MD-14 | 157

MILITARY TECHNICAL COLLEGE

CAIRO - EGYPT

A DATA STRUCTURE FOR PROCESSING THREE-DIMENSIONAL ENGINEERING DRAWINGS IN A SMALL GRAPHIC SYSTEM

M.ZAKI*, F.RASHED**

ABSTRACT

The use of computers for processing three-dimensional (3D) engineering drawings has been investigated in this paper. The essential properties of a hierarchical data structure are pointed out and employed to provide a natural computerized data structure for representing the geometry of the drawn part. The intrinsic flexibility of that structure makes it possible to keep track of a full 3D description of the part under consideration.

Although a hierarchical data structure is not the neatest structure for expressing geometrical properties, it may form a convenient framework for data processing. It also has the advantage of being embedded in high-level languages. In the system presented here auxiliary light buttons are introduced to facilitate unambiguous stepping up or down through the structure hierarchy. Consequently the system can adequately transfer program control to a particular part of the graphic system to perform the exact specific tasks.

The given system handles solid objects with plane surfaces and can be extended for processing cylindrical surfaces when they are represented by their canonical parameters. Such a system will be economical and quite convenient for drawings that are widely used and subjected to frequent minor changes.

* Associate Professor, Dept of Computers and Systems Engineering, Al Azhar University, Cairo, Egypt. ** Assistant Professor, Dept of Production Engineering, Helwan University, Cairo, Egypt.

1. INTRODUCTION

The application of computer displays for processing engineering drawings has been considered and carried out by many practical systems (1, 2). However, in these systems the resolution of the display screen is limited. Such a disadvantage can be offset to some extent by data structure techniques which link the displayed points and lines with their exact dimensional values instead of their truncated screen coordinate values. The essential properties of these structures 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 intrinsic flexibility of these systems is such that, with a slight increase in complexity, it is possible to keep track of a full 3-dimensional description of the drawn part and this in turn could form the basis of a comprehensive data base for design analysis problems concerned with 3D geometries (3).

The purpose of this paper is to point out an evaluation for such structures in graphic processing of 3D engineering drawings. One significant advantage is the way of generating powerful algorithms to update automatically dependent data items that otherwise would have to be handled individually by a time-consuming procedure. Another important attribute (4) is the way structures can be used to partially overcome the difficulties of working in 3D while the display screen itself is 2D. The technique used to solve this problem consists of performing rotation, shift and scaling operations on the 3D data base until the region under consideration is projected. In this situation, knowing the new set of axes, the usual light pen constructions can be reconverted automatically into the 3D reference axes of the data base.

In the given processing system, the characteristics of computer displays have been taken into consideration, namely, the use of combined sequences of light pen and keyboard operations. However, the speedy response of the CRT display makes it ideal for the role of processing engineering drawings where a designer can interactively update and regenerate his drawings.

2. GEOMETRY DATA BASE

A particular structure for representing the geometry of a solid object (5) consists broadly of the hierarchy of data items outlined in Fig.1. In this representation the object is defined in terms of its surrounding surfaces. These surfaces are defined by their boundary lines which in turn have end points. Such a framework can equally well describe curved surfaces and their curvilinear intersections although, naturally, a different set of data values would be needed. Physically, the properties of the object can be entirely determined from the canonical parameters of the bounding surfaces, however, since graphic processing in this system is expressed mainly in terms of lines and points, then it is more

convenient to omit the parametric form and represent the shape explicitly by expanding the canonical form for its surfaces entirely in terms of the end points of lines. Thus, the bulk of the structure represents the topology of the object including the relationships and interdependencies between surfaces, lines and points in a form amenable to subsequent displaying operations.

In contrast, a conventional approach would be needed to manipulate arbitrary curved surfaces (6, 7). For these it is better to revert to the parametric representation and solve the surface intersections as required. In any case, language statements are more convenient editing/displaying techniques for curved surfaces when compared with point/line displaying procedures described in the later sections. These are only applicable to plane surfaces.

