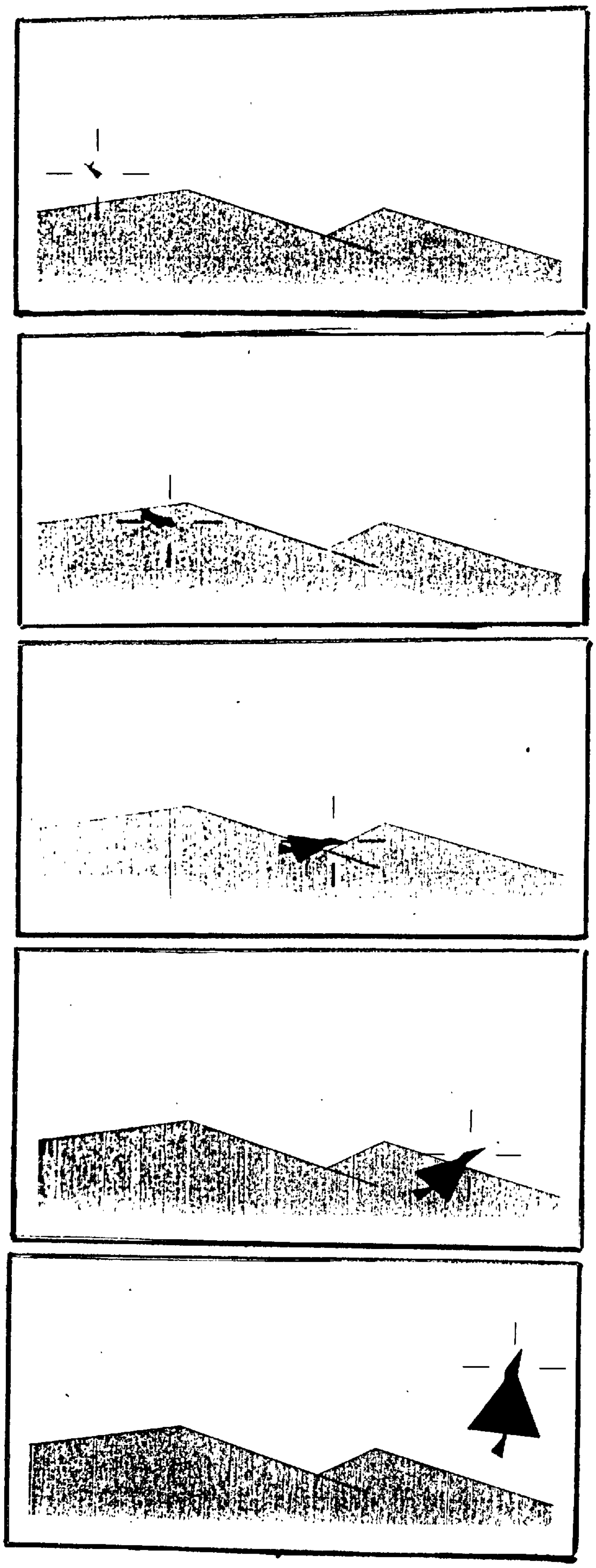


Fig 6.31 Flight sequence

CONCLUSIONS

The work described here demonstrates the use of computer-aided-design techniques in areas of work not normally associated with CAD. It also demonstrates that the normally exclusive areas of arts and science can have grounds in common and that developments ascribed to one can be successfully applied to the other.

The systems described in the thesis provide two and three dimensional interactive graphics facilities which can be used in the study of certain dynamic phenomena in science and technology. The system can be used as a design tool, an animating machine and as a recording device, all combined into one.

The operation of the system is almost self-explanatory and should not be too difficult to master. For every user there should be a learning period, not just to learn how to operate the system, but to discover its potentials and to learn to plan the work in order to exploit these potentials to the fullest.

The existing facilities provide the user with most of his basic requirements, but in order to produce more advanced types of scientific simulation, the system is modular enough to allow the experienced programmer to incorporate his own applications programs. The basic data description is restricted to the geometry of the constructs. The qualifying parameters in the data records may be redefined to contain data relevant to their physical properties. However, it is often preferable to create separate files for the geometric data description and the data for analytical purposes. This provides quicker graphical interaction and also removes the need to process large files, containing redundant data, which would only delay the display interaction. Typically the system operates as a graphics input processor and manipulator to generate data, which can then be processed either on the mini itself or on a larger machine. GAGS-3 is currently being adapted for work in finite elements, pipe layout and routing and n.c. surface machining.

From the animation point of view, although Computer Animation has already been proved technically feasible on an experimental level, it remains to be proved economically and practically viable. This will depend upon a number of major factors:

1 - Ability to resolve certain technical problems, so that the resulting films will be commercially acceptable.

2 - Ability to reduce costs through economical system design and through appropriate management of a wide variety of human and technical skills.

3 - Ability to persuade the established market to use the facilities provided.

4 - Emergence of a new market, willing and able to understand and absorb the new product.

In some cases, many of the old technical problems, often encountered in all animation processes, may be solved in a more elegant manner by the use of computers (e.g. sound and image matching), but on the other hand new problems may be created by generating pictures by computer (e.g. quality, compatibility and colour range).

Commercially the system has had considerable success, as demonstrated by the great interest aroused and the amount of work, of very varied nature, already undertaken by Video Animation Ltd. The system is already being used in full production work by VAL.

The economic advantages offered by the system are difficult to estimate in exact terms of cost, because the product is marketed on the basis of current prices. The true savings are in time, where, compared with conventional hand techniques, they are at least of the order of thirty per cent, but can be much more in some cases. Films, that normally would have taken three or four weeks to produce by hand, have been completed in less than a week with the aid of the computer. Technically the animation is of a better quality, the movements more precise, and including certain sequences that would normally have been regarded as too difficult to be done at all by hand.

The advantages of using mini-computers for this type of application are quite obvious; lower cost, better on-line response and less restricted usage, but there are still some doubts as to the use of a relatively expensive tool, such as a computer, in a, presently, still quite restricted market. The true economic advantages of the system will only be realized in the long run, for it will still be some time before the potentials of computerised films are fully appreciated. But there is already a favourable trend towards its acceptance, an emerging new market, especially in educational and technical films, such as the Open University, where computer films are already playing a very important role.

SUGGESTIONS FOR FURTHER WORK

Before any further development is carried out, overall improvements on the operation of the system are recommended. These improvements should preferably be accompanied by the introduction of some more powerful hardware.

GAGS-2 should be redesigned to incorporate most of the improvements made in the design of GAGS-3: more compact storage, better filing techniques, improved overlay and menu handling, faster display and interactive editing, and overall elimination of redundant processes and data storage.

The performance of both systems can be further improved by incorporating the Batch mode facility described in Appendix I. This will not only speed up the process but also eliminate some of the more time-consuming procedures.

The two systems can be integrated to provide a single, very powerful general, two and three dimensional package. This, however, represents no practical advantage because the two systems can communicate externally, via disc or magnetic tape files. Also keeping the size of the system small ensures greater flexibility in any future system upgrading.

Most future developments will depend on the acquisition of more powerful hardware. There are several areas already under consideration.

Areas which have already been tentatively investigated are: computerization of movement notation, sound-track breakdown, voice analyser for lip-synchronization, digital video output, scan-conversion techniques, optical digitisers, video scan input, n.c. rostrums, computer operated cameras and micro-programmed functions.

In two dimensional work, the advent of the full colour video display seems to point to a future in digital image processing, with data defined in terms of suitable matrix character representation. The conversion from random x-y data can be performed by suitable software scan-conversion techniques, similar to block-filling (Section 5.4.8). To overcome the problem of hard edges a degree of 'fuzziness' can be introduced in the characters with suitable mixed-colour (fuzzy) characters. The effectiveness of such a technique can only be determined experimentally because the effect is mostly subjective. The apparatus should also provide a compatible output for recording directly on to video tape.

In three dimensional work, attention should be turned towards special displays, incorporating hardware transformations. Better means of interaction can be obtained by the use of more natural input and output hardware such as light-pens, joy-sticks, function boxes and better digitising tablets. The manipulation and perspective routines can be hardwired to permit direct interaction from a joy-stick. On the software side a hidden-line and shading algorithm should be implemented on the system. But, because of the nature of the computation involved, the program should be regarded as a post-processor, only invoked on special applications. The final output should perferably be video compatible too.

From the animation point of view, the system has great potential in automation and mechanization.

The information provided by the camera compound program can be used to generate numerical tapes to control a rostrum camera. Rostrums are already highly mechanised devices and it would not be too difficult to replace manual controls with computer controlled motors. The arrangement would not be unsimilar to the electromechanical driving mechanism of a flat bed plotter. Stepping motors would allow very precise positioning of the camera with respect to the artwork. One important advantage of n.c. rostrums is the ability to expose the film while the camera is moving, thus eliminating the unpleasant effect of strobing.

The ultimate in animation is to provide the total system, which has sound, image and mechanical capabilities. The initial soundtrack broken down into the number of frames and sequences, and used to time the movements. Artwork is input directly by sketching from the tablet. Animation is produced by transformation routines. Lip-synchronization is provided from the information originally derived from the soundtrack. Output is produced directly, in full colour, on video, or on cel, with a colour plotter. The program also generates the instructions to drive a numerically controlled rostrum on which the cels are photographed. The task of the animator would be to concentrate mainly on the creative aspects of his work.

ACKNOWLEDGEMENTS

The author would like to thank the following for their help in the present work:

Video Animation Ltd for their financial support.

Dr A Jebb and Dr C B Besant for making the project possible.

Mr A R Diment and Mr P Chandler for help in implementing and testing many of the programs.

And Mr S Hayward for the many ideas.

REFERENCES

1 - Besant, C B et al	'CADMAC-11, a Fully Interactive Computer Aided Design System', Journal of CAD, Vol 4, No 5, 1972.
2 - Neilson, P Roberts, M	'Animation Stand Computer', BKST Journal, Dec 1973.
3 - Eshkol, N Wachmann, A	'Movement Notation', Weidenfeld and Nicolson, London 1956.
4 - Causley, M	'An Introduction to Benesh Movement Notation', Max Parrish, London 1967.
5 - Newman, W M Sproull, R F	'Principles of Interactive Computer Graphics', Mcgraw Hill 1973.
6 - Zajac, E E	'Film Animation by Computer', New Scientist, Vol 29, p 346, 1966.
7 - Zajac, E E	'Computer Made Perspective Movie as a Scientific and Communication Toll', Communications of ACM, Vol 7, p169, 1967.
8 - Chapman, G A Omann, J J	'Vista - Computer Motion Picture for Space Research', AFIPS - Proceedings of Joint Computer Conference, Vol 31, p59, 1967.

9 - Gott, A H et al	'Teaching Heart Functions - One Application of Medical Computer Application', AFIPS - Proceedings of Joint Computer Conference, Vol 34, p637, 1969.
10 - Huggins, W H Entwisle, D R	'Computer Animation for the Academic Community', AFIPS - Proceedings of Joint Computer Conference, Vol 34, p623, 1969
11 - Weiner, D D Anderson, S E	'A Computer Animated Movie Language for Educational Motion Pictures' - AFIPS Proceedings of Joint Computer Conference, Vol 33-2, p1317, 1968.
12 - Theiss, C M	'Computer Graphics Display of Simulated Automobile Dynamics - AFIPS Proceedings of Joint Computer Conference, Vol 34, p289, 1969.
13 - Fetter, W A	'Computer Graphics in Communication', Mcgraw Hill, 1964.
14 -	'Specifications of Standards for Information Transmission by Digitally Coded Signals in the Field-Blanking Interval of 625 Line Television Systems', published jointly by BBC, IBC and BREMA.
15 - Citron, J Whitney, J H	'CAMP - Computer Assisted Move Production', AFIPS Proceedings of Joint Computer Conference, Vol 33, Part 2, p1299, 1968.

16 - Knowlton, K C	'A Computer Technique for Producing Animated Movies', AFIPS Proceedings of Joint Computer Conference, Vol 25, p67, 1964.
17 - Hopgood, F R A Ralphs, D	'Camp and Camper on Atlas', report 1971
18 -	'CAMAC - Modular Instrumentation Systems for Data Handling', Euraton EUR 4100 e report, March 1969.
19 - Zurner, J A Ritchie, G J	'Linear Current Division in Resistive Areas', University of Essex 1973.
20 - Grindley, R E	'Development and Application of a Low Cost CAD System in Mechanical Engineering', PhD/DIC thesis, Nuclear Power Section, Imperial College, July 1973.
21 - Hamlyn, A D	'The Application of CAD Techniques to Building Engineering Design', PhD/DIC thesis, Nuclear Power Section, Imperial College, July 1974.
22 - McClintock, P H	'The Applications of CAD in Structural Engineering Analysis', PhD/DIC thesis, Nuclear Power Section, Imperial College, Nov 1974.

23 - Nutbourne, A W	'Curve Fitting by a Sequence of Cubic Polynomials.' Part 1 and 2, Computer Aided Design, Winter 1969.
24 - Roberts, L G	'Homogeneous Matrix Representation and Manipulation of N-Dimensional Constructs', Computer Display Review, 1969.
25 - Smith, W G	'Photogrametry as an Aid to Manufacture of Ship Piping', BSRA Report NS 306, 1971.
26 - Yi, C Besant, C B Jebb, A	'Three Dimensional Digitising Techniques for Computer-Aided-Animation', Proceedings of an International Conference on Computers in Engineering and Building Design, CAD 76, March 1976.
27 - Parke, F I	'Measuring Three Dimensional Surfaces with a Two Dimensional Data Tablet', Computers and Graphics, No 1, Vol 1, 1975.
28 - Sutherland, I E Sproull, R F Schumacker, R A	'A Characterization of Ten Hidden-Surface Algorithms', Computer Surveys, Vol 6, No 1974.
29 - Watkins, G S	'A Real Time Visible Surface Algorithm', PhD thesis 1970, Dept of Electrical Engineering, University of Utah.

30 - Gourand, H	'Computer Display of Curved Surfaces', PhD Thesis, 1971, Dept of Electrical Engineering, University of Utah.
31 - Newell, R G	'The Visualization of Three Dimensional Shapes', Proceedings of the Conference of Curved Surfaces in Engineering, Cambridge, March 1972.
32 -	Dos Monitor Programmer's Handbook, DEC Document DEC-11-OM ONA-A-D, Oct 1972.
33 - Madsen, R P	'Animated Film Concepts, Methods, Uses', Interland Publishing Inc. N.Y. 1969.
34 - Yi, C Besant, C B Jebb, A Diment, A R Hayward, S	'Computer Animation', Proceedings of an International Conference on Computers in Engineering, CAD '74; Imperial College, London, 25-27 Sept 1974.

APPENDIX A

SPECIAL FUNCTIONS

This appendix describes the use of the parameters requested by the special effect programs in the system. Typical values are in parentheses.

A.1

Functions	Parameters
SPIN	Centre of spin - digitise point on table - defines vertical axis about which spin takes place. Spin is clockwise.
	Frames per revolution - input by BUTNUM.
	No. of revolutions - input by BUTNUM. total number of frames = frames/rev.* no. of revs.
	Zoom factor (0.2) - input by BUTNUM.
FLIP	Axis of revolution - digitise 2 points - defines the axis in the plane of the picture.
	Frames per revolution - input by BUTNUM.
	No. of revolutions - input by BUTNUM.
	Perspective factor (4000) - input by BUTNUM - required to give three-dimensional effect.
SQUASH/STRETCH	Scaling axis - digitise 2 points axis about which scaling takes place.
	Scale factor (1.5) - input by BUTNUM.
	No. of frames - input by BUTNUM.

