SEMANTOGRAPHY IN RELATION TO TELEVISION

A Thesis

Submitted to

The Faculty of Graduate Studies

of

The University of Manitoba

In Partial Fulfilment

of the Requirements

for the Degree

Master of Science

Department of Electrical Engineering

R. J. Palmer

January, 1979

SEMANTOGRAPHY IN RELATION TO TELEVISION

BY

RONALD JOSEPH PALMER

A dissertation submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

ABSTRACT

The use of symbols, in particular Bliss Symbols, have been used for communication with non-vocal handicapped people. Due to the high resolution required, it is difficult to present them on a graphic display. This thesis investigates methods for displaying Bliss Symbols on a television.

An equation scheme was developed to describe the symbols which allowed a set of raster points to be computed for each symbol. Several methods of displaying the raster points were investigated; the runlength method, the partial symbol method and a new encoding technique called the 'path direction' method. This technique exhibited the most merit in coding Bliss Symbols.

ACKNOWLEDGEMENTS

I wish to thank my advisor and friend, Professor Ed Shwedyk for his patient support and encouragement throughout the course of this work. For the many hours of symbol compiling and debugging, I wish to acknowledge my brother, Danny. I also express a heartfelt thank you to the capable typist and photographer, Debbie Edbom and Everitt Sanville, respectively.

TABLE OF CONTENTS

(TABLE OF CONTENTS continued)

LIST OF FIGURES

(LIST OF FIGURES continued)

LIST OF TABLES

CHAPTER I

INTRODUCTION

1.1 Bliss Symbols

Graphics have been used for communication throughout the ages and even now the Japanese and Chinese use a highly stylized picture writing. Charles K. Bliss (1) analyzed these languages and many others to develop a logic symbolic language known as Bliss Symbols (Appendix A). The symbols remained unused for many years despite the attempts of Bliss to introduce them to the world.

In 1974, the symbols were experimentally used (2) with handicapped children at the Ontario Crippled Children's Hospital in Toronto. These handicapped people were non-verbal and had only limited physical control of their limbs. The results of these experiments were very encouraging and the use of the symbols spread throughout North America.

The handicapped people learned the new symbols quickly and could communicate more efficiently. Large boards with a matrix of Bliss Symbols were placed in front of the child who would then select the proper symbol. Refer to Fig. A.1. (Appendix A) (5-7)

Various techniques for this selection are used. The physically able child can point to the symbols directly with a finger or a pointer. The more handicapped child requires the teacher to point, with the facial expression or eye movement of the child indicating the proper selection. The scanning, verification, recording and interpretation of the symbols require the full attention of the teacher.

To alleviate the teacher's function, electronic scanning boards were designed to sequentially scan the symbols. With these boards and a suitable interface, the handicapped child can independently select symbols; however, only one symbol can be viewed at any time.

A graphical display would permit a sentence of Bliss Symbols to be presented and would provide the handicapped child with the ability to verify and edit his entries and also allow him to communicate directly with others. (9-17)

1.2 Previous Work

As compared to alphanumerics, the graphical display of Bliss Symbols presents some unique challenges. Some of the difficulties of presenting the symbols are outlined by Sawchuk (11). The symbols are not consistent in width, relative size, position, ratio or spacing. Other variables in implementing Bliss Symbols are the size and number of a set. When a handicapped child begins to learn Bliss Symbols, a set large in size and few in number is chosen; but, as the child progresses, more symbols are added to the set with a corresponding decrease in symbol size. For this reason, the feature of variable size would be advantageous.

Recognizing the graphical diversity inherent in Bliss Symbols, J.R. Storey (3) described the symbols using sixty-four component parts. This preliminary work was concerned with estimating the cost and ultimately establishing the feasibility of using a television as a Bliss Symbol display terminal. Vanderheiden (4) also broke the symbols into components to fabricate symbols on a portable dot matrix printer.

1.3 Present Work

This thesis investigates techniques in presenting Bliss Symbols on a television using a commercially available micro-processor system and display unit.

The symbols were first described using equation descriptors which provided a standard to scale symbols to a variety of sizes. This feature was useful in the initial stages to determine such things as minimum resolution and later it was useful in compiling symbol sets of any size.

Several techniques were investigated to determine the most efficient means of displaying the symbols; the known run length method, the partial symbol method as proposed by Storey and a new technique called the 'path direction' method. The path direction method exploited the contiguous dot patterns of the symbols and was found to be the most favourable method.

CHAPTER II

BACKGROUND

2.1 A Typical System

A typical system is shown in Fig. 2.1. The scanning board contains the matrix of Bliss Symbols and the handicapped person selects the proper symbol via the computer. The selected symbol is saved in the computer's memory and also displayed on television.

Although the scanner is usually a board a portion of the television screen could be used as the scanner. However, there are disadvantages with such a system:

1. Lost screen area for sentences and stories;
2. Less anticipation of when the symbol will be displayed for scanning purposes at any one time;
3. A more sophisticated program with two cursors, split screen, etc.

A scanning board system along with a commercially available unit to interface to the television was employed by the author.