3. GENERATION OF DISPLAY DATA

The necessary hardware is relatively unsophisticated. It is required that the sequential file of display commands can be repeatedly transmitted to refresh and maintain the displayed picture. Essentially, therefore, the display sequence is different from the hierarchical geometry data structure described above. In order that the dimensional data may be systematically converted into suitable display coordinates, a display list is maintained within the structure to link together in correct sequence the blocks of line data with their corresponding display commands, Fig 2. Because such a display list is itself a type of data structure, it can be readily updated when lines are added or removed, however,no such intersections are possible in the sequential display command file which has to be regenerated by a display algorithm. The basic mechanism consists of stepping through the display list checking to see if adjacent line segments can be strung together, otherwise each line needs an extra command to specify its start point.

Ultimately the 3D point/line data are projected into a set of 2D screen coordinates but because the flexibility of such 3D package depends on repeatedly transforming (rotating and shifting) the 2D view of the displayed object, it is convenient to form an intermediate file of 3D data in display sequence, Fig 2. During any transformation operation the 3D data file is unchanged. Normally such graphic operations would have to be written in a low-level language.

4. PROCESSING LIGHT PEN INPUT DATA

Ideally in a graphic system, the light pen should explicitly identify the data structure block corresponding to the item the pen has detected (8). However, the display hardware does not possess such a feature. Instead, the data item that accompanies the light pen interrupt merely identifies the

display command that the light pen has detected by its relative position (word-count) to a central word (end of frame) within the display file, Fig 2. In such a system the display command is of little value since its screen coordinates are truncated projected values of the 3D data they represent. What is needed is a link to interrelate the screen data with the data in the geometry structure. One possibility is to include in each data block its corresponding word-count and then search the whole structure to find the item that matches the light pen word-count. Since lines are the only detectabke screen items, a faster technique consists of searching the display list since this consists of line blocks linked together in the same order, ie in the same word-count sequence, as their corresponding display commands. The increasing order of word-count of such a list virtually halves the search time. On a larger drawing, consisting of many objects, the search time can be significantly reduced by first checking each object to determine which one has the word count range which the light pen has detected.

Ultimately, since lines are the only light pen sensitive items on the screen, the light pen interrupt is converted into a pointer to a line block within the data structure. However, in many instances, the user may wish to refer to one of the end-points of the lines or the surfaces intersecting at a certain line. In this case a certain ambiguity may exist in stepping up or down through the structure hierarchy. Therefore, the processing algorithm may need supplementary information which the user of this system supplies through auxiliary light buttons.

5. AUXILIARY LIGHT BUTTONS

Normally a panel of function keys would be an appropriate method of qualifying light pen actions to the graphic system. However, when the graphics terminal does not have this facility one can rely on certain auxiliary light buttons to behave in a similar fachion. For convenience, since they qualify the light pen actions, these buttons appear in the vicinity of whatever screen item has been detected by the light pen to provide the user with features not otherwise available in the hardware. These are the option when the pen detects a line on the screen of selecting one of three possibilities. These possibilities are the end points of the line, a point on the line seen by the pen, or generating a tracking pattern to locate some other point near the line. In this way the light pen can identify piint blocks even though these do not exist as detectable screen items.

The main purpose of the auxiliary light buttons is to transfer program control to a particular part of the graphic system to perform specific tasks in shich the light pen data may have quite different meanings. Again working in 3D makes the whole problem more complex. Therefore, as an attempt to simplify matters and spare the user the burden of remembering elaborate 3D construction sequences, the system guides the user by means

of commands and messages which prompt him to perform the needed actions. The main sequences are outlined in Fig 3. To further redice the possibility of error, only auxiliary light buttons valid for a prticular operation are displayed. Noemally the choice, as outlined by the system commands, is at the most limited to three options, however, these buttons in turn may call for a series of subsidiary interactions. When a particular light button is fully processed, it generates the next combination of buttons with their particular commands and messages.