SKEW	Skew axis - digitise 2 points - defines plane of zero shear.
	No. of frames - input by BUTNUM.
TUBE	Cylinder diameter - digitise 2 points - axis of revolution normal to plane containing the diameter.
	Perspective factor (10000) - input by BUTNUM - required to give three-dimensional effect.
	No. of frames - input by BUTNUM.
SPHERE	Sphere Diameter - digitise 2 points - axis of revolution normal to plane containing the diameter.
	No. of frames - input by BUTNUM.
SINE	Wavelength - digitise 2 points - gives wavelength of ripple surface.
	B1 - plane corrugated - corrugation on picture plane.
	B2 - line corrugated - corrugation normal to picture plane. Select option with button 1 or 2.
	Frames/wavelength - input by BUTNUM - no. of frames until fully corrugated.
DISC	Centre point - digitise a point - centre of circle.
	Radius - digitise a point - the second point defines the circle with centre at first point.
TWIST	Twist axis - digitise 2 points - similar to FLIP.
	No. of frames - input by BUTNUM.

TORQUE	B1 - Increment angle w/original picture - original picture is twisted with new angle value. Intermediate frames not stored. B2 - Increment picture w/original angle - same angle of twist used on each subsequent frame. Intermediate frames stored. Select option with pen button 1 or 2. Centre of twist - digitise one point. Angle of twist at centre - input by BUTNUM. No. of frames - input by BUTNUM.
ELASTIC	B1 - Increment stretch w/original picture. B2 - Increment picture w/original stretch same as above except scale factor instead of angle. Point to move - digitise a point. Place point - digitise a point, move first point to position of second. No. of frames - input by BUTNUM.
ERODE	Select file - digitise one of the permanent file menu squares. Mince step size - input by BUTNUM - size of each segment to be reduced. No. of frames - input by BUTNUM.
EXPLODE	Select first file. Select final file (optional) Select background file (optional) select by digitising corresponding permanent file menu squares. Last two are optional.

No. of explosion frames - input by BUTNUM -
  outwards frames.

No. of return frames - input by BUTNUM -
  inwards frames.

Type of return: B1 (X+Y per frame)
  B2 (X only then Y only)
  B3 (Y only then X only)
  select with button 1, 2 or 3 for return movement along diagonal or at right angles.

Movement factor (100) - input by BUTNUM
  explosive strength.

Angle of line rotation - input by BUTNUM
  each line rotates by this amount.

Angle of picture rotation - input by BUTNUM
  whole picture rotates by this amount.

Centre of explosion - digitise a point
  movements away/towards this point.

INBETWEEN   Select 1st file

Select 2nd file.

B8 - Use old files
  select by digitising corresponding permanent file menu squares, or by pressing button 8 to use files set previously.

No. of frames - input by BUTNUM.

No. of fair-in frames - input by BUTNUM.

No. of fair-out frames - input by BUTNUM.

PCURVE	Select path.
	Select data.
	B8 - use old files
	select by digitising corresponding permanent file menu square or by pressing button 8 to use files set previously.
	Digitise centre - digitise point
	B1 - no rotation - rigid movement
	B2 - with rotation - tracked movement, this point follows the path description.
	Dscale - input by BUTNUM - amount by which each frame is increased in size.

A.2 Button Control

The control of the programs is effected through the use of the pen-button interrupts. Unless indicated otherwise the buttons have the following meanings appended to them:

1 - erase screen, generate and display new frame

2 - generate and display new frame (superimpose frames)

7 - skip frame

8 - exit

The coordinate display unit on the table is used as a frame counter, showing the number of the frame being currently processed.

A.3 BUTNUM

The functions use a facility called BUTNUM to input numerical parameters. This is done by using the pen buttons on the cursor. There are 8 buttons on the cursor and by pressing one or a sequence of buttons a number can be input.

Button no.	Value
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8,1	-
8,2	8
8,3	9
8,4	0
8,5	.
8,6	E
8,7	enter number
8,8	rub out mistake

APPENDIX B

PHOTOGRAPHING THE CATHODE RAY TUBE

B.1 Film Gauge

35mm - Satisfies most professional requirements. Video transfer better quality than 16mm but expensive equipment, expensive film, bulky and probably union problems.

16mm - Blow ups from 16 to 35mm always lower in quality than 35mm original. However, acceptable by most modern standards, easier to handle and experiment with, less bulky, less costly to operate.

B.2 Camera

Two types of cameras can be used for the purpose of photographing the screen: General purpose (Bolex) or scientific (Vinten).

Basically there are few significant differences in most modern camera bodies. Most reputable manufacturers can produce a reliable, high precision drive and gating mechanism.

Main differences are type of motor drive (spring/electric: all spring have electric motor option). Optical systems (view finder: reflex viewing, reflex focusing, non reflex, semi-silvered prism, mirror shutter, boresights) and small details: single exposure, time exposure (both essential in our case), lens mount (preferably 'C' type), cariable shutter, back rewind and range of accessories.

Most features in the Bolex are also available in the Vinten's, the main practical difference being in the viewfinder.

Bolex uses a reflex viewfinder system. This allows positioning of camera, framing and focusing at any time.

The Vinten has no viewfinder. It uses a boresight. Requires a much more solid mount and once set should not be moved. The magazine must be removed every time the camera is repositioned, fogging a few frames.

They both have single frame release, instantaneous or time exposure.

B.3 Lenses

There are lenses especially designed to photograph CRT traces. These are not necessarily expensive and should give better results compared with ordinary lenses of the same make. But are unlikely to be remarkable when compared with high quality manufacturers' standard lenses. Some of these are in fact far superior to those 'special' lenses.

In general, quality is proportional to price.

B.4 Films

General purpose film: e.g. Ilford FP4. Although an excellent film, it is for general purpose photography, not recommended for high contrast high resolution work. Good results for not very high contrast line tests.

Special scientific film, Kodak Dacomatic A, the film used by the ULCC microfilm centre. The disadvantages of this film are low speed, special processing by Kodak and no immediate delivery in the U.K. Can be obtained by special order from U.S.A., minimum order quantity 12000ft at £236.70, delivery time approximately 14 weeks. Ilford makes a similar film but orders must also absorb 12000ft.

The Dacomatic has peak spectral sensitivity at the wavelength of

525nm. The maximum emission of green phosphor is in the range 500-525nm. This is indicated to be the best film for the job.

B. 5 Exposure

Tests carried out were not very stringent just to obtain approximate exposure for line tests.

Normal camera speeds are too high for good exposures. Exposures were at single frame, at 12 f.p.s. setting which on the Bolex corresponds to 1/40 secs. This forces the use of maximum aperture of lens (f 1.4) which seldom gives best results and still under exposed.

For longer exposures stop watch timing was required, which does not produce consistent results.

A computer controlled motor/timer is the solution here. A motor which allows a wider range of speeds can be bought for approximately £15 from Jim Smith Rostrum Limited, London. It requires slight modification to the camera, 3 gears and two microswitches. With the Vinten a slightly modified intervalometer could do the job.

The range of exposure required is 1/20 sec to 5 secs.

B. 6 Processing

Results from processing laboratories have been inconsistent, especially prints. Some high contrast prints had, in fact, less contrast than previous standard prints. They also increase the halo round the outlines of pictures probably caused by over exposure in the copying process.

APPENDIX C

CUBIC SPLINE

This appendix describes the mathematics used in deriving the equations for a cubic fit spline. It is a simplified version of the method described by Nutbourne (20) giving continuity of slope and curvature between spans.

The equations used are parametric form so they can be generalised to any number of dimensions, e.g. for three-dimensions

$$X = A + Bt + Ct^2 + Dt^3$$

$$Y = E + Ft + Gt^2 + Ht^3$$

$$Z = I + Jt + Kt^2 + Lt^3$$

C.1 Consider a single span between two adjacent points (Fig C-1) . For a general equation

$$U = A + Bt + Ct^2 + Dt^3 \qquad (1)$$

$$U' = B + 2Ct + 3Dt^2 \qquad (2)$$

$$U'' = 2C + 6Dt \qquad (3)$$

$$U''' = 6D \qquad (4)$$

The third derivative is constant throughout the span. Taking the second derivative and ascribing subscripts

$$U''_i = 2C + 6Dt_i$$

$$U''_{i+1} = 2C + 6Dt_{i+1}$$

subtracting

$$U''_{i+1} - U''_i = 6D\,\Delta t_i \quad \text{where } \Delta t_i = t_{i+1} - t_i$$

substituting in (4)

$$U''' = \frac{U''_{i+1} - U''_i}{\Delta t_i} \qquad (5)$$

Using Taylor's expansion

$$U_{i+1} = U(t_i+\Delta t_i) = U_i + \Delta t_i\, U_i' + \frac{\Delta t_i^2}{2} U_i'' + \frac{\Delta t_i^3}{6} U_i'''$$

$$U_i = U(t_{i+1} - \Delta t_i) = U_{i+1} - \Delta t_i\, U_{i+1}' + \frac{\Delta t_i^2}{2} U_{i+1}'' - \frac{\Delta t_i^3}{6} U_{i+1}'''$$

The fourth derivative is zero because U''' is a constant.

Rearranging:

$$\frac{U_{i+1} - U_i}{\Delta t_i} = U_i' + \frac{\Delta t_i}{2} U_i'' + \frac{\Delta t_i^2}{6} U'''$$

$$\frac{U_{i+1} - U_i}{\Delta t_i} = U_{i+1}' + \frac{\Delta t_i}{2} U_{i+1}'' + \frac{\Delta t_i^2}{6} U'''$$

subscript for U''' can be dropped because it is a constant.

Let $\Delta U_i = U_{i+1} - U_i$, substituting U''' from (5) gives:

$$\frac{\Delta U_i}{\Delta t_i} = U_i' + \frac{\Delta t_i}{2} U_i'' + \frac{\Delta t_i}{6}(U_{i+1}'' - U_i'')$$

$$\frac{\Delta U_i}{\Delta t_i} = U_{i+1}' - \frac{\Delta t_i}{2} U_{i+1}'' + \frac{\Delta t_i}{6} (U_{i+1}'' - U''_i)$$

which give

$$S_i = U_i' + \frac{\Delta t_i}{6} (2 U_i'' + U_{i+1}'') \qquad (6)$$

$$S_i = U_{i+1}' - \frac{\Delta t_i}{6} (U_i'' +2 U''_{i+1}) \qquad (7)$$

where $S_i = \dfrac{\Delta U_i}{\Delta t_i}$

Now consider two spans as in Fig C-2 at point (i+1) U' and U" must be continuous for a smooth curve. A recurrence equation is required whereby one of the characteristics of the curve can be propagated for a given initial condition.

Applying equation (6) to the second span and equation (7) to the first span

$$S_{i+1} = U'_{i+1} + \frac{\Delta t_{i+1}}{6}(2\,U''_{i+1} + U''_{i+2})$$

$$S_i = U'_{i+1} - \frac{\Delta t_i}{6}(U''_i + 2\,U''_{i+1})$$

Since at point (i+1) U' and U" are common, subtract

$$(S_{i+1} - S_i) = \frac{\Delta t_{i+1}}{6}(2\,U''_{i+1} + U''_{i+2}) + \frac{\Delta t_i}{6}(U''_i + 2\,U''_{i+1})$$

$$6(S_{i+1} - S_i) = 2\,U''_{i+1}(\Delta t_{i+1} + \Delta t_i) + \Delta t_i\,U''_i + \Delta t_{i+1}\,U''_{i+2}$$

dividing by Δt_{i+1} and rearranging

$$U''_{i+2} = \frac{6}{\Delta t_{i+1}}(S_{i+1} - S_i) - \frac{\Delta t_i}{\Delta t_{i+1}}U''_i - 2\,U''_{i+1}\left(1+\frac{\Delta t_i}{\Delta t_{i+1}}\right)$$

let $r_i = \dfrac{\Delta t_i}{\Delta t_{i+1}}$

therefore $$U''_{i+2} = \frac{6(S_{i+1} - S_i)}{\Delta t_{i+1}} - r_i\,U''_i - 2\,U''_{i+1}(1 + r_i) \qquad (8)$$

Equation (8) is the required recurrence equation from which successive values of U" can be calculated. For i = 1 and 2 the values of U" are determined from boundary conditions. Equation (8) can be represented in the form:

$$U''_{i+1} = C1_{i+2} + C2_{i+2}\,U''_i + C3_{i+2}\,U''_{i+1}$$

where

$$C1_{i+2} = \frac{6(S_{i+1} - S_i)}{\Delta t_{i+1}} = \frac{6}{\Delta t_{i+1}}\left(\frac{\Delta U_{i+1}}{\Delta t_{i+1}} - \frac{\Delta U_i}{\Delta t_i}\right)$$

$$C2_{i+2} = -r_i = -\frac{\Delta t_i}{\Delta t_{i+1}}$$

$$C3_{i+2} = -2\,(1 + r_i) = -2\left(1 + \frac{\Delta t_i}{\Delta t_{i+1}}\right)$$

The terms in the coefficient equations are all known in advance so the coefficients can be calculated and stored in three arrays for i = 3 and i = n.

The form of the equation is that of a propagation problem. Results are marched out from an initial boundary condition. Therefore errors are also marched out. In order to make the curve meet the end conditions a corrector weighting equation is needed, otherwise severe rippling will occur on the far end.

The weighting equation is similar to the quartic used in the Chebyshev method. A secondary curve is chosen where the values of U are all zero at the data points(Fig C-3).

For all $U_i = 0$ recurrence equation (8) becomes

$$US''_{i+2} = r_i\, US''_i - 2US''_{i+1}\, (1+r_i) \qquad (9$$

S for secondary.

$$US''_{i+1} = C2_{i+2}\, US''_i + C3_{i+2}\, US''_{i+1}$$

This secondary curve is used to update the primary curve.

New primary curve = Old primary curve + k* secondary curve where r is a constant determined from the end conditions.

Considering the 2nd derivatives

$$U'_{(new)} = U''_{(old)} + kUS'' \qquad (10)$$

The primary curve is recomputed until convergence occurs. Usually four iterations are sufficient.

C.2 Boundary Conditions

A choice of end conditions can be considered for the spline: by choosing specific values of the first or second derivatives at the end points or by fitting a specific equation to the end points.

Two conditions are considered: one in which the two adjacent spans at the end have a common cubic equation and one in which the first derivative has a specified value.