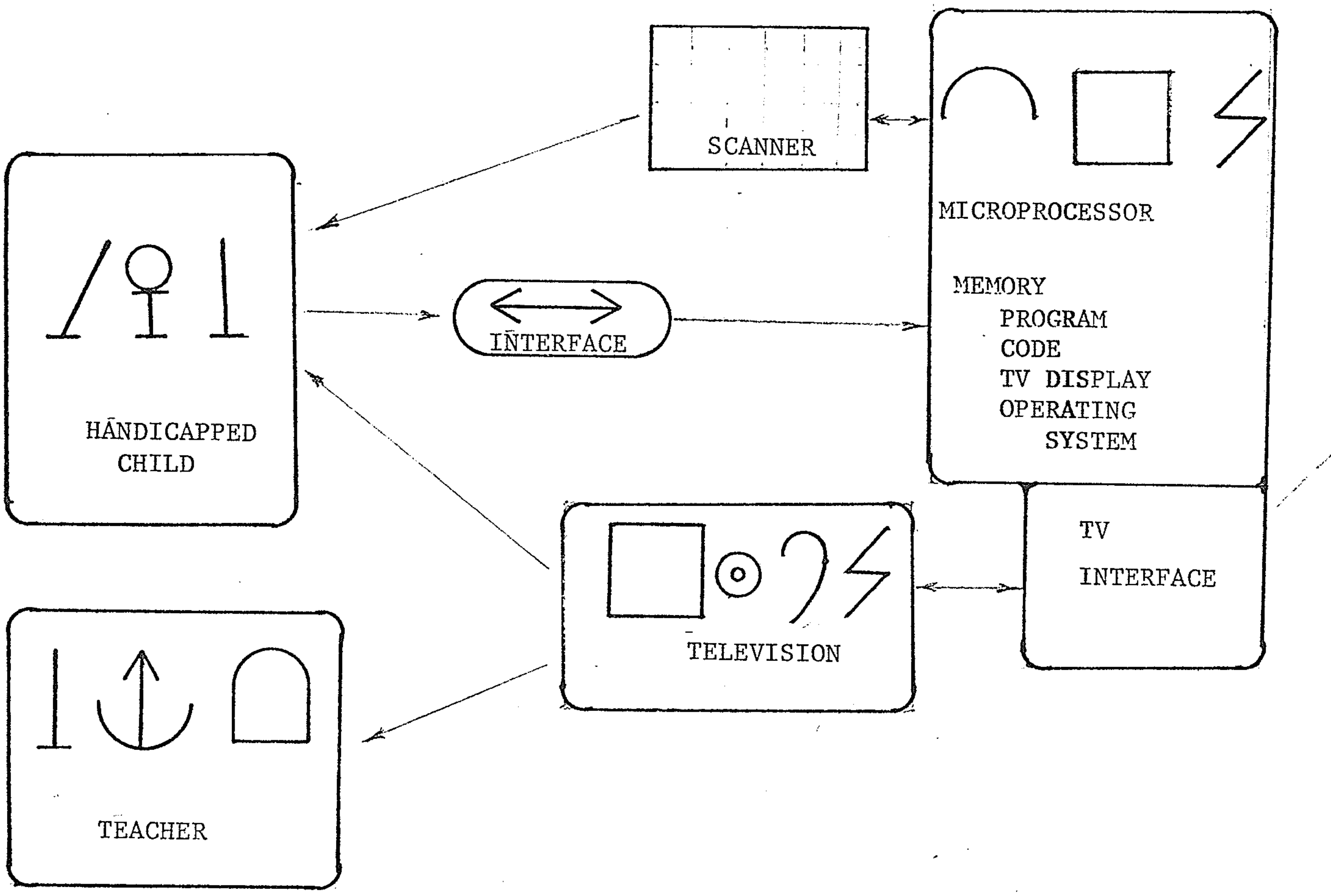


Fig. 2.1 Micro-processor based Communication System

2.2 Television Resolution and Timing

A North American television has a resolution of 700 dots or pixels across with 525 interlaced rows. However, due to overscan a reduction of at least ten percent can be realized or 630 x 461 - 290,430 bits of on-screen information. The memory requirements using bit-mapping would be 36K bytes (8 bits/byte). However, this is for maximum resolution and it has been shown by the author's experiments, as will be seen in later chapters, that eight K bytes is adequate.

A television uses a high speed raster scan. Alphanumerics use code generators to create the character as the raster scans that particular pixel. To do this, the code generator would have to operate in the order of 5 MHz; much too fast for a contemporary microprocessor with a clock frequency of 1 MHz. Buffer hardware is, therefore, necessary to interface the high speed television to the slower microprocessor.

2.3 Display Unit

The computer was interfaced to the television by means of a commercial unit called "Merlin." This device bit maps eight K bytes (64 K bits) of memory onto the T.V. screen with a resolution of 320 lateral dots by 200 vertical dots. This eight K of memory is shared with the computer by DMA (Direct Memory Addressing) and the bits written into these memory locations are the bits displayed. Merlin has control of the bus 42 percent of the time leaving only 58 percent of the real time for computer execution time.

Any theoretical technique considered in displaying the symbols must take these restrictions of timing and resolution into account.

2.4 Problems in Displaying Bliss Symbols

Consider first what Bliss Symbols are comprised of and how they were designed. Bliss (1) contends that practically all definite and indefinite objects can be depicted by using one hundred strokes or basic shapes. This does not seem excessive if compared to the one hundred plus (26 lower case, 26 upper case, plus punctuation and symbols) characters used on the conventional keyboard. However, to compact so much information into each symbol, other problems arise in presenting them graphically. These are:

1. The detail or resolution required for presentation of the symbols far exceeds that of normal alphabetical characters. Vanderheiden contends that as many as eighty dots vertical by an indefinite horizontal count must be used. (4)

2. The width of the symbols or the aspect ratio is undefined. Simple symbols can be arranged side by side to create more complex ones. In addition, action indicators placed above some symbols add to their overall height.

3. Unlike alphanumerics, symbols can be laid on top of each other (superimposed) to form new symbols. Therefore, there is no definite set of symbols. Simple symbols can be arranged into a new larger symbol, much like letters make a word.

4. Characters are written between imaginary upper and lower lines; the placement of a symbol relative to these may indicate a different meaning.

5. The relative size of the symbol may emphasize the degree or extent of a meaning or it could have an entirely different meaning.

6. The size and/or number of symbols to be displayed on a television may be different depending on the visual acuity of the user.

Many of the above problems in displaying Bliss Symbols have no conterpart with alphanumerics. These problems combined with the restrictions imposed by the display unit, television and microprocessor make the task of displaying Bliss Symbols non-trival. Chapter III outlines the theoretical aspects of the symbol display process.

CHAPTER III

DISPLAYING BLISS SYMBOLS

This chapter discusses theoretical means of displaying Bliss Symbols efficiently on television with the actual experiments being discussed in Chapter IV.

3.1 Equation Format

As was discussed, Bliss Symbols are not as easy to display as alphanumerics, but they do have some common features that Bliss has incorporated. The symbols are all composed of relatively simple shapes; straight lines and circle segments. Since the microprocessor is the core of the system, a scheme using equations to describe symbols would be feasible. Of course, one equation would not describe the entire symbol but rather only a portion of the symbol and then these portions would be superimposed to make a whole symbol.

Only two equation types were chosen, a line and a circle working on a cartesian coordinate system with the origin being the lower left corner of each symbol. Two starting codes were chosen, Q for a circle, / for a line. The parameters given to describe the circle were as follows:

X,Y,R,A,B

where -X,Y are the coordinates for the centre of the circle

-R is the radius of the circle

-A is the start of the arc (in degrees) going counterclockwise with zero degrees being horizontally right from the centre.

-B is the termination of the arc in degrees.

The symbols are normalized to a ten unit high structure, therefore, if a semicircle was to be described in the upper half of the symbol structure, the following parameters would be given: Q 5,5,5,0,180 (See Fig. 3.1).

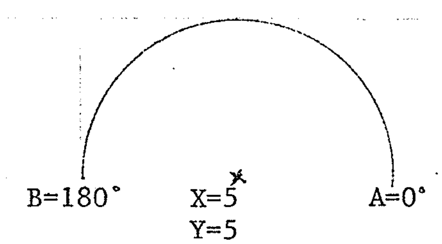


Fig. 3.1 An Example of an Arc

Similarly a line is given a start and a stop position. The parameters are as follows:

/ Xo,Yo,X1,Y1

where Xo,Yo are the start coordinates

X1,Y1 are the terminal position coordinates

A diagonal drawn across the entire height of the symbol may appear as in Fig. 3.2.