Because of this logical structure, list processing techniques can be usefully applied. In place of the usual technique where each light button does its related operations under program control, light buttons are controlled by an algorithm operating on the button data blocks, Fig 4. The flow chart for that algorithm indicates how the sequence of commands, messages and light button lists are generated.

6. CONSTRUCTION ROUTINES

Since a solid object is completely described by the parameters defining its boundary surfaces, then the lines of intersection and endpoints can be evaluated. However, in the presented system a simpler technique has been adopted to resemble conventional drafting practice. This technique consists of defining a surface by drawing out its edge lines and then joining these surfaces together to form a solid shape. The geometry structure is hierarchical from the top downwards. To make the construction rounines fit into this underlying structure before print/line items are constructed, the system asks the user to define the general topology of the object without necessarily being explicit about the coordinate values. The light button sequence, shown in Fig 3, automatically leads the user through the correct selection sequence such as DRAW, POLYFACE, RECTANGLE, LINE, POINT. In this way the system steps through the structure until it reaches the correct level where dimensional values can be inserted into data blocks.

Initially, the user would sketch out the shape in which the system automatically inserts the appropriate 3D coordinates derived by reconverting the light pen screen coordinates back in terms of the 3D data structure axes. Once the object has been sketched, the precise data values can be inserted to generate the required shape. One of the main features of the structure is to allow powerful constraint algorithms to operate and perform radical retrieval and updating procedures automatically. For example, instead of specifying a rectangular box, Fig 5a, by constructing its eight corner points (each has 3 coordinates, making 24 data values altogether), the shape may be defined by selecting the light button 'RECT BOX' and then defining the height and width of the first face by '3 POINTS'. The 'ROTATE' light button is then activated to bring an orthogonal plane into the plane of the screen so that the 'LINE' and 'POINT' constructions can be made to define the depth of the box. The system then completes the object by

defining the remaining four sides in order to satisfy the underlying constraint. Such a constraint has arisen from the fact that the faces of the box should be parallel to the current set of projected axes. More complicated shapes, Fig 5b, can be developed by the 'POLYFACE' constuction facility and its related surface definition light buttons. For each case, interactive point/line constructions can be added to a surface if and only if it lies in the plane of the screen.

7. VISUALIZATION

Although an isometric projection does give some indication of depth, there are still ambiguous views which can only be solved by perspective and brightness modulation. One immediate improvement to the visualization can be achieved by making the working plane brighter whilst the attached lines in the background are made less bright as well as insensitive to the light pen. Knowing the active plane, it is possible to show only the edges/lines attached to the plane so that the problem of hidden lines does not arise. Unwanted details, such as dimension lines, may be suppressed since they would not normally contribute to the editing process.

The drawing data structure should make provision for these data items since they would be required during the drawing regeneration stage. This is essentially a post-processing operation preparing data for an output device such as a plotter. To be realistic a full description complete with dimensions, labels and parts lists may be required together with an orthographic projection system to suit the plotter instead of the isometric system which suits the CRT display.

If the presented system were extended to handle cylindrical surfaces, these surfaces would probably be represented implicitly by their canonical parameters. In many instances the actual boundary curved lines of such surfaces are not so important, since the user is mainly interested in their corresponding radius and centre. Thus, there is no need to devise surface generating contours to show the curvature of the surface. The representation might consist of the axis of the cylindrical surface together with radius vectors to certain tangent points.

8. CONCLUSION

This work has presented an identification of a realistic mode in which a computer CRT can be employed to display 3D engineering drawings. The following remarks summarize the significant features of this work.

1. Since data preparation is likely to be an expensive overhead, computer processing of drawings can only be justified in drawings that are widely used and subjected to frequent minor modifications. Such processing can take place economically in design offices as routine tasks.
2. Because of its interactive features, the CRT display can

perform a useful role in the intermediate modifying stage rather than the input/output stages. However, these stages may be performed efficiently by using off-line devices.