C.2.1 Common Cubic

For the start condition consider the first three points to lie on the same cubic. Therefore the values of U" are linearly related (Fig C-4).

$$U''_1 \, \Delta t_2 + U''_3 \, \Delta t_1 = U''_2 \, (\Delta t_1 + \Delta t_2)$$

divide by Δt_2

$$U''_1 + r_1 \, U''_3 = U''_2 \, (1 + r_1)$$

from (8) for i = 1

$$U''_3 = \frac{6}{\Delta t_2} (S_2 - S_1) - r_1 \, U''_1 - 2 \, (1 + r_1) \, U''_2$$

$$U''_1 + \frac{6}{\Delta t_2} \, 1 \, (S_2 - S_1) - r_1^{\,2} U''_1 \, (1 + r_1) \, U''_2 =$$

$$= U''_2 \, (1 + r_1)$$

$$U''_1 (1 - r_1^{\,2}) + \frac{6 r_1}{\Delta t_2} \, (S_2 - S_1) = U''_2 (1 + r_1)(1 + 2 r_1)$$

divide by $(1 + r_1)$

$$U''_1 (1 - r_1) + \frac{6}{\Delta t_2} \, \frac{1}{1 + r_1} \, (S_2 - S_1) = U''_2 (1 + 2 r_1) \qquad (11)$$

for primary curve set arbitrary condition $U''_1 = 0$

$$U''_2 = \frac{6}{\Delta t_2} \, \frac{r_1}{1 + r_1} \, \frac{(S_2 - S_1)}{(1 + 2 r_1)}$$

for secondary curve (11) becomes

$$US''_1\ (1-r_1) = US''_2\ (1+2\,r_1)$$

An arbitrary value for US''_1 must be chosen. If US''_1 were set to zero degeneration to a straight line would occur.

let $US''_1 = 1$

$$US''_2 = \frac{(1-r_1)}{(1+2\,r_1)}$$

Therefore the start conditions for cubic are

$$(12)\left\{\begin{array}{ll} U''_1 = & 0 \\ U''_2 = & \dfrac{6}{\Delta t_2}\ \dfrac{r_1}{(1+r_1)(1+2\,r_1)}\quad (S_2-S_1) \\ US''_1 = & 1 \\ US''_2 = & \dfrac{(1-r_1)}{(1+2\,r_1)} \end{array}\right.$$

C.2.2 Fixed Slope (Encastered spline)

The second start condition used is by specifying a starting U' value, i.e. a fixed slope.

For the condition in which $U'_1 = \alpha$

from (6) $S_1 = \alpha_1 + \dfrac{\Delta t_1}{6}\ (2U''_1 + U''_2)$

arbitrarily set $U''_1 = 0$

$$U''_2 = \frac{6(S_1 - \alpha_1)}{\Delta t_1}$$

For the secondary curve $S_1 = 0$. Also α must be zero in order that the slope of the primary remains unchanged when the secondary is added to it.

set $US''_1 = 1$

$US''_2 = -2$

Therefore for an encastered spline the initial conditions are

$$(13)\begin{cases} U''_1 = 0 \\ U''_2 = \dfrac{6(S_1 - \alpha_1)}{t_1} \\ US''_1 = 1 \\ US''_2 = -2 \end{cases}$$

C2.3 End Conditions

The cubic spline is used again to determine the end conditions. U" at the last three points lie on a straight line(Fig C-5).

From equation(10)

$$\bar{U}''_{m-2} = \bar{U}''_{n-2} + kUS''_{m-2}$$

$$\bar{U}''_{m-1} = U''_{m-1} + kUS''_{m-1}$$

$$\bar{U}''_{m} = U''_{m} + kUS''_{m}$$

To lie on a straight line

$$\frac{\bar{U}''_m - \bar{U}''_{m-1}}{t_{m-1}} = \frac{\bar{U}''_{m-1} - \bar{U}''_{m-2}}{t_{m-2}}$$

$$r_{n-2}\ (U''_m - U''_{m-1}) + k\ (US''_m - US''_{m-1})$$

$$= U''_{m-1} - U''_{n-2} + k\ (US''_{m-1} - US''_{n-2})$$

therefore

$$k = \frac{(U''_{n-1} - U''_{n-2}) - r_{n-2}\ (U''_m - U''_{m-1})}{r_{n-2}(US''_m - US''_{m-1}) - (US''_{n-1} - US''_{m-1})} \qquad (14)$$

This equation gives the value of k when the end values of U" and US" are known.

C.3 Procedure

The steps involved in generating the values of U" are:

1 - U''_1 and U''_2 are set to initial conditions (12) or (13)

2 - All U''_i's are calculated using equation (8)

3 - Repeat for US''_is using equation (9)

4 - Calculate k from U''_m, U''_{m-1}, U''_{m-2} and US''_m, US''_{m-1}, US''_{m-2}

5 - Recompute U''_i using equation (10)

6 - Repeat from 4 until convergence.

This procedure produces satisfactory results up to about 20 points above which it starts to deteriorate. This is because the 'clamping power' of k starts losing its effect once the distance from the start position increases. To avoid this long splines are divided into overlapping pieces 10 spans long, as shown in Fig C-6.

First spline is Cubic - Cubic type

Subsequent splines are Encastered - Cubic

Last spline is Encastered - Cubic.

Start conditions for subsequent splines are given by 6.

$$\alpha = U'_r = S_r - \frac{\Delta t_r}{6} (2U''_r + U''_{r+1}) \qquad (15)$$

for r = 6, 11, 16, ... r+5

C.4 Span Equations

Once all U" s have been computed the coefficients of the span equations can be calculated.

To avoid excessively large coefficients a moving origin is used. The first point in the span is always at t = 0, i.e. $t_i = 0$ always.

For span i:

$$U_i = A_i + B_i t + C_i t^2 + D_i t^3$$

Therefore

$$(16)\begin{cases} A_i = U_i \\ B_i = U'_i = S_i - \frac{\Delta t_i}{6}(2U''_i + U''_{i+1}) \\ C_i = U''_i/2 \\ D_i = \frac{1}{6\Delta t_i}(U''_{i+1} - U''_i) \end{cases}$$

The use of parametric equations allows X, Y and Z to vary in terms of one single variable t. For general purposes t can be a constant and normalised to unity.

Rewriting all equations:

$$(8) \qquad U''_{i+2} = 6(\Delta U_{i+1} - \Delta U_i) - U''_1 - 4U''_{i+1}$$

$$U''_{i+2} = 6(U_{i+2} - 2U_{i+1} + U_i) - U''_i - 4U''_{i+1}$$

$$(9) \qquad US''_{i+2} = -US''_i - 4US''_{i+1}$$

$$(12)\begin{cases} U''_1 = 0 \\ U''_2 = U_3 - 2U_2 - U_1 \\ US''_1 = 1 \\ US''_2 = 0 \end{cases}$$

$$(13)\begin{cases} U''_1 = 0 \\ U''_2 = 6(U_2 - U_1 - U'_1 \\ US''_1 = 1 \\ US''_2 = -2 \end{cases}$$

$$(14) \qquad k = -\frac{U''_n - 2U''_{n-1} + U''_{n-2}}{US''_m - 2US''_{m-1} + US''_{m-2}}$$

$$(15) \qquad \alpha = U'_r = (U_{r+1} - U_r) - \frac{(2U''_r + U''_{r+1})}{6}$$

$$(16)\quad \begin{cases} A_i = U_i \\ B_i = (U_{i+1} - U_i)-(2U''_i + U''_{i+1}) / 6 \\ C_i = U''_i /2 \\ D_i = (U''_{i+1} - U''_i)/6 \end{cases}$$

Two special conditions may arise in curve fitting, When only two or three points exist to define the curve.

For the first one the curve is simply a straight line.

For the second case a parabola is fitted through the three points (Fig C-7).

$U = P_1 + P_2t + P_3 t^2$

$U_1 = P_1$

$U_2 = P_1 + P_2 + P_3$

$U_3 = P_1 + 2P_2 + 4P_3$

give $P_1 = U_1$

$P_2 = (4U_2 - 3U_1 - U_3)/2$

$P_3 = (U_1 - 2U_2 + U_3)/2$

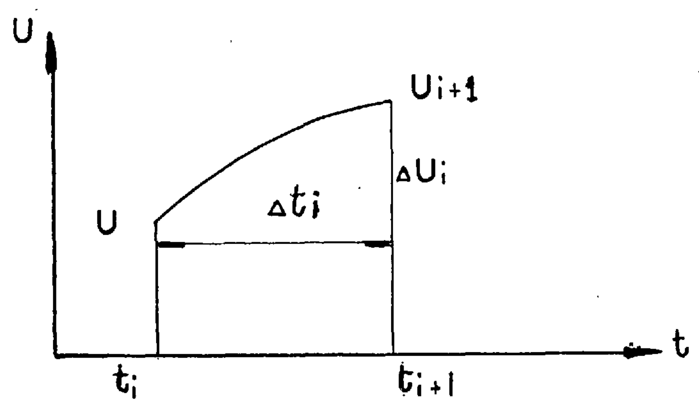


Fig C-1 Single span in function U = f(t)

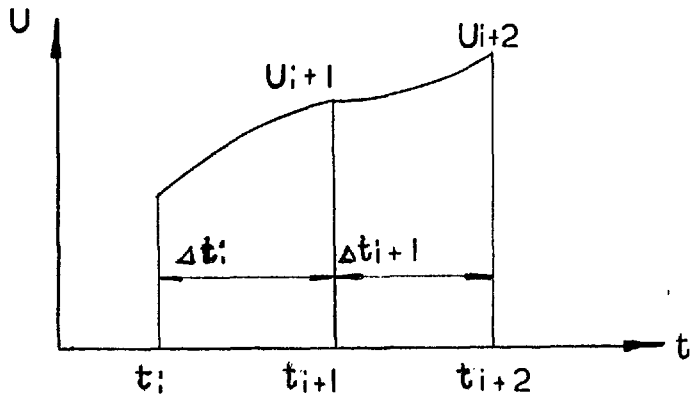


Fig C-2 Two spans

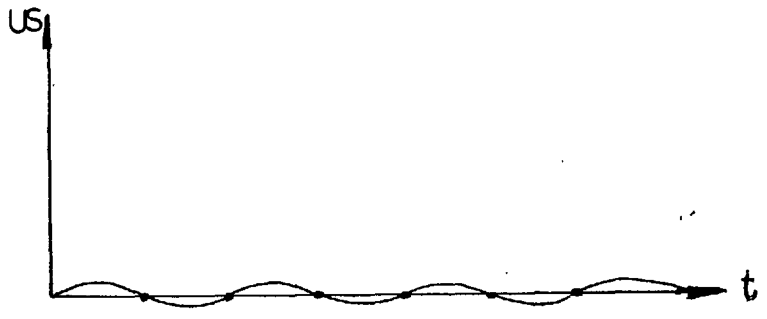


Fig C-3 Secondary weighting equation

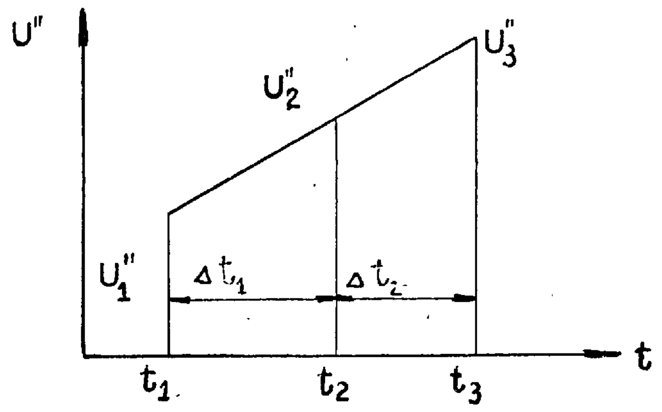


Fig C-4 Start conditions for cubic spline

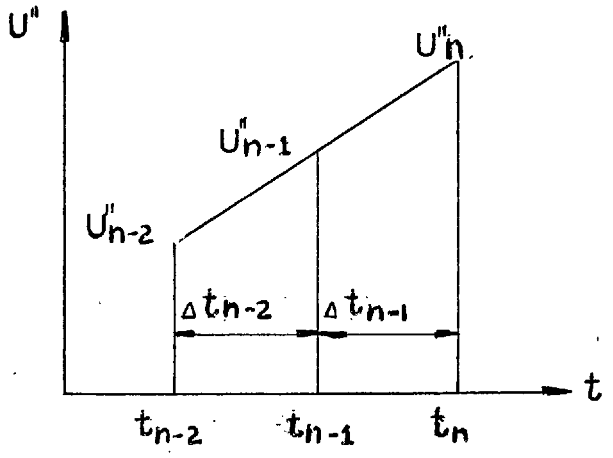


Fig C-5 End conditions for cubic spline

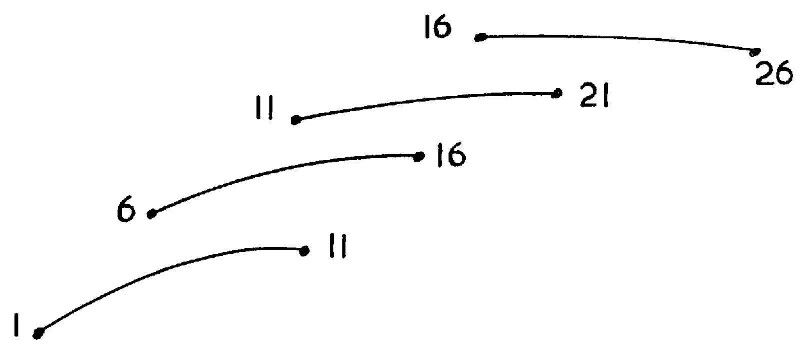


Fig C-6 Overlapping long splines

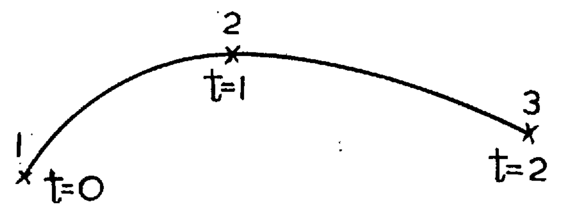


Fig C-7 Parabola through three points

APPENDIX D

SUBROUTINE RECORD

D.1 INTRODUCTION

This appendix describes the entries in subroutine RECORD which handles input and output of data records I, X, Y, Z or I, X, Y in the three-dimensional data and display files. Also extracts and sets the working parameters in the data records (INT, LEV, NPEN, LTYPE).

The routine works on the first in, first out principle in auto-incremental mode by default, but facilities enabling random access of data records in files have been included. The internal data structure is transparent to the user.

D.2 I/O ENTRIES

```
CALL GETR (NRA, I, X, Y, Z)    for NRA = 1 and 2
CALL GETR (NRA, I, X, Y)       for NRA = 3
```

for NRA > 3 the call is ignored.

NRA is file number

I, X, Y < Z > the data record <> optional

This entry extracts data records in auto-incremental mode.

```
CALL SETR (NRA, I, X, Y, Z)
CALL SETR (NRA, I, X, Y)
```

Arguments defined as above.

Store next data record in auto-incremental mode.

```
CALL SETR (NRA, 0)
```

If I = 0 the routine writes off the core buffer and closes the respective NRA file. The coordinate terms are redundant.

D.3 POINTER ENTRIES

CALL GETIP (NRA, IDPI)

NRA - as above

IDPI - is the input data record sequence number (IDPI > 1)

Returns the pointer to the next record to be stored.

CALL SETIP (NRA, IDPI)

Arguments as above.

Sets pointer for the next record to be stored. This entry allows random storage of data records.

CALL GETOP(NRA, IDPO)

NRA as above

IDPO output data record sequence number (IDPO > 1)

Returns the pointer to the next record to be output.

CALL SETOP (NRA, IDPO)

Arguments as above

Sets pointer for the next record to be output. This entry allows random access of data records.

D.4 PARAMETER ENTRY

This entry sets or gets the values of the parameters in the data record. The call may have 2 or 5 arguments of the form:

CALL PARAMS (N, IPARAM(I)) I = 1 or 4

N > O set parameter

N < O get parameter

N	= 1	IPARAM = INT	(O-7)
	= 2	= LEV	(O-3)
	= 3	= NPEN	(O-3)
	= 4	= LTYPE	(O-3)
	= 5	= (INT, LEV, NPEN, LTYPE)	

The numbers in brackets are the range of values for each parameter.

D.5 CHANGE

This is an additional subroutine used in the random editing of data.

CALL CHANGE (NRA, IDP, I, X, Y, X)

CALL CHANGE (NRA, IDP, I)

NRA as above

IDP - record position pointer

I, X, Y, Z - new data record. (X, Y, Z) are optional, if they are not included only I is changed.