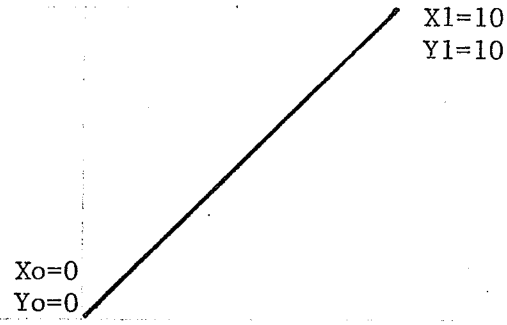


Fig. 3.2 An Example of a Line

A number of these circle and line segments superimposed would be sufficient to describe any symbol. One additional code E would act as an "end of symbol" delimiter and would be useful in compiling and decompiling sequential computer data files. Fig. 3.3 gives an example of an entire symbol.

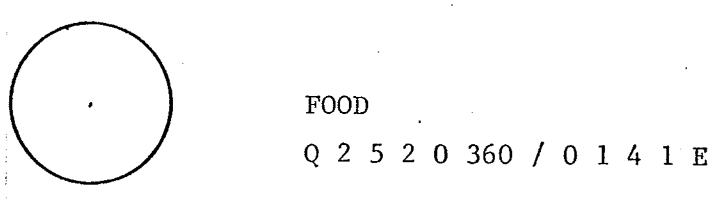


Fig. 3.3 The symbol "FOOD", Described by Equations

The advantages of using equations are as follows:

1. The symbols could be standardized independent of their size.

2. Smooth, regular shapes result with no special artistic talent or subjective criticism necessary.

3. Memory is conserved since only a few parameters are necessary for each symbol regardless of symbol size or resolution.

The disadvantages will become apparent in Chapter IV where it is discovered that to convert the equation into a symbol takes an unreasonable amount of time, if the parameters are to be used directly in displaying the symbols.

3.2 Digitizing the Equations

Equations can be used to describe the symbol but cannot be used to display the symbol directly. In a digitized system, only the presence or absence of a white dot on the screen is displayed. To convert the equation into a pattern of dots that resembles the equation is the next topic of discussion.

When using digital techniques, as is the case here, it is not possible to display the symbols as straight and curved lines 'directly', but only as the presence or absence of illuminated dots in discrete positions. At a distance the eye and/or brain join the dots to resemble a line. Stating this in another way, the specific resolution is one which involves visual perception on a television screen. The given equation must be presented in such a fashion that the dots representing it may be perceived as a symbol component.

With the symbols being represented as equations or an infinite number of dots, a scheme must be devised to match these equations with a finite number of dots placed in a definite position. Criteria must be formulated for assigning samples from the equation to the dots of the array. Two problems arise which the criteria must reflect:

1. A decision must be reached on the number of samples taken along the graph of the equation(s) representing the symbol. To conserve computer time only as many samples as necessary should be computed. The number of samples required depends on the resolution.

2. A resolution must be selected to represent the symbols. With the television having a fixed resolution, the only method to increase the apparent resolution of the symbol would be to increase the size. Therefore, the size and resolution of the symbol become inter-related.

3.2.1 Sample Rate For a Straight Line

As discussed it is not possible to draw a continuous line with a point-plotting display; the discrete grid permits intensification only of points lying on intersections of the coordinate grid (dots). To approximate the desired line a sequence of dots can be displayed as shown in Figure 3.4. To utilize all the dots close to the line, the sample space (distance between samples) must not exceed the distance between grid dots. This constitutes the minimum sampling rate; however, to assign dots to samples a technique called 'smoothing' (discussed later) is employed requiring a somewhat higher sampling rate. Newman and Sproull (15) describe similiar techniques to convert equations to discrete grids.

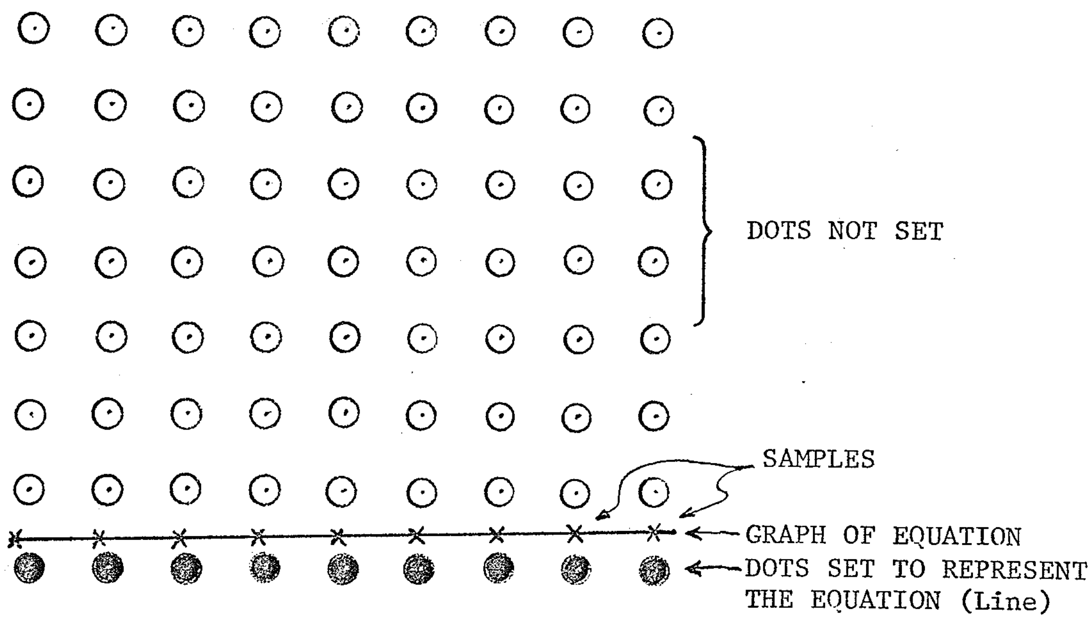


Fig. 3.4 Minimum Sample Rate for a Line

3.2.2 Sample Rate for a Curve

With reference to Fig. 3.5, it can be seen that curved portions of the symbol require more samples than a horizontal line; otherwise portions of the symbol could be missed entirely. In Fig. 3.5a, the samples are taken at a rate with respect to the horizontal direction. Taking ten samples of the graph and designating a dot to each is not adequate. One could sample the graph at a rate proportional to the arc length (distance along the graph) and not the horizontal distance. This would provide more samples as is shown in Fig. 3.5. This would at first appear to be sufficient, however, there are cases in which it is not. Take for example (Fig. 3.6a) a very small circle with a circumference less than the required arc length for the sample, or a very sharp discontinuity (Fig. 3.6b). Both of these require dots and, therefore, should be given sample points. The following criterion is suggested:

The number of samples taken on a graph to represent an equation shall be proportional to the curvature with the following exceptions:

1. A line has no curvature, and as discussed earlier should have a sample rate exceeding the resolution in one direction (x or y).