3. It is essential that the 3D drawing data should be structured to permit the display to overcome its inherent limited resolution and allow information to be retrieved and updated precisely. Other advantages of structured data are: the facility to represent geometry constraints, rapid random access instead of sequential search, and a flexible data base for CAD.
4. Compared with orthographic 2D drawings, the display can give an excellent visualization of the part shape by displaying an isometric projection.

Actually a considerable programming effort is required for the implementation of the presented system, however, here the basic architecture only is investigated without pointing out the details of graphic processing.

REFERENCES

1. Newman, W.M. and Sproull, R.F., 'Principles of Interactive Computer Graphics', McGraw-Hill Book Co, New York, 1973.
2. Farrell, E.J., 'Color Display and Interactive Interpretation of 3-Dimensional Data', IBM J of Res and Develop, Vol 27, No 4, July 1983.
3. Baxter, B., '3D Viewer for Interpretation of Multiple Scan Sections', AFIPS Conf. Proceedings, Anaheim-California, May 19-22, 1980.
4. Chock, M., Cardenas, A.F. and Klinger, A. 'Manipulating Data Structures in Poctorial Information Systems', Computer, November 1981
5. Lang, C.A. and Gray, J.C., 'ASP - a Ring Implemented Associative Structure Package', CACM, August 1969.
6. Zucker, S.W., Hummel, R.A. and Rosenfeld, A., 'An Application of Relaxation Labeling to Line and Curve Enhancement', IEEE Trans. on Comp, Vol. C-26, No 4, April 1977.
7. Shapira, R. and Freeman, H., 'Computer Description of Bodies Bounded by Quadratic Surfaces from a Set of Imperfect Projections', IEEE Trans. On Comp. Vol. C-27, No 9, September 1977.
8. Newman, W.M., 'Trends in Graphic Display Design', IEEE Trans. on Comp., Vol. C-25, No 12, December 1976.