This subroutine replaces the contents of the data record pointed by IDP with the contents of the record argument(s).

APPENDIX E

CONSUR a program for automated contour-surface mapping - User's manual

E.1 Introduction

CONSUR is an automated contour mapping program. It constructs a smooth surface with data digitised from contour maps. Uses a gridding method to compute gridded data from randomly spaced input data. The user controls gridding and contouring.

E.2 Operating Procedures

Before digitising the contours the Z value for each line must be normalised to a maximum of 60. (A future version of the system will enable actual values of Z to be input).

Digitising should preferably be done in continuous mode. If not, as many points as possible should be input. This can be achieved by using either MINCE or CURFIT.

The sequence of operations is:

1 - Set input level to the normalised value of Z.

2 - Digitise respective contour lines.

3 - Repeat 1 and 2 until input is complete.

4 - Call CONSUR from the corresponding menu square (contour map must be in workspace. File, if to be saved as it will be overwritten).

5 - Enter parameters as described below (see examples).

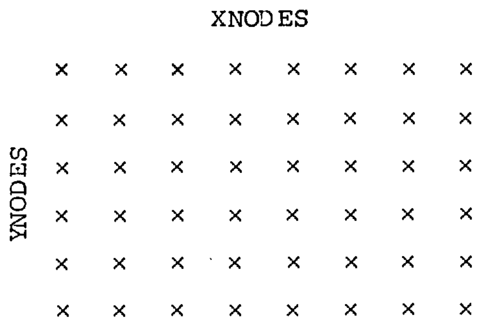
E.3 Parameters (typical values)

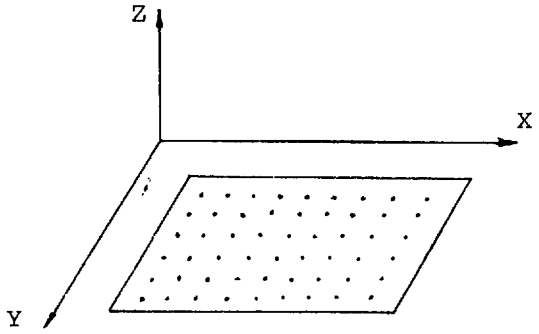
ZMAX (12)

Maximum height of surface above level zero (may be negative number for troughs).

XNODES, YNODES (30, 20)

Number of grid points on the X and Y axes respectively. Maximum grid area is 50 x 35.





XINS, YINS (30, 0)

Define the overall size of the picture (including deformation but excluding the sides added by SOLID = 1).

For XINS = x YINS = 0

The surface will be scaled so that the width of the picture is x inches.

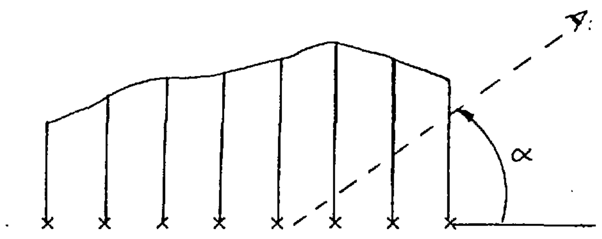
Similarly if XINS = 0 and YINS = y the surface will be scaled so that the height of the picture is y.

If XINS = x and YINS = y the surface will be scaled so that it is the largest size that will fit within an area x by y.

XINS = 30 equals the width of the screen.

ALPHA (30)

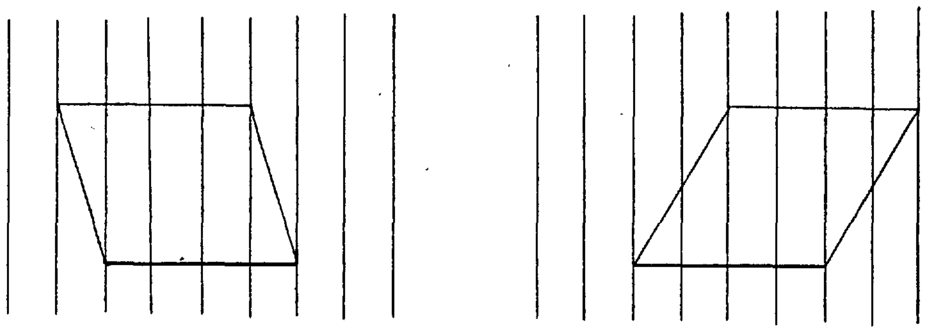
is the viewing angle in degrees.



DEFORM(50)

is the percentage deformation required for semi-oblique viewing.

e.g.



DEFORM = -25 DEFORM = 50

SOLID = 0 - gives hollow surface

1 - adds solid sides.

E.4 Reference

Communications of the ACM algorithm 420 February 1972.

"Hidden-Line Plotting Program", Hugh Williamson.

APPENDIX F

PERSPECTIVE TRANSFORMATION

Perspective is one of the most valuable methods of visualizing objects. The transformation is always used when displaying three-dimensional data but also in many cases with two-dimensional data to represent a plane lying in three-dimensional space.

Based on elementary optical laws it is easily represented in mathematical form.

Considering Fig E-1, the x, y, z axes correspond to the spatial cartesian coordinates. The system of reference axes used here is in the conventional 'right-handed reference directions' set of coordinates. If the x-y plane is made to coincide with the screen the positive z direction is out of the screen towards the observer.

For the purpose of the projection a plane PP with coordinates (X, Y) must be defined. The projection maps the point $\bar{P}$ to a point $\bar{S}$ on the projection plane PP. The projection plane in this case corresponds to the viewing screen, with X to the right and Y upwards.

The observer's eye is situated at the point $\bar{E}\left[E_x, E_y, E_z\right]$ and the line of sight makes angles α, β and γ with the spacial x, y and z axes respectively. (Coordinates with x, y, z subscripts are absolute space coordinates.) The angles α, β, γ can be supplied as their cosines, i.e. the direction cosines or by a point defined as the centre of attention, from which the direction cosines can be calculated. Normally the centre of attention is made to coincide with the origin or the volumetric centre of the object being observed.

Consider the plane of projection PP at a distance D from $\bar{E}$, normal to the line of sight. $\bar{R}$ (R_x, R_y, R_z) is the foot of the normal from $\bar{E}$ to the plane. The centre of attention if used instead of the direction cosines to define the line of sight can be made to lie on the projection plane and would be represented by $\bar{R}$ in this case. The scalar D is given by:

$$D = \overline{ER}$$

and the direction cosines

$$\cos\alpha = (E_x - R_x)/D$$

$$\cos\beta = (E_y - R_y)/D$$

$$\cos\gamma = (E_z - R_z)/D$$

The straight line from $\bar{E}$ to a point $\bar{P}\,[P_x, P_y, P_z]$ intersects PP at some point $\bar{S}\,[S_x, S_y, S_z]$, $\bar{S}$ being the perspective image of $\bar{P}$ in the plane PP with respect to $\bar{E}$.

To generate the perspective view of the object defined by points like $\bar{P}$ the spatial coordinates $[P_x, P_y, P_z]$ must be converted to the $[X, Y]$ coordinates of the screen.

Referring to Fig F-1, from geometry:

$$\bar{R} = \bar{E} + D\,[\cos\alpha, \cos\beta, \cos\gamma] \qquad (1)$$

The equation of the straight line $\overline{ESP}$ is given by

$$\frac{S_x - E_x}{P_x - E_x} = \frac{S_y - E_y}{P_y - E_y} = \frac{S_z - E_z}{P_z - E_z} = K \qquad (2)$$

where K is a constant. From (1)

$$\overline{ER} = D\,[\cos\alpha, \cos\beta, \cos\gamma] \qquad (3)$$

since $\bar{S}$ lies on plane PP

$$\overline{ER}.\overline{RS} = 0 \qquad (4)$$

but $\quad \overline{RS} = \overline{ES} - \overline{ER}$

$$\overline{ER}.(\overline{ES} - \overline{ER}) = 0$$

$$\overline{ER}.\overline{ES} = \overline{ER}.\overline{ER} = D^2 \qquad (5)$$

$$\overline{ES} = [S_x - E_x, S_y - E_y, S_z - E_z]$$

From (3) and (5)

$$D \left[\cos\alpha,\ \cos\beta,\ \cos \right] \cdot \left[S_x-E_x,\ S_y-E_y,\ S_z-E_z \right] = D^2$$

$$(S_x-E_x)\cos\alpha + (S_y-E_y)\cos\beta + (S_z-E_z)\cos\gamma = D \qquad (6)$$

Substituting from (2)

$$K(P_x-E_x \cos\alpha + K(P_y-E_y)\cos\beta + K(P_z-E_z)\cos\gamma = D$$

$$K = D/\left[(P_x-E_x)\cos\alpha + (P_y-E_y)\cos\beta + (P_z-E_z)\cos\gamma \right] \qquad (7)$$

from which K can be calculated as all other terms are known. Once the value of K is known $\overline{S}$ can be determined. From (2)

$$\overline{S} = \overline{E} + K(\overline{P} - \overline{E})$$

$$\begin{cases} S_x = E_x + K(P_x-E_x) \\ S_y = E_y + K(P_y-E_y) \\ S_z = E_z + K(P_z-E_z) \end{cases} \qquad (8)$$

Now express $\overline{S}$ in terms of the two-dimensional system of coordinates in PP $\left[X, Y \right]$. The equation of plane PP is given by (4)

$$(S_x-R_x)\cos\alpha + (S_y-R_y)\cos\beta + (S_z-R_z)\cos\gamma = 0 \qquad (9)$$

Consider the horizontal plane H given by $y-R_y = 0$. The intersection of planes PP and H is taken as the X direction (Fig F-2).

Equation of PP

$$x\cos\alpha + y\cos\beta + z\cos \quad = R_x\cos\alpha + R_y\cos\beta + R_z\cos\gamma \qquad (10)$$

Equation of H

$$y-R_y = 0$$

Let x, y, z be a point on the line of intersection

$$y = R_y$$

From (10)

$$x\cos\alpha + z\cos\gamma = R_x\cos\alpha + R_z\cos$$

$$x = R_x + (R_z-z)\frac{\cos\gamma}{\cos\alpha}$$

Which gives the equation of line along X

$$\frac{x-R_x}{\cos \gamma} = \frac{y-R_y}{0} = \frac{z-R_z}{-\cos \alpha} \qquad (11)$$

A vector in the X direction is given by $(\cos \gamma, \phi, -\cos \alpha)$.

Let the unit vector in X direction be $\bar{U}_x$

$$\bar{U}_x = \pm\left[\frac{\cos \gamma}{\sqrt{\cos^2\alpha+\cos^2}}, 0, \frac{-\cos \alpha}{\sqrt{\cos^2\alpha+\cos^2}}\right]$$

but since $\cos^2\alpha + \cos^2\gamma = 1 - \cos^2\beta = \sin^2\beta$

$$\bar{U}_x = S_i\left[\frac{\cos \gamma}{\sin \beta}, 0, -\frac{\cos \alpha}{\sin \beta}\right] \qquad (12)$$

with $S_1 = \pm 1$ to be defined later.

A unit vector in the Y direction must also be defined.

$$\text{Let } \bar{U}_y = [a, b, c] \qquad (13)$$

$\bar{U}_y$ must satisfy the following conditions:

$$\bar{U}_x . \bar{U}_y = 0$$

$$|\bar{U}_y| = 1$$

$$\bar{U}_y . \overline{OR} = 0$$

i.e.

$$a \cos \gamma - c \cos \alpha = 0 \qquad (14)$$

$$a^2 + b^2 + c^2 = 1 \qquad (15)$$

$$a \cos \alpha + b \cos \beta + c \cos \gamma = 0 \qquad (16)$$

(14) gives

$$a = c\,\frac{\cos \alpha}{\cos \gamma}$$

substitute in (16)

$$c\,\frac{\cos^2 \alpha}{\cos \gamma} + c \cos \gamma = -b \cos \beta$$

$$b = -c\left(\frac{\cos^2\alpha+\cos^2\gamma}{\cos\beta \;\; \cos \gamma}\right) = -c\,\frac{\sin^2 \beta}{\cos\beta \; \cos \gamma}$$

substitute in (15)

$$c^2 \frac{\cos^2\alpha}{\cos^2\gamma} + c^2 + c^2 \frac{\sin^4\beta}{\cos^2\beta \cos^2} = 1$$

$$c^2 = \frac{\cos^2\beta \cos^2\gamma}{\sin^2\beta}$$

$$c = S_2 \frac{\cos\beta \cos\gamma}{\sin\beta}$$

where $S_2 = \pm 1$

substituting back

$$a = S_2 \frac{\cos\alpha \cos\beta}{\sin\beta}$$

$$b = -S_2 \sin\beta$$

$$\overline{U}_y = S_2 \left[\frac{\cos\alpha \cos\beta}{\sin\beta}, -\sin\beta, \frac{\cos\beta \cos\gamma}{\sin\beta}\right] \qquad (17)$$

Now the signs of S_1 and S_2 must be determined.

For Y upwards

$$\overline{U}_y . \overline{j} > 0$$

where $\overline{j}$ is the unit vector in the y direction.

$$-S_2 \sin\beta > 0$$

$$S_2 \sin\beta < 0$$

since $\quad 0 < \beta < \pi$

$$0 < \sin\beta < 1$$

then $\quad S_2 = -1 \qquad (19)$

Also $\overline{U}_x \times \overline{U}_y$ is normal to plane $\overline{PP}$, i.e. parallel to the line of sight and in the opposite direction.

$$\overline{U}_x \times \overline{U}_y = -\overline{OR}/D = -\left[\cos\alpha, \cos\beta, \cos\gamma\right] \qquad (20)$$

$$= S_1 S_2 \left[\frac{\cos\gamma}{\sin\beta} \overline{i} + 0\overline{j} - \frac{\cos\alpha}{\sin\beta} \overline{k}\right] \times$$

$$\left[\frac{\cos\alpha \cos\beta}{\sin\beta} \overline{i} - \sin\beta \overline{j} + \frac{\cos\beta \cos\gamma}{\sin\beta} \overline{k}\right]$$

Consider $\overline{i}$ term

$$\overline{j} \times \overline{k} \cos\alpha \; S_1 S_2 = -\cos\alpha \; \overline{i}$$

$$S_1 S_2 = -1$$

and $$S_1 = 1 \qquad (21)$$

Substituting for the values of S_1 and S_2 in (12) and (17)

$$\overline{U}_x = \left[\frac{\cos\gamma}{\sin\beta}, \; 0, \; -\frac{\cos\alpha}{\sin\beta}\right] \qquad (22)$$

$$\overline{U}_y = \left[-\frac{\cos\alpha \cos\beta}{\sin\beta}, \; \sin\beta, \; -\frac{\cos\beta \cos\gamma}{\sin\beta}\right] \qquad (23)$$

Components of $\overline{S}$ in terms of the new axes will be

$$X = \overline{RS}.\overline{U}_x = (S_x - R_x)\cos\gamma - (S_z - R_z)\cos\alpha \;/\sin\beta \quad (24)$$

$$Y = \overline{RS}.\overline{U}_y = -(S_x - R_x)\cos\alpha \frac{\cos\beta}{\sin\beta} + (S_y - R_y)\sin\beta -$$

$$- (S_z - R_z)\cos\gamma \frac{\cos\beta}{\sin\beta}$$

From (9)

$$(S_x - R_x)\cos\alpha + (S_z - R_z)\cos\gamma = -(S_y - R_y)\cos\beta$$

which gives

$$Y = (S_y - R_y)/\sin\beta \qquad (25)$$

Recapitulate: Given eye position $\overline{E}\left[E_x, E_y, E_z\right]$

direction of sight $\left[\alpha, \beta, \gamma\right]$

distance from PP = D

coordinates of P $\left[P_x, P_y P_z\right]$

compute:

$$(i) \quad \begin{cases} R_x = E_x + D\cos\alpha \\ R_y = E_y + D\cos\beta \\ R_z = E_z + D\cos\gamma \end{cases}$$

(ii) $K = D/\ (P_x-E_x)\cos\alpha + (P_y-E_y)\cos\beta + (P_z-E_z)\cos\gamma$

(iii) $$\begin{cases} S_x = E_x + K(P_x-E_x) \\ S_y = E_y + K(P_y-E_y) \\ S_z = E_z + K(P_z-E_z) \end{cases}$$

(iv) $$\begin{cases} X = \left((S_x-R_x)\cos\gamma - (S_z-R_z)\cos\alpha\right)/\sin\beta \\ Y = (S_y-R_y)/\sin\beta \end{cases}$$

The above works for any case except when $\sin\beta = 0$, i.e. $\beta = 0$, π when the last two equations degenerate. The line of sight is vertical and the above transformation cannot be used. PP coincides with plane $y-R_y = 0$. For such cases the X axis can be defined by the intersection of PP and $z-R_z = 0$ for which the singular transformation can be used (Fig F-3):

$$X = S_x-R_x$$

$$Y = (R_z-S_z)\cos\beta$$

The above applies for both cases when $\beta = 0$ and π.