2. A discontinuity has infinite curvature but only one sample point will be taken. With the line and circle segments used, discontinuities occur only at the start or end of a segment.

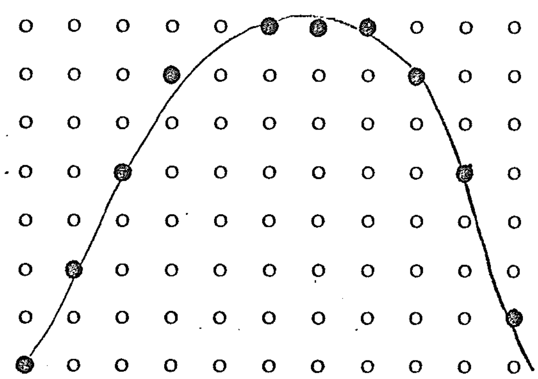


Fig. 3.5a Sample Rate Constant with Respect to Horizontal Direction

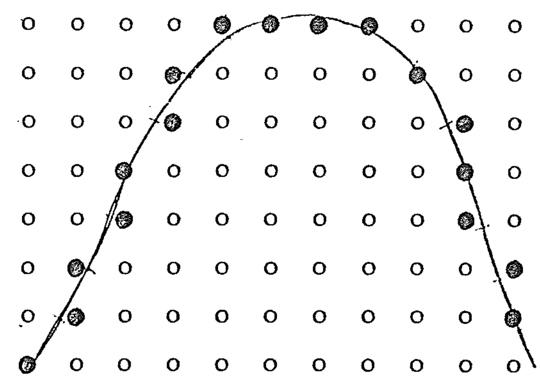


Fig. 3.5b Sample Rate Constant with Respect to Arc Length

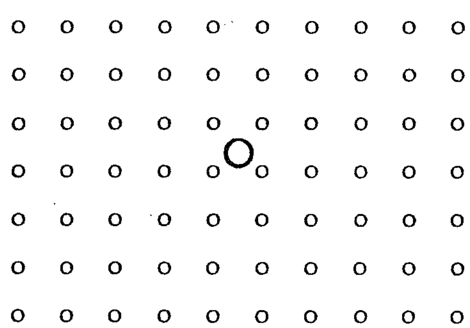


Fig. 3.6a Small Circle

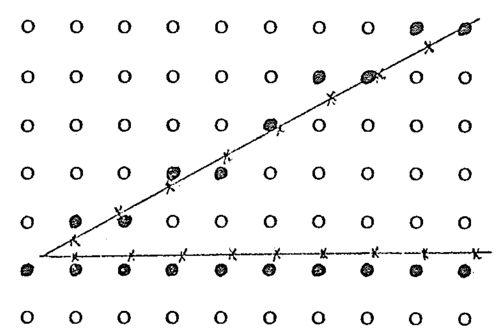


Fig. 3.6b Discontinuity

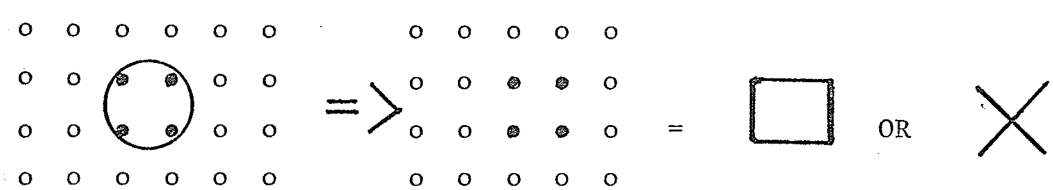


Fig. 3.7 Representing a Circle

Superposition usually overlaps endpoints and the probability of getting at least one sample near the discontinuity is very high. (Refer to Fig. 3.6b).

The above states the criteria for selecting a minimum sample rate in order to minimize computation time in designating the dots. As will be seen in Chapter IV a somewhat higher sample rate will be used to overcome other problems.

3.2.3 Symbol Resolution

The resolution or the dot matrix size required to define a symbol is difficult to choose because of the diversity in the Bliss Symbols. Various manners of choosing this were considered.

Refer to Fig. 3.7 where a circle is to be represented by a dot matrix. What dot matrix size is required to represent the circle? Four dots could be interpreted as a cross or square. Six, eight or ten dots would probably be adequate depending on the imagination of the viewer. Arbitrarily picking eight dot rows and knowing that some Bliss Symbols have the equivalent of three circles stacked vertically, one could estimate a minimum vertical resolution of twenty-four rows of dots. The horizontal resolution is dependent upon the symbol being presented and, therefore, remains variable.

Another way to estimate the resolution required would be to consider the 'error', defined as the average distance between the sample of the graph and the dot designated. This is done with respect to both the X and Y directions and could give some indication as to the accuracy of presenting the symbol if compared to the error of some arbitrary shape such as a circle. However, error can vary greatly depending on how the graph is positioned in the dot array. Therefore, the error can only be used as an indicator for resolution if it is minimized by optimum placement of the symbol. This will be discussed in more detail later.

In deciding to use a television as a display another resolution factor is limiting. The grid point spacing is constant; typically for a 12 inch screen the spacing is .014 inch. Therefore, if the symbols are to have apparent increased resolution they will have to be increased physically in size on the television screen. However, this reaches a limit for two reasons:

1. It restricts the number of symbols that can be placed on the screen at one time.

2. It greatly increases the memory used for symbol generation.

Although the above points are important considerations in choosing the resolution, the deciding factor in a minimum resolution will be the subjective opinion of the