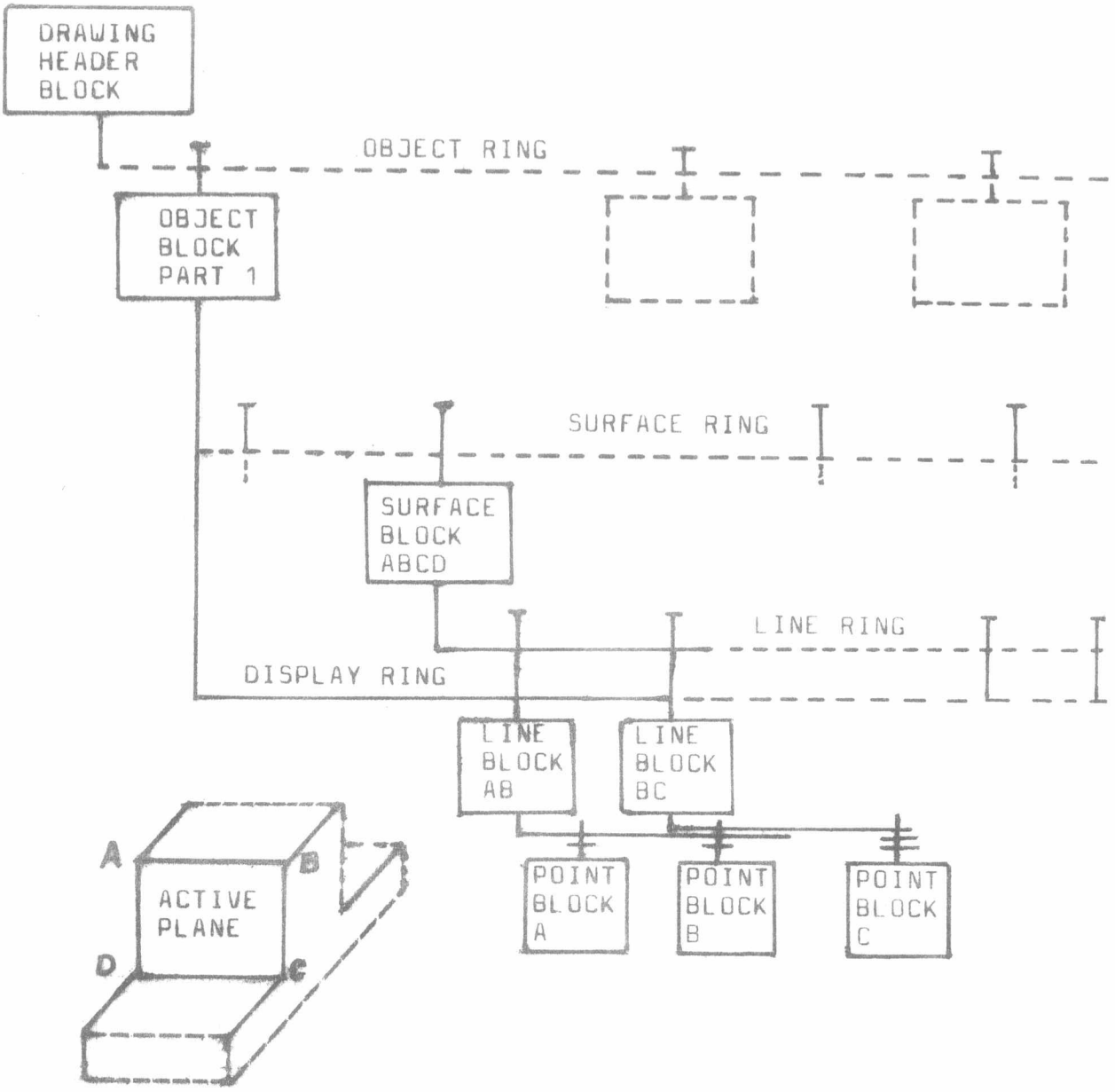
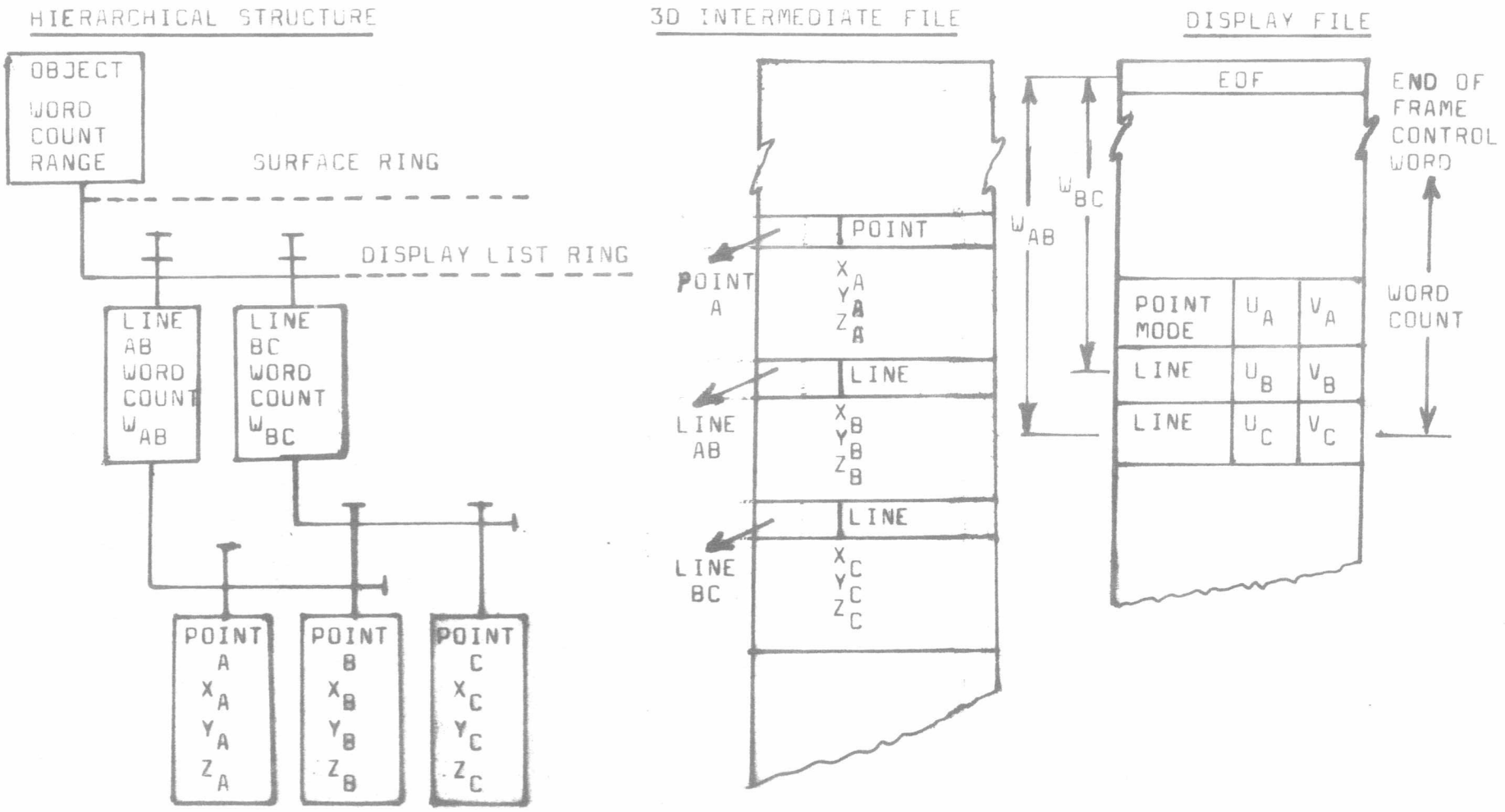