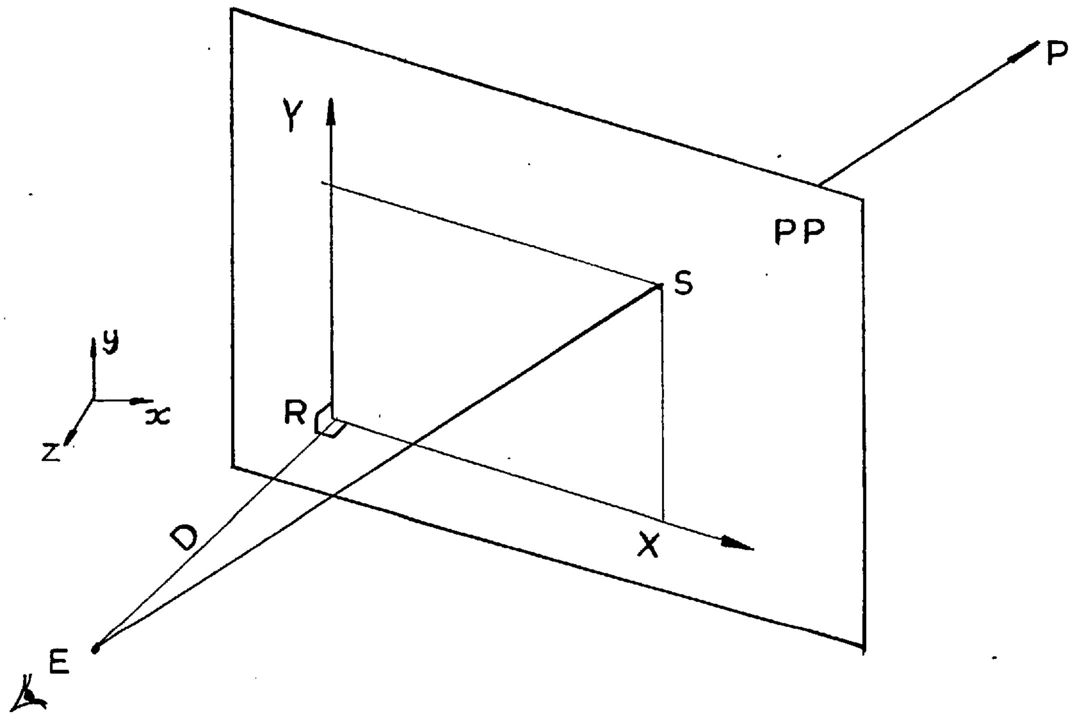


Fig F-1 Projection on plane PP

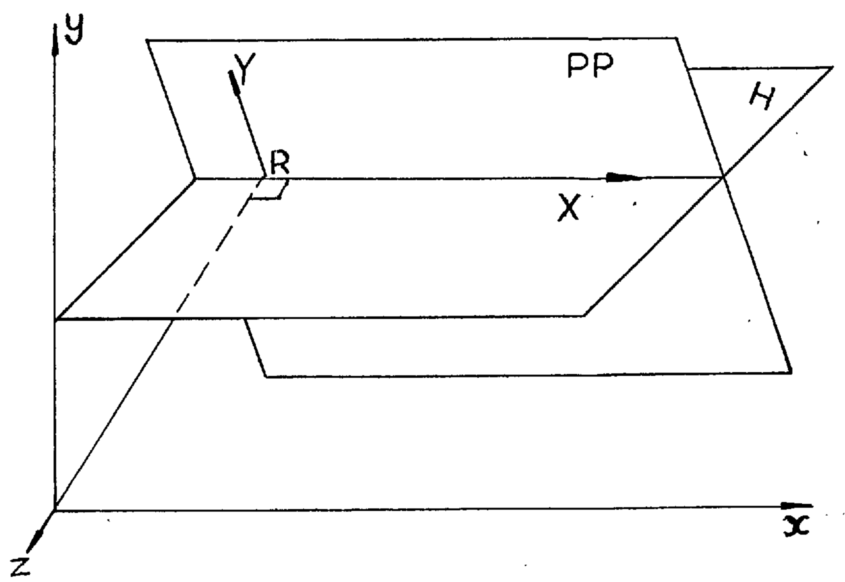
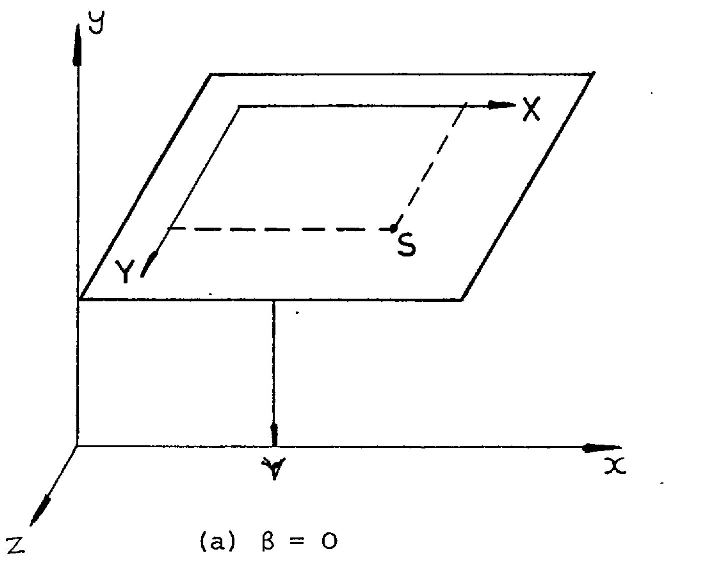
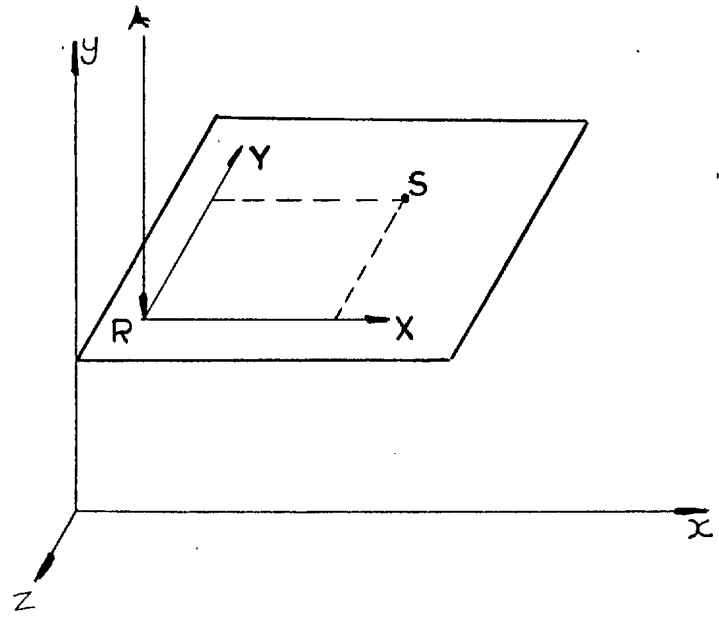


Fig F-2 Intersection of planes PP and H



(a) β = 0



(b) β = π

Fig F-3 Singular transformations

APPENDIX G

NOTES ON SYMBOLS

G.1 INTRODUCTION

Each symbol overlay may handle one or more symbols. Each symbol has an entry point in the overlay determined from ISUB. The overlay number (IVNO) and the parameter ISUB are returned by the menu mapping routine (MENMAP) when the symbol is first time evoked on the corresponding menu square.

ISFLG is the symbol flag which determines where to go within each symbol section.of the program and also works as a counter in dictating the number of the point being entered. The maximum number of points required to define an individual symbol is monitored by the symbol program and once this number has been reached, the program will generate the symbol and reset ISFLG.

ISFLG can only be changed by the symbol program. DIGOV also uses it but does not modify its contents.

There are two message lines for symbols in MESS (11-15)
and MESS (16-20).

The first one containing the symbol mode and the second one the parameter request.

Symbol data is transferred and stores via the arrays XSYM (5), YSYM (5), ZSYM (5).

G.2 EXAMPLE (circle mode)

ISFLG is initially set to zero.

When the symbol is requested from the menu IVNO and ISUB are returned by MENMAP and the appropriate overlay loaded in core.

The messages MESS (11-15) = 'CIRCLE MODE'

MESS (16-20) = 'CENTRE POINT'

are set up.

ISFLG is incremented and control returned to DIGOV. The symbol overlay itself must be stacked (CALL STACK (INVO, ISUB)) before return by CALL OVLINK (2) (DIGOV IVNO = 2).

For each point digitised the symbol overlay is recalled. For the first point ISFLG = 1.

Therefore XSYM (ISFLG) = X1

YSYM (ISFLG) = Y1

ZSYM (ISFLG) = Z1

MESS (16-20) becomes '2ND POINT' and ISFLG = 2.

When the second point is digitised the entry has again to be prestacked and entry within the symbol is determined by ISFLG = 2.

Again XSYM (ISFLG) = X1

YSYM (ISFLG) = Y1

ZSYM (ISFLG) = Z1

Since the number of points required by circle has been reached the program branches to the symbol generation section and calculates the points lying on the circle. These are transformed and stored into the data and display files and displayed in the current ND mode.

Symbols are stored in expanded data form with delimiting markers

ICODE = 3	start of symbol
ICODE = 4	end of symbol

On completion of the symbol the program resets ISFLG to 1 and sets the messages back to

MESS (16-20) = 'CENTRE POINT'

stacks the symbol and returns.

Symbols can be interrupted by exiting to another symbol mode and clearing ISFLG. With ISFLG = O the values XSYM, YSYM, ZSYM are over-written by the new symbol. The symbol delimiters should only be stored at the symbol generation stage.

On first entry (ISFLG = O) the overlay stack must always be cleared of any previous symbol entry.

To add symbols to the system only the above rules need be observed, otherwise the programs follow the same norm as other overlays.

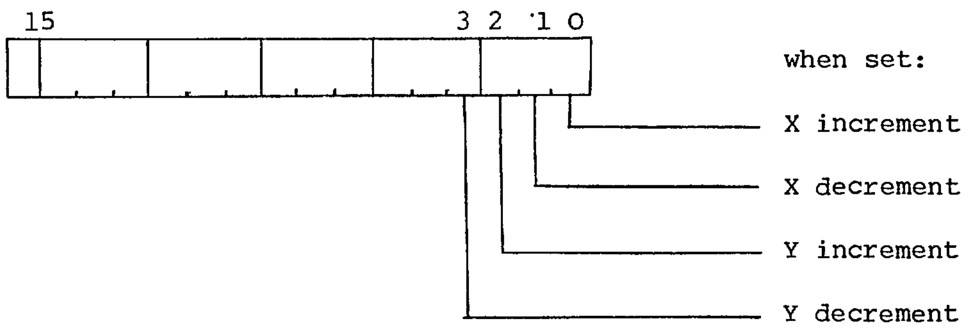
APPENDIX H

BACKGROUND PLOTTING WITH THE D.C. PLOTTER

H.1 CAMAC MODULES

H.1.1 Error Controller EC 1004 (M7)

ECSND - M7, A1, F17 (164712) - write increments



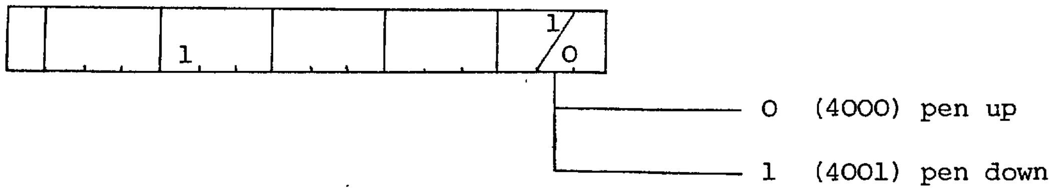
Each pulse corresponds to one step.

9 pulses = 1 unit = 1/40mm

M7, A0, F0 (164710) - read logical OR of X and Y registers (error)

H.1.2 Drawing Table Interface (M5)

M5, A2, F16 (164520) -pen position register



H.2 DATA STRUCTURE

The data is stored in random I, X, Y in the workspace. The program converts this data into plotter format and transfers it to RA8.

Vectors are stored consecutively as 5 integer words, therefore each block contains 55 vectors with one spare word. Blocks of data are accessed by the standard RAREAD and RAWRIT subroutines.

Data Format

IPEN	DOM	NDOM	IG	IET

IPEN	=	0 pen up
		1 pen down
		-1 end of file
DOM	-	Dominant increment/decrement
NDOM	-	Non-dominant increment/decrement
IG	-	Gradient (< 1: * 32768)
IET	-	Dominant vector (+ve)
	-	The method used to generate the vector is called Digital Differential Analyser. It generates points along the axis of greater delta (x or y). Thus the line length estimate is the absolute value of the largest delta (IET). The ratio $\frac{\Delta y}{\Delta x}$ or $\frac{\Delta x}{\Delta y}$ is always less than one (IG) and determines the spacing in the non-dominant direction.

H.3 SOFTWARE DESCRIPTION

1 - BKSET

2 - PLWRIT

3 - BKPLOT

H.3.1 Overlay BKSET

This program transfers data from workspace (RA1) to the background plotting file (RA8).

The data is scaled, windowed and written off in plotter format.

It calls BKPLOT to initialize the background plotting routine.

Plot size is determined by PSETUP.

H.3.2 Subroutine PLWRIT (IPEN, IXSN, IG, IET, INX)

IPEN -	0 pen up
	1 pen down
	-1 end of file
IXSN -	0 +ve DX
	1 -ve DX
IYSN -	0 +ve DY
	1 -ve DY
IG -	gradient (< 1)
IET -	dominant DX or DY (in 1/10mm)
INX -	1 X dominant
	0 Y dominant

This subroutine converts the data to plotter format and stores it in RA8 as 5 word records as described in the data structure.

It returns the new values of DX and DY in the two words of IXSN and IYSN respectively.

H.3.3 Subroutine BKPLOT

(CALL BKPLOT)

(CALL INT)

This double entry subroutine does the actual plotting.

Status of plotter is determined by IFLAG.