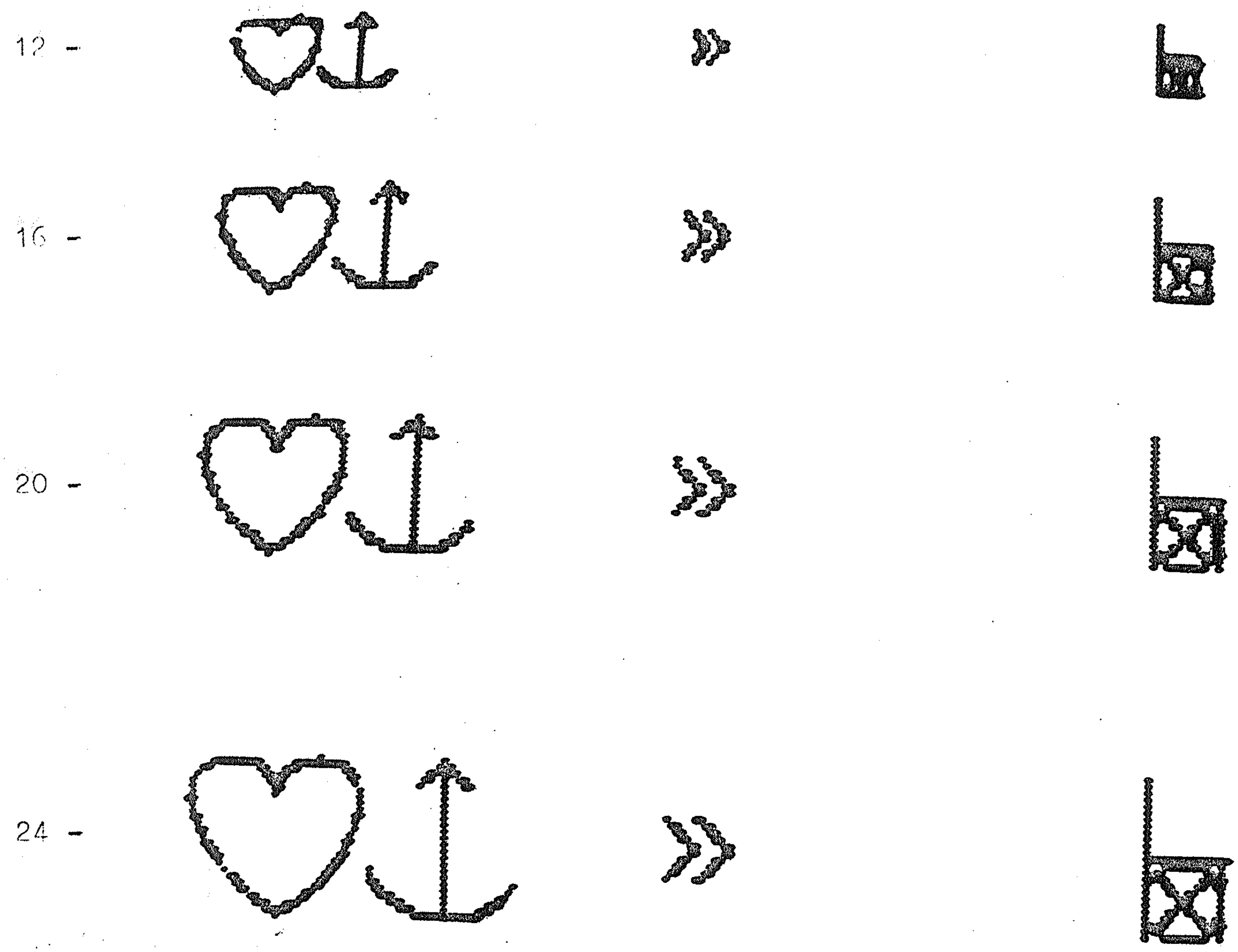


Fig. 3.8 Comparison of Resolutions of 12, 16, 20, 24 Vertical Grid Lines "THANKS FOR THE WHEELCHAIR"

viewer. It is for this reason that lengthy debates on the resolution are omitted. Figure 3.8 displays a Bliss Symbol sentence with vertical resolutions of 12, 16, 20, and 24 grid lines. From this the author concluded that a resolution of 20 vertical grid lines reasonably represented the symbols and accordingly this resolution was used in the experiments of Chapter IV.

3.3 Error in Designating Dots for Sample Points

The symbols were represented by a set of equations each with its own limits. The equations were then sampled at a rate according to the criteria stated previously and a set of sample points, each with its x, y coordinate were compiled. From this list of points a set of dots were designated to represent the symbol. Since dots and the sample points were not in the same place; there was error which was defined as:

$$E_x = \sum_{i=1}^{N} \frac{S_{xi} - D_{xi}}{N} \qquad E_y = \sum_{i=1}^{N} \frac{S_{yi} - D_{yi}}{N} \qquad (3.1)$$

where: E_x - error in horizontal direction (units of dot spaces)

E_y - error in vertical direction

N - number of sample points

S_{xi} - x coordinate of sample point S

D_{xi} - x coordinate of dot designated by sample point S

S_{yi} - y coordinate of sample point S

D_{yi} - y coordinate of dot designated by sample point S

Error could be used as an analytical tool in evaluating how well a symbol could be presented if the error was minimized by repositioning the symbol. Consider the box symbol (Fig. 3.9) as an example with samples taken every dot space. It is possible to position the symbols exactly over the dot positions resulting in error of zero; or translating the same box up and to the right one half dot space results in an error of: E_x = .5 dot spaces and E_y = .5 dot spaces.

Positioning a symbol does not change its size or shape and since the greatest shift in repositioning is less than one half a dot space, it is insignificant in relative placement to other symbols. In view of this, a symbol should be positioned to give the least error which in turn means the 'best fit' between sample points and designated dots representing the symbol.

3.3.1 Minimizing Error

Considering only one orthogonal direction, suppose there were N sample points for a symbol and that on the average they were .1 grid or dot spaces to the right of any grid line.

This .1 grid space average deviation known as the error will have the general definition of equation (3.1).

Fig. 3.9 Error Dependence on Placement

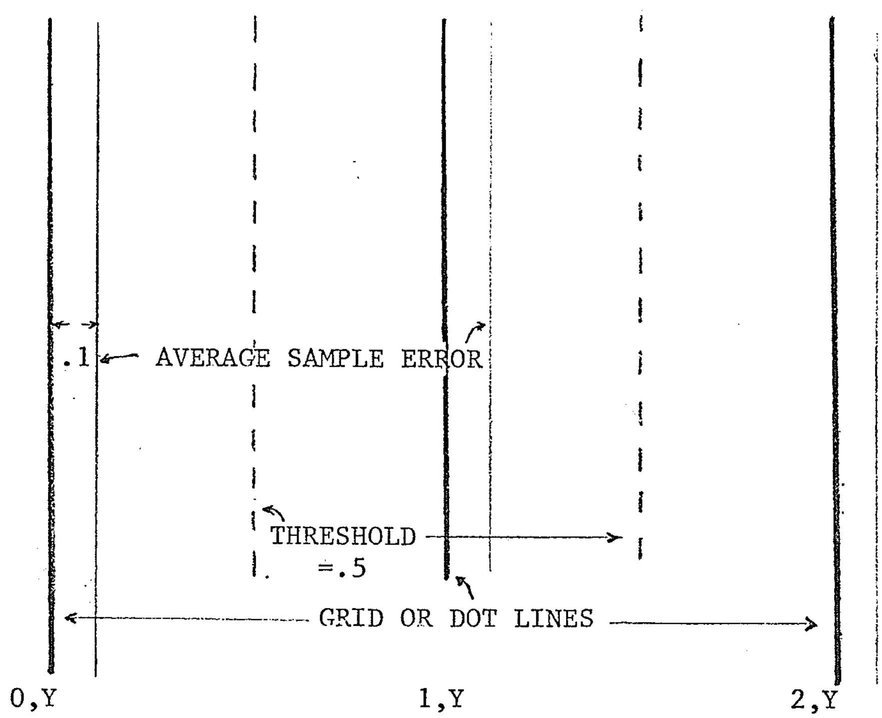


Fig. 3.10 Calculating Error

If all the sample points were shifted E, or in this case .1 grid spaces to the left, the error would supposedly be reduced to zero. However, there are some points that now will be related to another grid line or the shifting will make their nearest dot line different. In the example above, it would be .1 x N. Therefore, the number of dots that do not have new dot lines would be (1 - .1) x N. If the average error E is subtracted from each sample point, the resulting error E_r with the repositioned sample points will be:

$$E_r = \left(\begin{array}{l}\text{\# of dots that do not} \\ \text{have new dot lines} \\ \text{designated}\end{array}\right)(\text{error}) + \left(\begin{array}{l}\text{\# of points} \\ \text{with new dot} \\ \text{lines}\end{array}\right)\left(\begin{array}{l}\text{average error} \\ \text{of points of} \\ \text{new dots}\end{array}\right)$$

$$= (1 - .1)(N)(-E+E) \quad + \quad .1\ (N)(.5-E/2)$$

$$= (1 - .1)(N)(0) \quad + \quad .1\ (N)(.5-E/2)$$

Although the average error of points that crossed the threshold could vary from 0 to E, a value of E/2 was chosen to represent a random distribution of dots.

Experiments in Chapter IV implemented equation (3.1) in an iterative process, shifting the samples the amount of the error, until the error converged to a practical limit.

3.4 Assigning Dots from Sample Points

The problem still remains as to how the sample points will be used to designate grid dots. Referring to Fig. 3.11 which is the graph of a symbol segment located in a matrix of grid points, a practical method of choosing the proper dots to represent this graph must be devised. The criteria to choose the sample points has already been discussed, now it remains to designate the proper dots from these sample points.

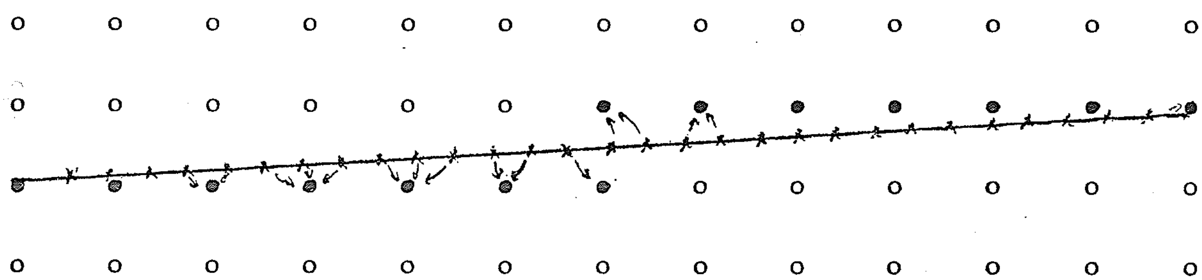


Fig. 3.11a Symbol Segment Located in a Matrix of Dots

A dot could be designated if any sample has it as its closest neighbour. In this case, the segment or line would be represented by Fig. 3.11b.

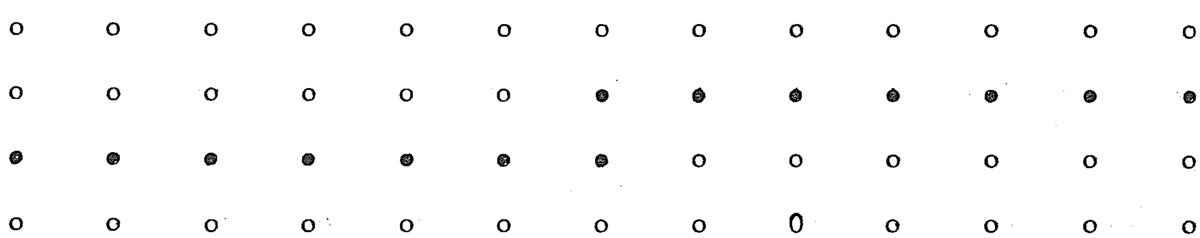


Fig. 3.11b 'Any Sample Point' Produces a Dot

The eye and brain of the viewer would interpret this as Fig. 3.11c.

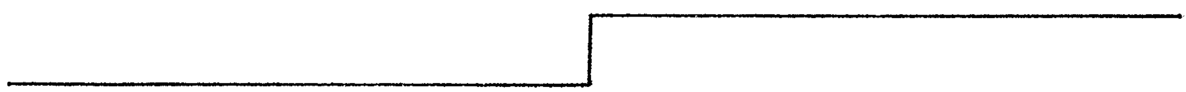


Fig. 3.11c The Eye/Brain Interpretation

It would improve the representation of the segment if one corner was omitted as in Fig. 3.11d.

Fig. 3.11d Smoothing

Fig. 3.11e Interpretation of Smoothing

This smoothing technique can be implemented by having a minimum number of 'closest samples' before a dot is designated.

In the example above at least two neighbours would be adequate, but then again this would depend on the sampling rate. Where there is high curvature in the graph more samples would be present and by following the above criteria more dots would be designated to represent the curve. A discontinuity would have many sample points in its immediate vicinity and would therefore be assured a designated dot. This feature should improve symbol representation and experiments proved this to be correct. This, of course, was the subjective opinion of the author and was accomplished by simultaneous, side by side comparison of identical symbols with and without this smoothing technique. (Fig. 3.12).

3.5 Using Equations To Describe the Symbols

To represent a Bliss Bymbol employing equation techniques requires a number of steps. First the symbol is separated into a number of equations, each describing a portion of the symbol. The equation parameters are then tabulated and entered into the computer by manual input or by means of a prepared data file. The computer then computes the sample points, minimizes the error and smooths the ultimate display by selecting the proper dots and then displays all the dots that it selected.

All variables such as height, resolution, aspect ratio, etc., would be inherent in the program and could be modified easily. When such a program is executed the apparent disadvantage is the excessive amount of time required. Therefore; a second approach to the problem was sought.