Figure 1 A Hierarchy of a Geometry Data Structure.



MD-14 165

Figure 2 Linkage Between Data Structure and Display

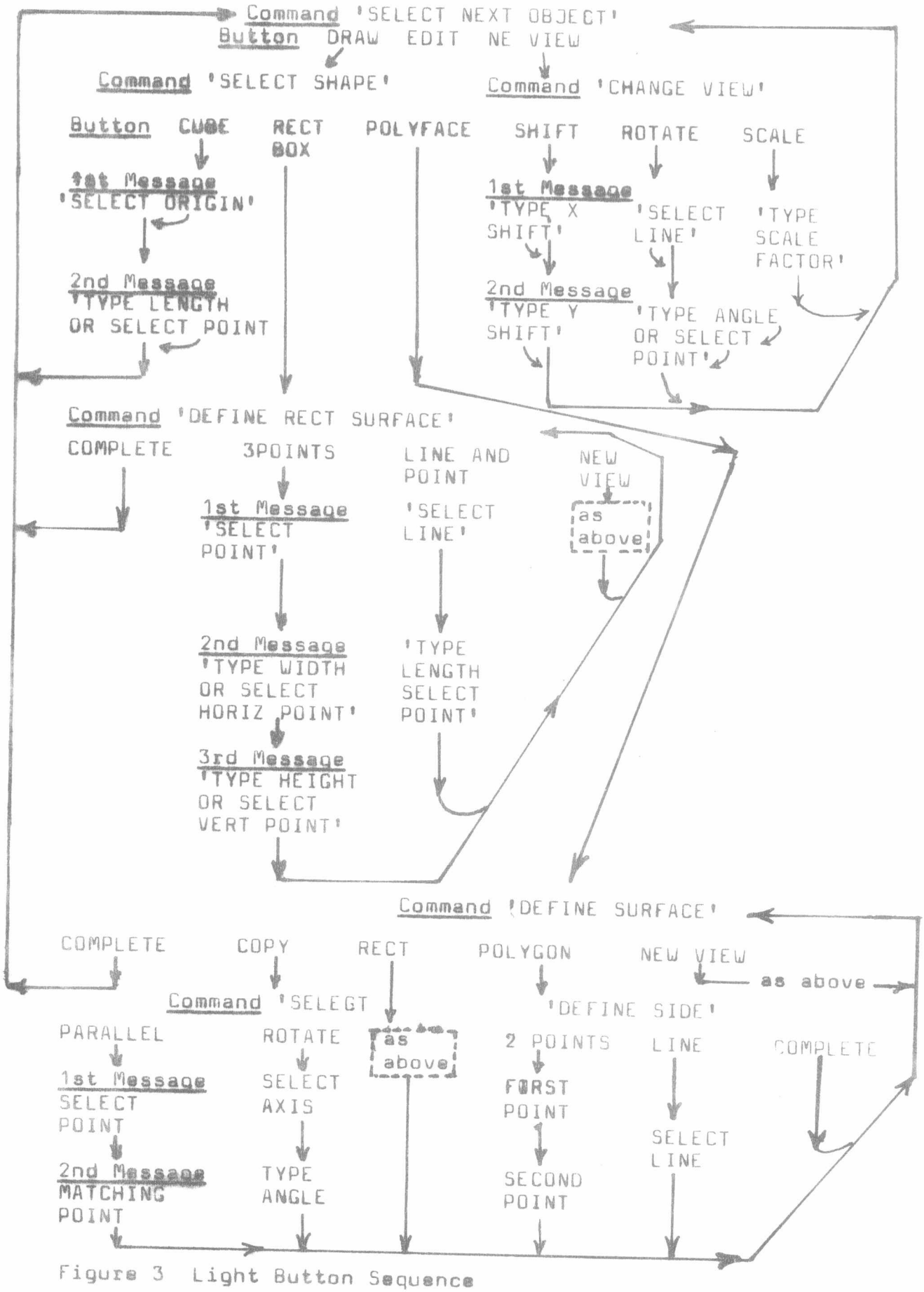
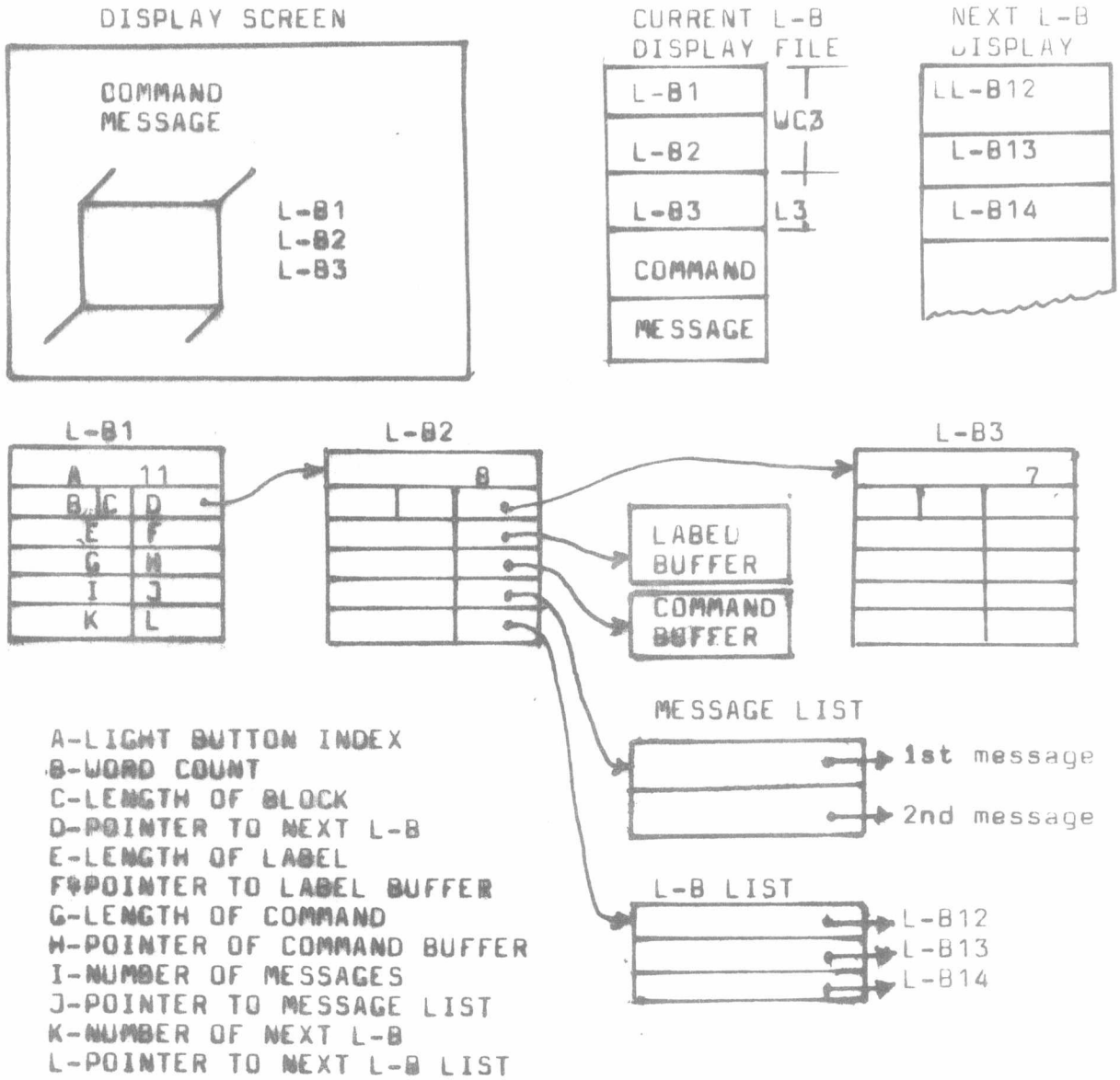