IFLAG -	0 idle
	1 busy

CALL BKPLOT - zero all pointers and set IFLAG to busy

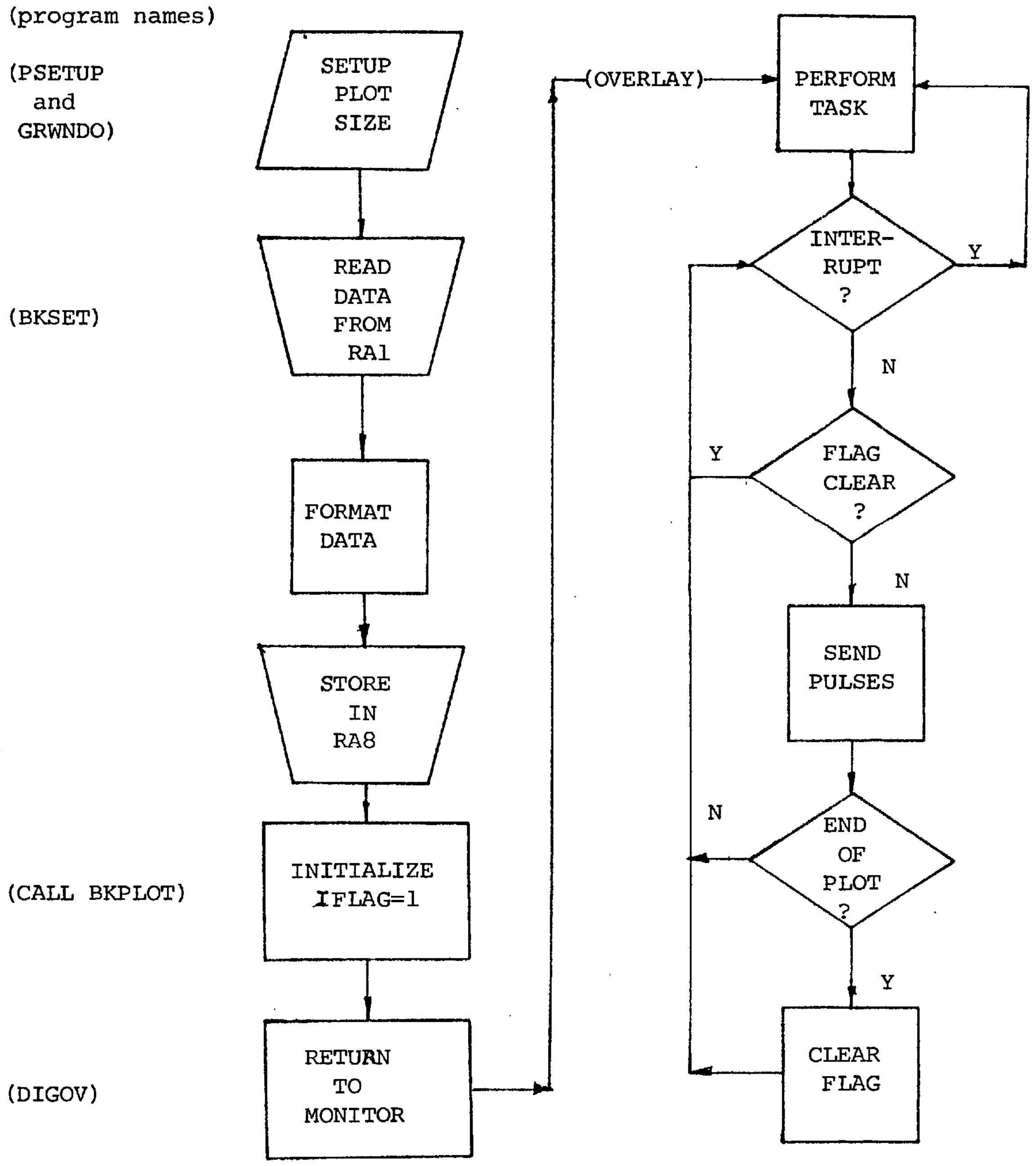
CALL INT - send off pulses corresponding to 1/40mm.

The plotting speed cannot be directly controlled by the user. It depends on user-machine interaction. 'INT' is called by GETINT, therefore, plotting is done at maximum rate until an interrupt is caused by user on a pen button. The plotter is actually in full operation, but can be interrupted at any time by task requesting full attention, e.g. to perform mathematical transformations.

H.3.4 Linking of Programs

```
BASYS, BASYS.STB < BASYS, BKPLOT, CADMAC/L, FORLIB/L/T:37776/U/E
BKSET < BASYS.STB, BKSET, PLWRIT, CADMAC/L, FORLIB/L/B:40000/U/E
```

H.4 OPERATING PROCEDURES



NOTES:

Do not call BKSET until plotting is complete or first plot will be lost.

When in DIGOV, leave bug on menu to speed up plotting.

Plotter may be set up at any time for next plot.

Plotting may be terminated by setting switch register to zero.

The origin will not be lost. Settings on SW do not affect plotting speed.

Do not use foreground plotting when in background mode or unexpected results will occur. Plots will become cumulative.

APPENDIX I

BATCH MODE OPERATION UNDER THE VAL SYSTEM

I.1 INTRODUCTION

This appendix describes the basic modifications required on the VAL system to enable the batching of operation commands with the present interactive mode.

The system will have a memory facility to enable a sequence of operations performed by the operator to be stored and subsequently processed.

I.2 MEMORY AND COMMAND STRUCTURE

The memory consists of a permanent file on disc containing the commands and the respective operating parameters.

The string of coded commands and parameters constitute a batch stream and are stored in a Batch Stream File (BSF). The BSF will contain data in ASCII form and it is possible to directly edit the batch stream.

The BSF is made up of records of fixed length each corresponding to one instruction. The record must be long enough to contain the maximum number of parameters likely to be encountered. A twelve variable array should suffice (Fig I-1). A COMMON/BATCH/PAR(10) is allocated in the resident core area to load/unload the parameters. This will be the communications area used by the overlays manager OVLINK.

I.3 OPERATION OF THE SYSTEM

The system can operate on two modes: batch or on-line (interactive), determined by a flag, BM being .TRUE. or .FALSE. respectively. The flag should be set before any operating procedure is commenced.

The system works on the fact that subroutine OVLINK is evoked each time a new program is loaded in core. Upon entry OVLINK checks whether the system is set in batch or on-line mode by testing BM. If it is in batch mode it searches for the next overlaying instruction (OVNO and ISUB) in the BSF and loads the corresponding parameters in COMMON/BATCH/. If on line mode then it stores the previous overlay parameters in the BSF and loads the next overlay from the stack into core. Then it opens a new record in the BSF for the newly executed overlaying instruction. The BSF store is loaded in a staggered fashion as shown in Fig I-2.

The format of each overlay command is determined by the requirements of the individual overlay, e.g. spin

COMMON/BATCH/XC, YC, NP, REV, ZOOM

At the end of the batch operation command will be automatically returned to the user since the last entry in the BSF is always DIGOV. BM will be reset and the system back to interactive mode.

I.4 IMPLEMENTATION EFFORT

The modifications affect mainly OVLINK which must test for the value of BM. The COMMON/BATCH/ block must be added to the resident portion of the system. The overlays will have to be modified to store the parameters in COMMON/BATCH/ and to operate in both modes. This can either be a mode test or a program split into two operation areas. One for batch one for on-line. GETINT must be disabled when batch mode is in operation.

DISALL and PLOTSC (PLOT) should preferably be modified to enabl- option between output to table or screen to be set in advance.

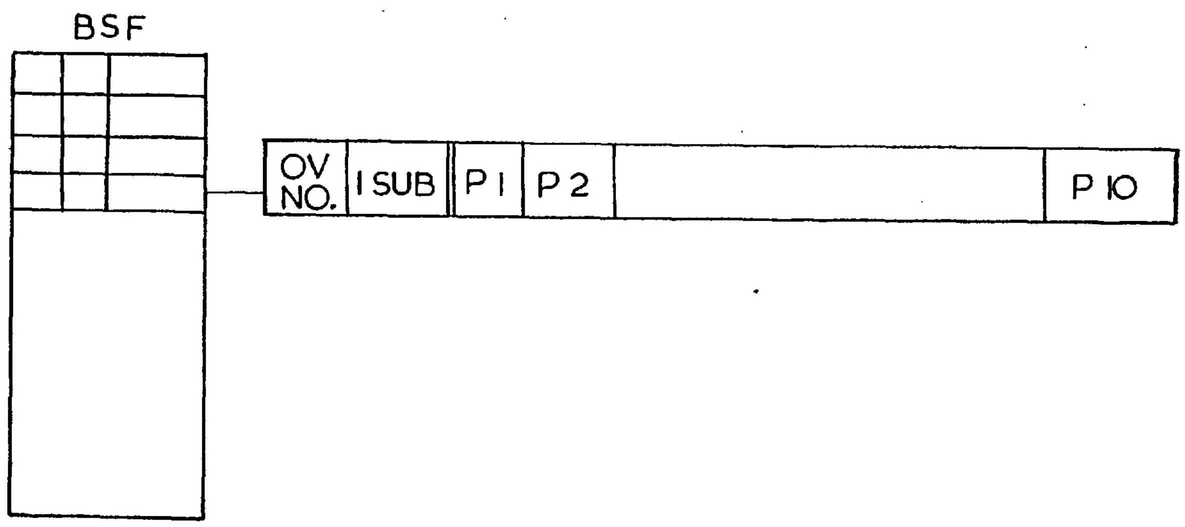


Fig I-1 BSF command record structure

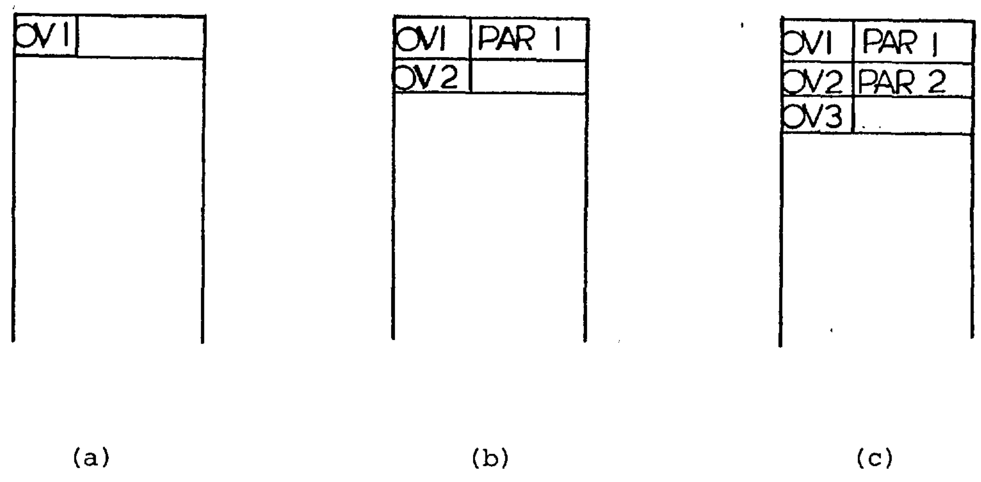


Fig I-2 Loading sequence in the BSF

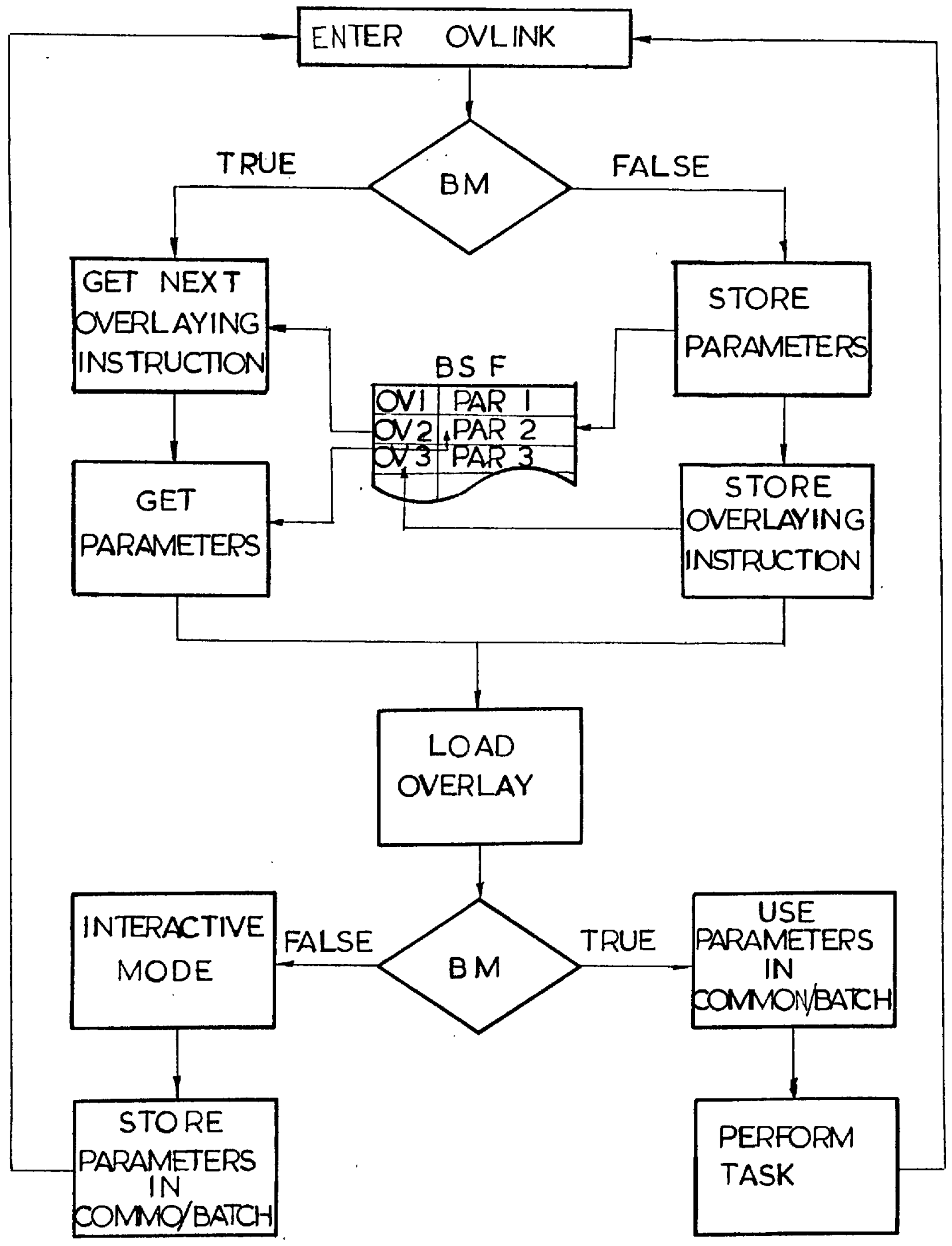


Fig I-3 Flow chart of Batch/on-line operation

APPENDIX J

OPERATING PROCEDURES

This appendix describes the operating procedures for the GAGS system. It is illustrated by two examples extracted from a film produced on the system. Although the examples refer specifically to the facilities in GAGS-2, they are also relevant to GAGS-3, because the two systems contain many features in common.

J.1 System Initialisation

To initialise the system the DEC's Disk Operation System (DOS) command structure is used (32). The system operates from a removable cartridge running on a DEC RK-05 disk drive. The system can operate on either a single or double disk configuration. The two disk configuration uses the top drive for system programs and the bottom one for data storage. On the single drive configuration, the top disk contains both programs and data, so the amount of store allocated is more limited.