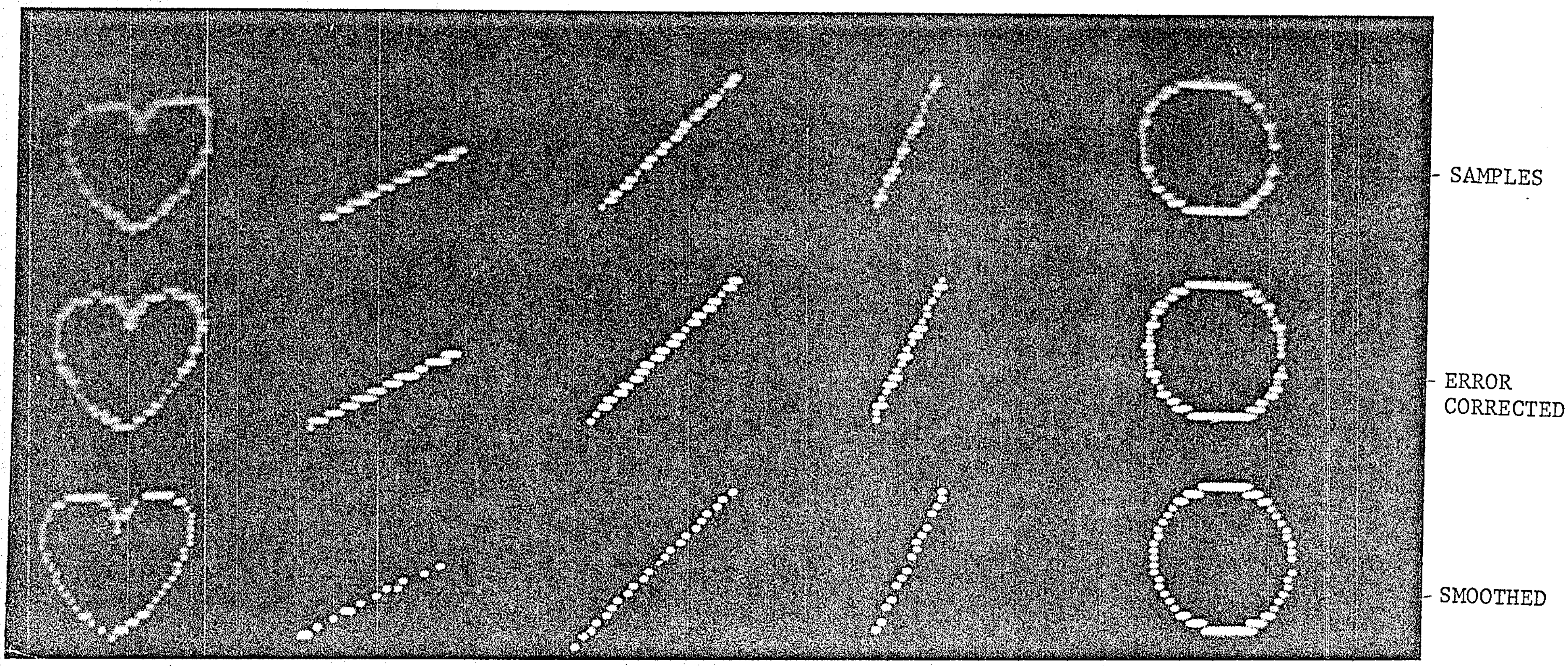


Fig. 3.12 Subjective Comparison of Smoothing

The equations could be compiled for each symbol and the coordinates of each dot stored. This in fact is the second method and it is much faster, however, at the price paid of less flexibility in changing the size of the symbol.

Also in comparison to the equation method, an increase in memory requirements is required since the coordinates occupy more memory than the equation parameters. This becomes more significant as the symbol resolution increases.

If anything at all was achieved in using equations, one feature was more outstanding than any other--the dots representing the symbols could be generated in an analytical manner; in this way standardization in representing the symbols regardless of size was possible. With the dots established, it would now be possible to seek other methods of representing the symbols efficiently.

3.6 Display Method Three--Path Direction

A matrix 20 x 20, containing 400 bits of memory, would be the requirement to represent a symbol of that resolution (20 x 20). However, there are certain properties with Bliss Symbols that could yield a reduction in memory.

In observing the symbols, it is seen that the same shapes or paths are repeated; sometimes translated vertically or horizontally to another position with the shape and size remaining the same. Therefore it would be advantageous in minimizing memory requirements to map out these symbol shapes only once.

Another observation is that the number of dots set in an array to display a symbol is always fewer than those not set. This becomes more apparent as the resolution or size increases.

Bliss Symbols or any calligraphy, especially writing, is continuous, that is the dots that represent it are not positioned randomly in the matrix, but are connected in a contiguous manner. Also in drawing a symbol the probability of going straight is much greater than of turning sharply.

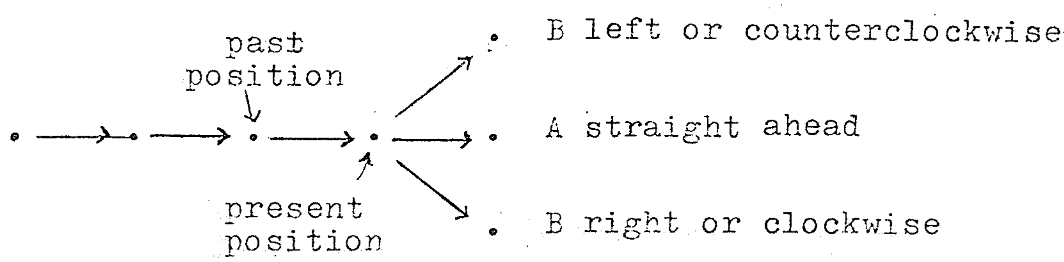


Figure 3.13

As an example, consider Figure 3.13. Imagine that this is a portion of a symbol being drawn, dot by dot. From observation it can be noted that the probability of the next dot being A is much higher than that of B and as the resolution of the symbol increases, this becomes more significant.

Based on the above properties a code was devised by the author with the objective of saving memory. The code is arranged in two bit portions as shown in Figure 3.14 with the most probable occurrence being given the smallest code as suggested by Huffman (8).

The sample shown in Figure 3.14 initially has a horizontal direction, after this the path could be any one of eight directions, and the counterclockwise, clockwise and straight ahead designations are relative to the direction. The direction is designated with three bits in a direction register and are given designation as indicated in Figure 3.15.