Figure 3 Light Button Sequence

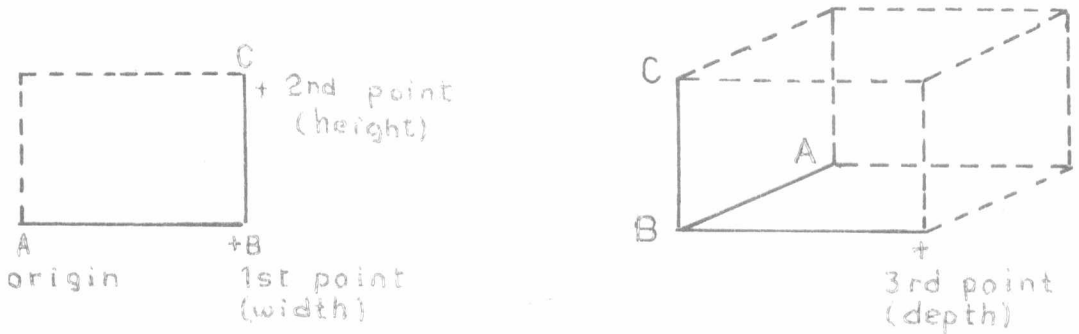


CALL BUTTON (WDCT, INDEX)

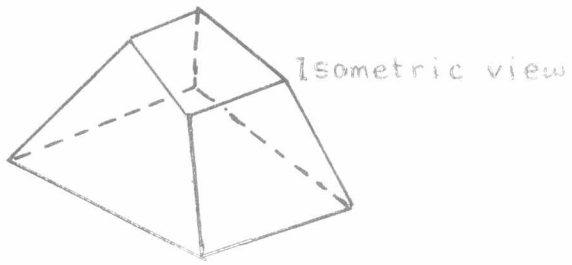
ALGORITHM

Begin
1.Search CHAIN L-B1, L-B2, L-B3, to find WDCT,
2.Set up Next L-B DISPLAY from L-B LIST,
3.Set up 1st message,
4.Return button index number
End

Figure 4 Outline of Light Button Data Block.



(a) Construction of 'RECT BOX'



(b) 'POLYFACE' Construction

Figure 5 Construction of Simple 3D Graphics.