In the description of the operations the following conventions have been adopted:

Boxed letters correspond to menu commands.

Bm corresponds to the operation of button number m.

Messages listed on the keyboard are underlined.

Messages displayed on the screen are within single quotes.

To start up the system the DOS monitor must first be loaded in core. This is normally done by using a hardware bootstrap. When the monitor is loaded into core it identifies itself by printing on the keyboard:

```
DOS V08-02
$
```

The monitor is now ready to accept a DOS command. The $ sign indicates that the monitor is awaiting user action. The system operates under a User Identification Number (UIC) of [2, 2]. To gain access to it, the user must log in under that number:

```
$ LO 2, 2
```

and the system replies by printing the date and the time:

```
DATE: - 01-JAN-72
TIME: - 00:00:05
```

The date is the date when the system was created.

The system is started by loading the program into core and running it. The instruction is:

```
$ RU GAGS
```

When ready the system replies on the screen with:

```
'ZERO TABLE'.
```

The user must respond by digitising the lower left-hand corner of the menu. After the table is zeroed the following message is displayed:

```
'B1 - DEFAULT'
'B2 - SPECIFY'.
```

If B1 is pressed the working parameters are set to their default values, otherwise they must be set individually:

```
'SET INPUT ORIGIN'
'DIGITISE HORIZONTAL LINE FOR SKEW CONTROL'
'GRID FACTOR' (0)
'INPUT SCALE' (1)
'OUTPUT SCALE' (1)
```

'ALPHANUMERIC SIZE' (3)

'FIELD TYPE' (1 - ACADEMY)

'FIELD SIZE' (12)

The default values are in parentheses. The default origin is on the bottom right-hand corner of the menu and the default skew control line is horizontal.

The system is now ready for data input or to execute the commands on the menu.

J.2 Preliminary Stages

The production of a film can be broken down in a number of stages. The preliminary stages, before the system is actually used are:

(a) Storyboard (script) - The first few weeks on the production are normally spent planning and discussing the storyboard which contains the basic key frames and the commentary or dialogue. The particular film used in this example, 'Rings and Travellers', is a promotional film showing the behaviour of a traveller, used in the spinning of yarns, under different loading conditions, and the advantages of using different designs for different applications.

(b) Bar-sheet - The next stage consists in breaking down the sound-track of each sequence so that accurate timing and sound synchronization, if required, can be obtained. The timing can be defined in terms of the number of frames between two key frames.

(c) Artwork - The key frames are supplied as accurate drawings of the objects to be animated: rings, travellers, logos, letters and other sketches. For this particular problem the accurate results, from different load condition computations, are also provided.

Two sequences from the film are described in detail, illustrating typical sequences produced on the system. Firstly the title sequence, then the stressed traveller sequence.

J.3 Example 1

Computer animation has always been particularly powerful in producing interesting title sequences for films. For this particular sequence only two drawings are required: The company logo (the lion) and the outline of the letters. (Fig J-1, a, d). The drawings are accurately digitised by tracing over the artwork. The curved sections on the drawing can be input by digitising discrete points with B1 (lines are broken with B2), or by using continuous digitising or curve fit. These programs are evoked by digitising the corresponding menu square:

$\boxed{\text{CONDIG}}$

'CONTINUOUS - DIGITISER'

'B1 START - BREAK'

'B4 TIME'

'B5 DISTANCE'

'B8 EXIT'

Distance mode is set by default, so only B1 is pressed to start and break lines. To exit B8 is pressed.

$\boxed{\text{CURFIT}}$

'CURVE FIT WORKSPACE'

'B1 DISPLAY ONLY'

'B2 STORE'

'B8 EXIT'

In this case B2 is pressed because the cubic spline is required as part of the data. The data in workspace consisting of long straight lines is overwritten by the curve.

The letters are provided in this case, but they could be generated by the lettering facility in the system.

$\boxed{\text{LETRA}}$ 'DIGITISE ORIGIN'

A point corresponding to the bottom left-hand corner of the first character must be digitised

<u>SIZE MM, SPACING</u>

_ _ 6 0 . 0 0 , 0 . 5 _

<u>INPUT CHARACTER STRING</u>

EADIE $

The size of the letters is 60mm and the spacing is equal to half the width of the character. The $ sign indicates the end of the input for the character string.

Once the artwork has been input the drawings should be saved in permanent files. The files used are 1 and 2 for the logo and the letters respectively.

$\boxed{\text{WORK- SPACE TO FILE}}$ $\boxed{1}$ for logo

Digitise letters and repeat for file number 2.

To perform the transformation as depicted through frames a to d in Fig J-1, in-betweening is used.

IN BETWEEN

'IN-BETWEENING'

'SELECT FIRST FRAME'

'B1 SELECT NEW FILES'

'B2 USE OLD FILES'

'B8 EXIT'

Since the files have not been set in advance, B1 is used. The first file is 1, therefore digitise square corresponding to file number 1

1

with B1.

2

'SELECT 2ND FILE'

digitise second file.

'NO. OF FRAMES' 36

The sequence lasts 3 seconds so 72 final frames on film are required. By photographing on twos only 36 display frames are required. The parameter is entered via BUTNUM (see Appendix A, section A.3).

'FAIR-IN FRAMES' 12

'FAIR-OUT FRAMES' 6

The initial ease-in is slower than the final ease-out. The result is required directly on 16mm film so the film is exposed by photographing directly from the screen. As each frame is generated and displayed two frames on the camera are exposed. The program is controlled by B1 to erase the screen between the frames.

The letters are required to be blocked in uniformly (Fig J-1, e). The block-filing program is used to shade the inside of the letters.

BLOCK FILL	'BLOCK-FILL'
	'B1 DISPLAY ONLY'
	'B2 STORE WITH OUTLINE'
	'B3 STORE WITHOUT OUTLINE'
	'B8 EXIT'

Since the outlines of the letters are still required B2 is pressed.

'I SPACING' 2

minimum scanning gap.

'ANGLE' 90

for vertical scanning.

J.4 Example 2

The second sequence consists of showing the effect of three external forces acting on an elliptic traveller (Fig J-2). The forces are the yarn tension, the centrifugal force and the normal reaction. These three forces should balance horizontally under optimum operating conditions. The reaction of the traveller against the ring provides the necessary friction to ensure that the yarn is properly tensioned.

For this sequence four drawings, as shown in Fig J-2, are required, giving the extreme positions and two intermediate key frames. In this particular case the accurate drawings for the different positions of the traveller, as it is loaded, are provided by the designer. This is the typical case, where the design solution obtained by different means is fed as data into the system.

The drawings are input by a method similar to that described in section J.3 and stored in files 35, 36, 37, 38. The yarn is popped-on the first frame. It is stored separately in file 39. The alternately shaded zones are obtained by block-filling the areas. The moving yarn is obtained by storing 4 different yarn positions in files 39-42 and filmed alternately.

The sequence starts with a hold for two seconds so both file 35 and 39 are displayed

$\boxed{\text{FILE TO WORKSPACE}}$ $\boxed{35}$ $\boxed{39}$ digitise in order

Both files are now in workspace so they may be appropriately scaled to the right window size. 48 frames are exposed to last 2 seconds on the final film.

The remaining sequences are obtained by in-betweening:

$\boxed{\text{IN BETWEEN}}$ same as above

$\boxed{35}$ $\boxed{36}$

'NO. OF FRAMES'	36
'FAIR-IN FRAMES'	20
'FAIR-OUT FRAMES'	4

to be shot on twos to last 3 seconds on the screen.

The process is repeated with files 37 and 38.

Although the process is not based on the actual parameters giving the accurate intermediate positions of the traveller, but rather obtained by interpolating between the key positions, the results are accurate enough for illustration purposes. The key frames can be regarded as numerical solutions at specific points, which are then joined by straight lines to give the complete curve. The results are obviously more accurate as the number of points used in the solution are increased. For this particular case four were found to be sufficient.

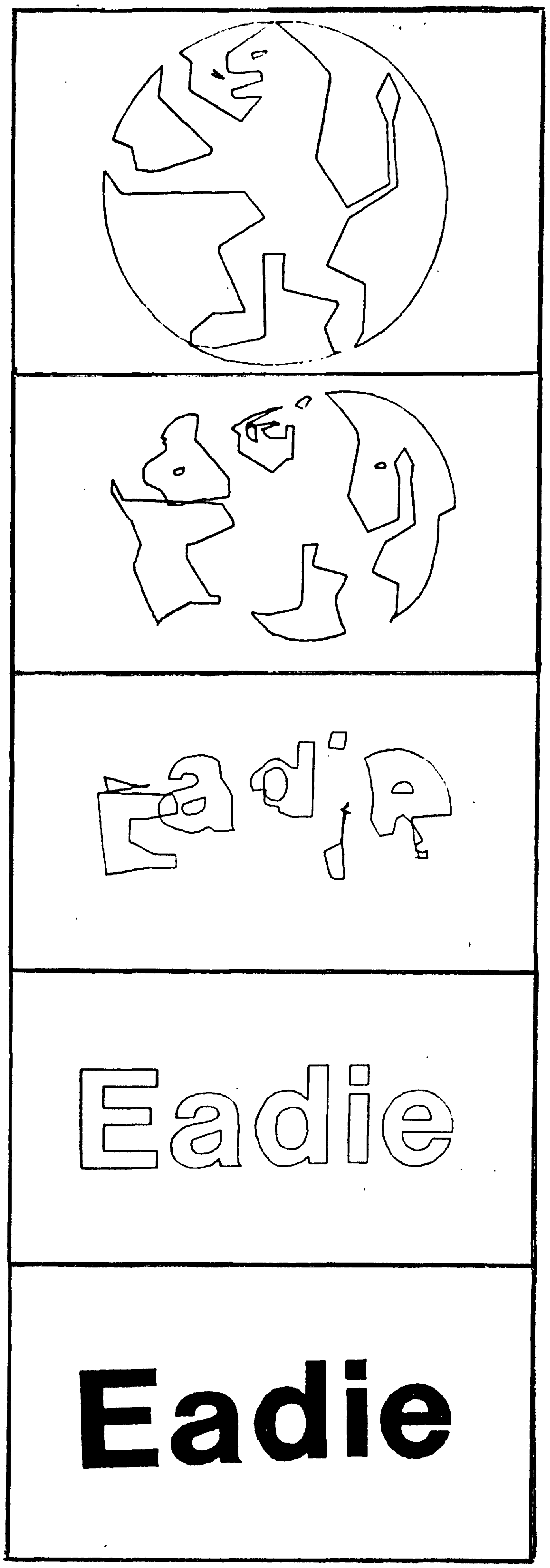


Fig J-1   Title Sequence

Photographs reproduced with kind permission
of Eadie Bros and Co. Ltd.

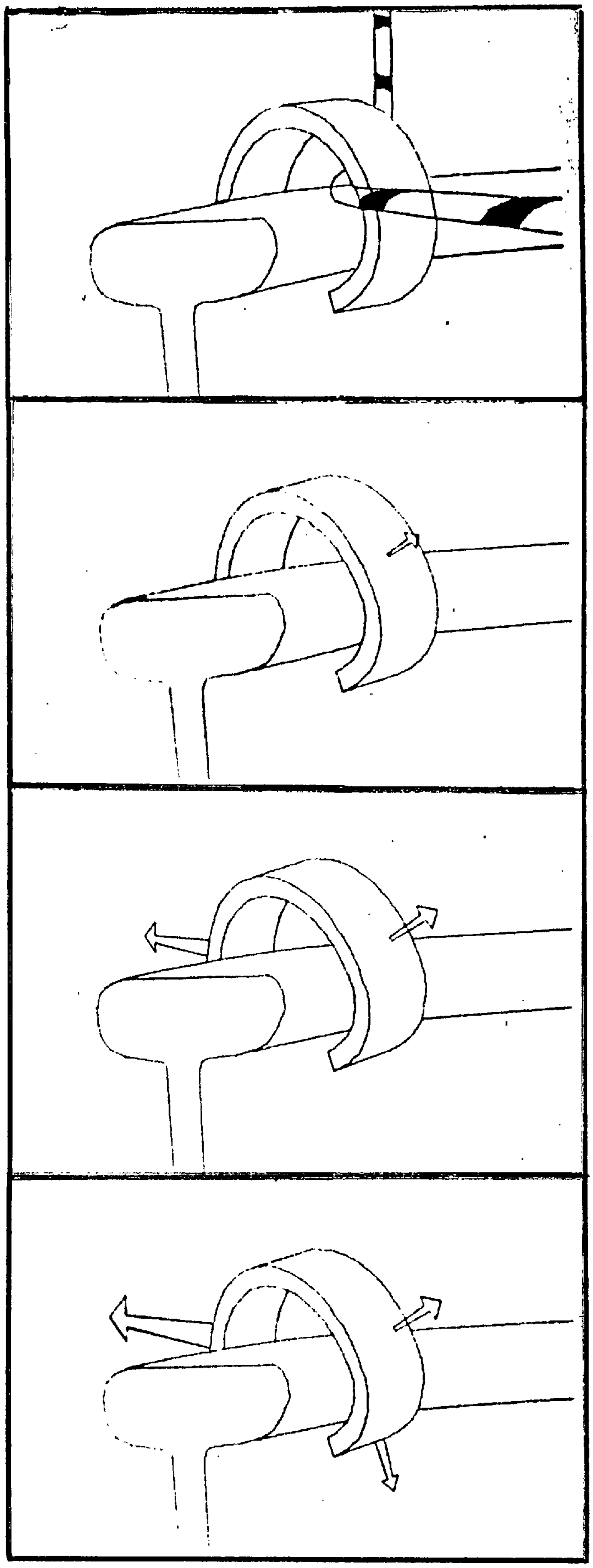


Fig J-2 Load Sequence

Photographs reproduced with kind permission of Eadie Bros and Co. Ltd.

Three-dimensional digitizing techniques for computer-aided animation

C. Yi, C.B. Besant and A. Jebb

The Video Animation Computer-Aided Animation System consists of a digitizer, plotter and storage tube display configured around a minicomputer.[1] The system is fully interactive allowing the user to create shapes and manipulate them in a variety of ways in order to produce the required movement for an animated sequence.

Much of the early animation work was concerned with the movement of shapes which were described in two dimensions.[2] However, since most objects are three dimensional, it is clearly desirable to have a three dimensional representation within the computer and animate in three dimensions.

The work described shows that, using a conventional digitizer, three dimensioanl data can be input to the computer. Three different methods have been developed and are discussed. The first method is based on digitizing two or more orthogonal projections which are subsequently assembled within the computer to give the required three dimensional data.

The second method consists of digitizing contours in two dimensions with the third dimension normal to the surface, being added numerically. A surface patch technique is then utilized to interpolate the contour data in order to produce data of sufficient quality for graphical presentation.

The third method is based on using photogrammetry techniques where two perspective drawings or photographs can be digitized and the data converted into a three dimensional model.

Orthographic representation of solids

There are several ways in which solids can be represented. A number of pictorial views of a solid in different positions will be required for complicated objects, if the shape is to be fully understood. Orthogonal projections are a well-established method of drawing solids. With this method, the length, breadth and depth of a solid are fully represented by at least two views which are plane areas. In practice, three or four main views will usually fully represent the most complicated solid.

Mechanical Engineering Department, Imperial College of Science and Technology, Exhibition Road, London SW7 2BX.

Since many of the objects we are likely to draw will require some sort of orthographic representation it is useful to provide a means of input for such drawings. Because of the input facilities available, many of the drawings need not be as accurate as for engineering drawing purposes. Often a rough sketch is enough as a source of input.

Either first or third angle projection drawings can be used, each with its advantages and disadvantages. Only the planes to the right of the basic X–Y front elevation are considered.

Third angle projection has the advantage of having the front elevation on the bottom left-hand corner which is the most accessible area on the digitizing table. This is important because the front elevation is in most cases the most representative view of the solid. Also two dimensional data can be positioned and digitized without the need to

First angle projection (Figure 1) offers a more natural layout of the views which is easier to visualize. Its main drawback is the high position on the table of the front elevation (Figure 1, view A). For low positioned two dimensional data, the y datum needs to be redefined. The sign of z is negative in the plan view. This can be disconcerting to someone not familiar with the system.

The most natural position for the drawings is a combination of first angle and third angle projection systems (Figure 2). This is achieved by exchanging the position of the front elevation and the plan. This ensures an X-Y plane on the bottom left hand corner of the table and there is no inversion in sign except in plane D but this is a rarely used plane.

To input data from an orthogonal projection, the table is positioned into two or more areas each corresponding to an X-Y, Y-Z or Z-X plans. Up to six elevations may be considered, but normally only three will be available, and in most cases, two will suffice.

When the system is operating in the 3-D mode, the cursor is located in the appropriate plane and the coordinates on the table are determined and converted automatically.

A point known as the *x-y* datum (*ZX, ZY*), (see Figure 2), must be defined prior to any data input. This point corresponds to the common vertex of the planes. It is defaulted at the initializing stage, but can be redefined at any time.

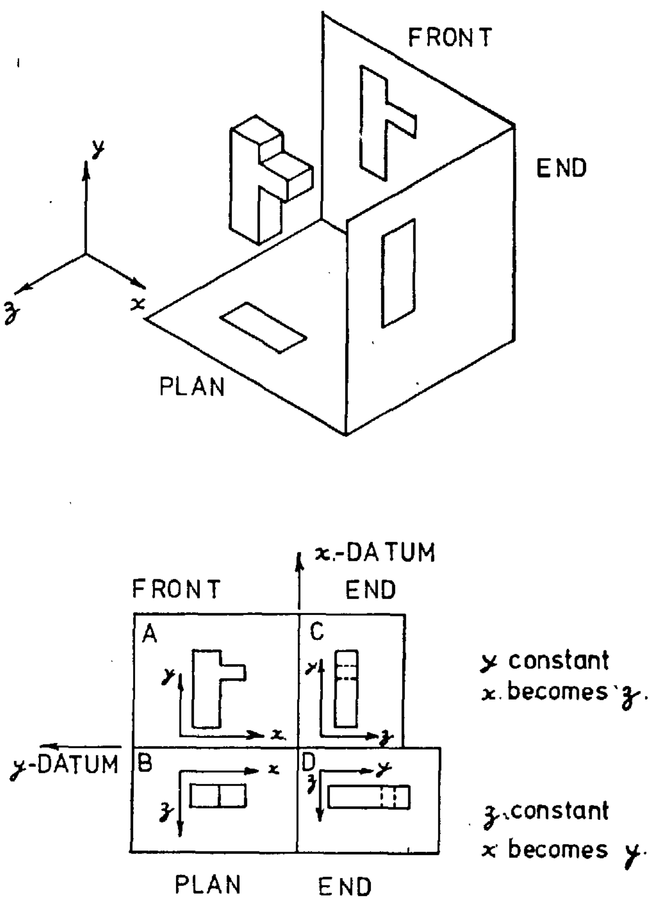


FIGURE 1. First angle projection

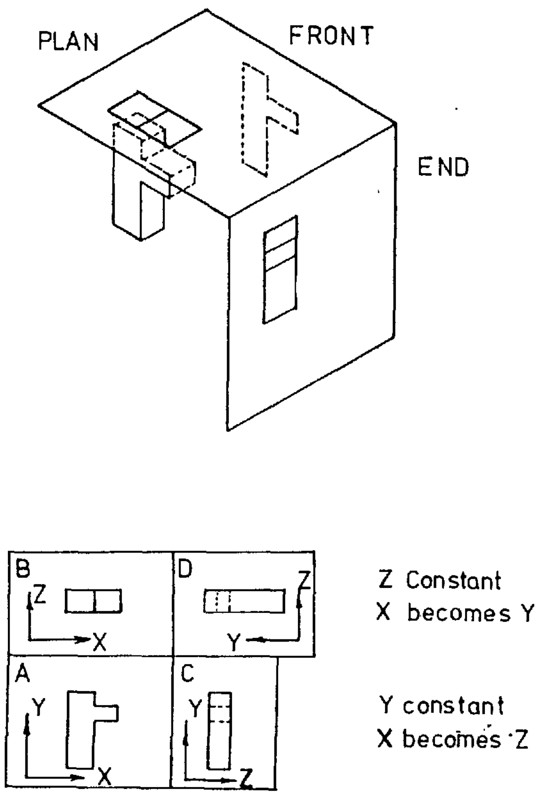


FIGURE 2. Modified first angle projection

The cursor reads data in two dimensional x-y coordinates so all the standard input facilities described by Besant et al[2] are operative, although they will lie in different spatial planes. The control mode facility is particularly useful as it allows the same point to be accurately positioned on different planes. This is the recommended mode of input when input requires many changes from plane to plane.

The tests in the input program used to determine the spatial coordinates are very simple. The x-y datum determines the plane since it defines the intersection of any two planes.

Let (x, y) be a point on the table and (X, Y, Z) the corresponding three dimensional coordinates of the point.

For $x < ZX$ and $y < ZY$ the point lies in the X-Y plane so

$$X = x$$

$$Y = y$$

If y is above the y datum (ZY), then Z is always given by $y - ZY$.

x remains unchanged until $x > x$ datum (ZX) when for

$$y < ZY \qquad Z = x - ZX$$

$$y > ZY \qquad Y \propto ZX - x$$

To input the first point or when a line is broken to change planes, two points on two different views are required to define the point in space. The first point sets the independent coordinate and the second determines the other two coordinates. From two points on the table four coordinates are always obtained. For the repeated coordinates the second value overwrites the first one.

The analysis only deals with the modified first angle projection, but similar results can be obtained by following the same reasoning with any other form of projection.

The system can be easily converted to accept data in any format.

Display file

For display purposes the data digitized in the three dimensional mode is arranged in three separate files, FILE 1, FILE 2, FILE 3.

FILE 1 – containes the actual three dimensional data consisting of vectors describing the object. The file consists of a sequence of (I, X, Y, Z) records. I qualifies the data contained in X, Y, Z. The coordinates are relative to an absolute origin. This data cannot be viewed directly. It exists for storage and data manipulation purposes only.

FILE 2 – This file containes the projected view of FILE 1. The data structure is identical to FILE 1, in the same sequence but with X, Y in terms of screen coordinates. Z contains the depth parallel to the line of sight. It is used in hidden-line tests to determine visibility of lines and illumination of surfaces. The same sequence enables editing by a parallel search address method, of the three dimensional data. FILE 1 is converted into FILE 2 by a suitable routine. This routine may be enabled at the input stage with FILE 2 being created at the same time as FILE 1, or at some later stage with revised parameters. The existence of this file should be transparent to the user.

FILE 3 – This file contains the original blueprint of the digitized artwork. It is used for checking data when ambiguities in three dimensional viewing arises. Data is converted from FILE 3 to FILE 1 by an input routine. This can be invoked at a later stage but care must be taken to ensure the same parameters are used. It usually becomes obsolete once the data has been inspected and corrected. This file is made up of (*I, X, Y*) records only.

Display modes

In the same way as there are three input files there are also three modes of digitizing and display. This is determined by a switch ND which can be set by the user.

ND = 0	No display (blank screen).
ND = 1, 2	3-D viewing with current parameters. The cursor on the screen moves in three dimensions.
ND = 3	Display blueprint – cursor in two dimensions. (Special input facilities can be used to advantage.)

Usually all three files are created simultaneously irrespective of the display mode. This is done at the input stage as it will represent a very small time load. As new data is created or old data modified, dependent data is automatically updated. This data structure technique links the displayed points and lines with their exact dimensional equivalent.

On our system, the views are presented alternatively on the screen. Ideally all views should be shown simultaneously, with the three dimensional picture made rotatable by a joystick or a similar device. (In the system a special menu patch is used instead of a joystick to obtain a similar effect). This, however, results in clustering of data due to the size of the screen, making details hard to visualize. It was therefore decided to display different views individually and improve the display speed. The use of FILE 2 greatly reduces the display time of perspective views.

Contouring

This technique requires the data to be in the form of contour maps. It uses several of the facilities already incorporated in the system:

An inherent *Z* level in the data-structure
The griding facilitiy at the digitizing stage
The continuous digitizing mode.

Before digitizing the contours, the level for each contour line must be normalized to a maximum of 60, the system allowing 60 levels to be used. This level value is defined by the *I* parameter in the data record. Prints of the same level will have the same value in the parameters defined by *I*.

The contours are digitized by just following the contour with the digitizing pen. Lines are started and broken by pressing two buttons on the digitizing pen.

The data generated by this method appears two dimensional, and is viewed in two dimensions.

A program can then be called to compute the three dimensional surface data. The program computes gridded data from the randomly spaced input data. The input data is sorted and made to lie on the nearest grid node. The size and the mesh of the grid is defined by the user. The program automatically adjusts it to envelope the whole of the data. Superimposed grid points are progressively averaged. Undefined grid points are linearly interpolated from the set of points. The interpolation is first performed round the edge of the mesh in a circular loop; then horizontally with points at different levels only and then vertically for the remaining void nodes.

Thus we have a surface defined by grid points on a $n \times m$ matrix with the *Z* values known. This surface can be viewed in several ways. The one we use is oblique projection with hidden line removal. The viewing parameters are defined by the user.

Photogrammetry

This technique was originally developed for shipbuilding design by Smith[3] but can be applied to any model which can be photographed unobstructively.

Photogrammetry consists of retrieving dimensional data from photographs. The use of a pair of stereo photographs allow measurements to be taken both in the plane of the photograph and in depth. Accuracy can be very high with good quality and well set up equipment, but this often shows in the high capital cost of the equipment. For applications which do not require a high level of accuracy, a simpler form of photogrammetry can be used.

The stereo photographs are taken from two cameras rigidly mounted on a strong frame. A single camera on a rack can also be used instead. Such an arrangement is shown in Figure 3. The method described here is known as comparative photogrammetry, and neither the exact position of the camera nor the focal length of the lens need by known. It gives an accuracy within 3% which is satisfactory for many applications. Two scales are required at the front and back of the model. It is convenient to have scales of equal length although this is not a necessity. The distance *Z* between the scales must be accurately known though.

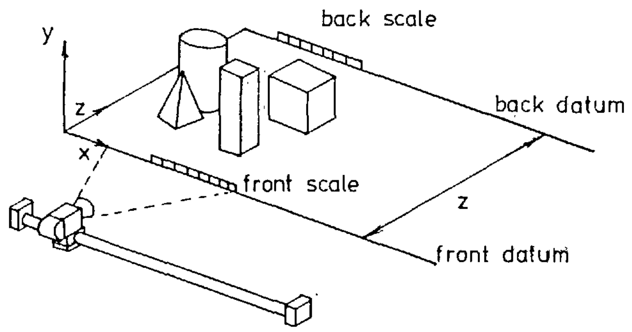


FIGURE 3. Photogrammetric camera and model arrangement.

Consider a model with two scales of equal length AB and CD placed at distance Z apart at the front and back of the model being photographed (refer to Figure 4).

The coordinate *z* of a point on a line *PQ* is given by

$$z = \frac{Z}{a-1} \frac{(\delta c - \delta p)}{\delta p}$$

where *a* is given by C_1D_1/A_1B_1 which can be measured from the photograph.

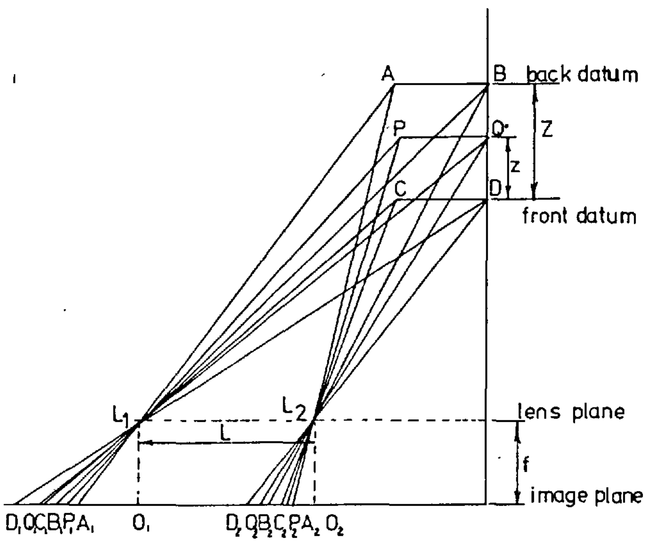


FIGURE 4. Light rays geometry for comparative photogrammetry

Note $A_1B_1 = A_2B_2$

$C_1D_1 = C_2D_2$

so the average can be taken.

c and p are the comparator readings for points C and P given by $(O_1C_1 - O_2C_2)$ and $(O_1P_1 - O_2P_2)$ respectively.

These are the coordinates read in by the computer from the two photographs. The points are digitized alternatively from the photographs.

Points O_1 and O_2 are the optical centres of the picture and must be determined in advance by the use of a baseboard grid placed in the middle of the model or by reference points on the model.

The theory makes the assumption of thin-lenses so that the light rays through the centre of the lenses L_1 and L_2 are not deflected.

Distances along the X and Y axes on the photograph will require correction for photograph scale. This is given by a scale F.

$$F = \frac{\text{object size}}{\text{image size}}$$

For a known object like PQ, F is given by

$$F = \frac{PQ}{C_1D_1}\left(\frac{z}{Z}(a-1) + 1\right)$$

i.e. F is proportional to z.

$$\therefore X_0 = F \times Xi$$

where X_0 and X_i are respectively the distances with respect to the same datum on the model and the photograph.

f, the focal length and L, the distance between camera position do not appear in any of the equations. Thus, the analysis is independent of focal length and camera position.

Conclusion

The paper describes three different input techniques to interpret three dimensional data from two dimensional representations. No doubt other techniques can and have been devised and used successfully. The techniques described are however very simple to use, and do not require any special piece of hardware. They use data sources that are commonly available and require little special preparation. Photogrammetry proved to be the most difficult to use due to the difficulties encountered by using photography, and has the disadvantage of not being applicable to conceptual models.

With the increasing importance of three dimensional graphics, it becomes crucial that simple and easy to use input data techniques are developed. The inclusion of special hardware should greatly simplify this task and would allow the input to be obtained directly from solid objects, thus precluding even the need to prepare drawings of the data. The techniques described will, however, still be required when analysing data which is not physically accessible.

Acknowledgements

The authors gratefully acknowledge the financial support from Video Animation Ltd. for this work. In particular, thanks are due to Mr A.R. Diment and Mr P. Chandler for their suggestions and advice throughout this work.

References

1. Yi, C., Besant, C.B., Jebb, A., Diment, A.R., and Hayward, S. (Computer animation'. Proceedings of an International Conference on Computers in Engineering (CAD '74). Imperial College, London, (25–27 September, 1974).

2. Besant, C.B., Diment, A.R., Hayward, S., Jebb, A., and Hawkes, P.L. 'Computer-aided animation.' Bulletin of the National Research and Development Council. (Spring 1975). No. 42.

3. Smith, W.G., Photogrammetry as an Aid to Manufacture of Ship Piping. BSRA Report NS 306, 1971.