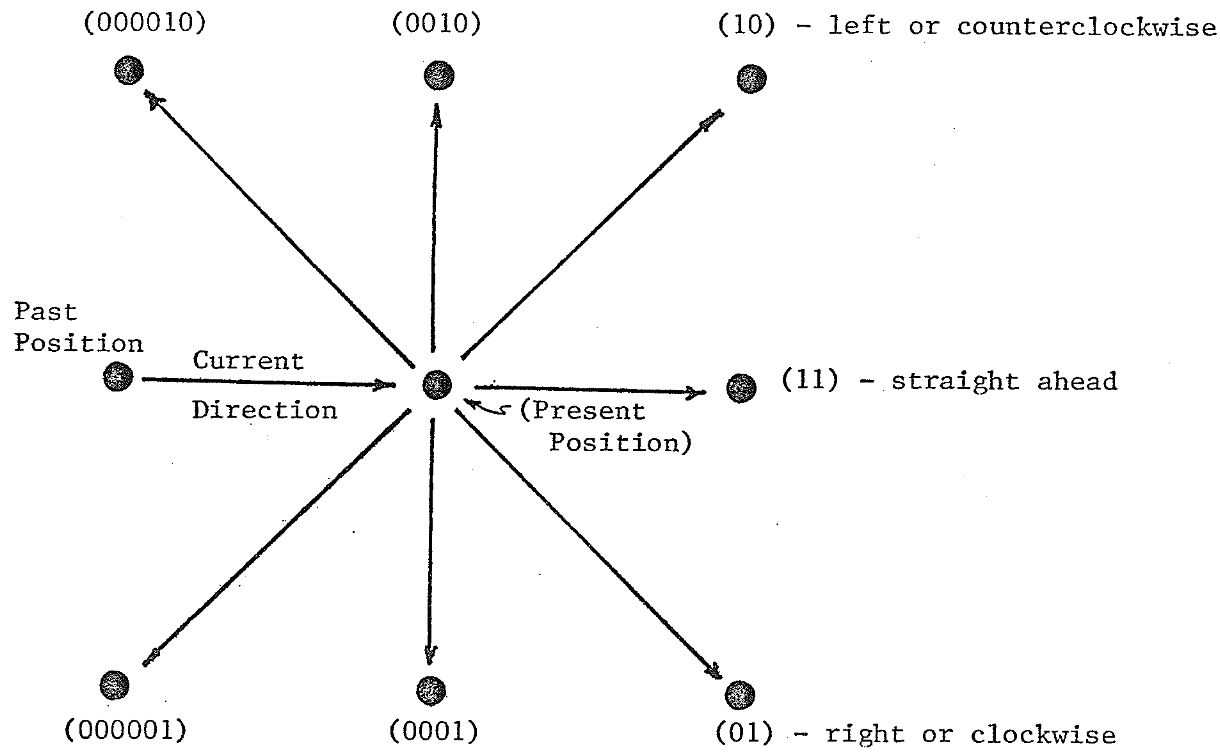


Fig. 3.14 Code for Path Direction Method

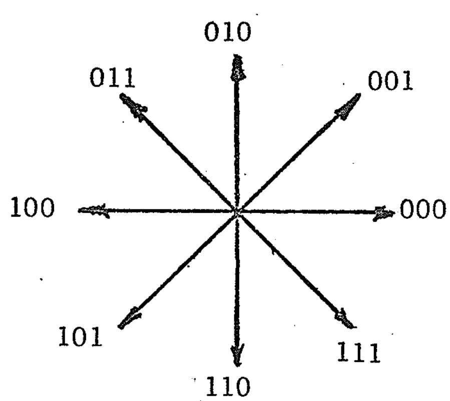


Fig. 3.15 Direction

The two bit code alters the direction register as follows:

11 - go straight ahead one position

01 - (clockwise) subtract one from direction register

10 - (counterclockwise) add one to direction register

00 - not enough information, get another two bit set

00 01 - (clockwise) subtract two from direction register

00 10 - (counterclockwise) add two to direction register

00 11 - pen down

00 00 - not enough information, get another two bits

00 00 01 - (clockwise) subtract three from direction register

00 00 10 - (counterclockwise) add three to direction register

00 00 11 - pen up

00 00 00 - stop, end of symbol

To clarify the above, consider the following example with the starting point in the lower left corner and the direction register set to zero. Refer to Fig. 3.16.

	Code	Direction Register
A - go ahead	11	000
B - pen down	0011	000
C - ahead	11,11,11, 11,11	000
D - left two	00,10	010

E - ahead two	11,11	010
F - left one	10	011
G - left one	10	100
H - ahead two	11,11	100
I - left two	00,10	110
J - ahead two	11,11	110
K - end of symbol	00,00,00	110

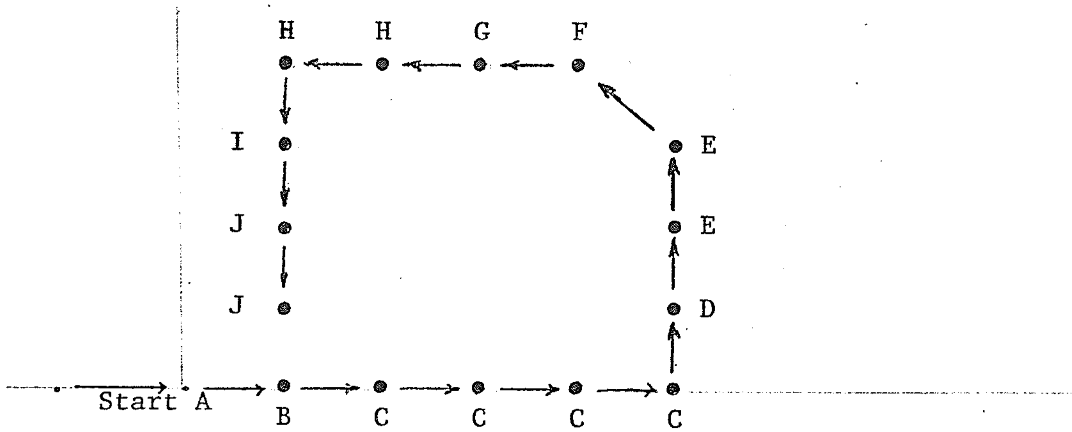


Fig. 3.16 Example of Path Direction Method

Combining the code of the previous example into 8 bit hex bytes results in:

11,00,11,11 = CF
11,11,11,11 = FF
00,10,11,11 = 2F
10,10,11,11 = AF
00,10,11,11 = 2F
00,00,00,00 00
01,01,XX,XX = 5X

The last six bits indicate the value of the width, that, is the number of spaces that the symbol occupies horizontally. The width is used in positioning the symbols on the screen. Here the width is put on the end of the code as a convenience in decompiling the code; but the width could also be computed from the pattern.

To represent this 5 x 6 symbol, 48 bits are required with this new method. To represent all the dots would requre only 5 x 6 = 30 bits directly and it can be concluded that this method is not efficient with low resolution characters.

If the arbitrary assumption of 2.5 bits per dot is suggested as the memory requirement for this new path direction method, less than 40% fill is required to justify its use. Fill is defined as:

$$\text{Fill} = \frac{\text{number of dots displayed}}{\text{number of total dots}}$$

In typical calligraphy or in this case symbol presentation, the fill decreases as the resolution increases.

An example is calculated below for a box shaped symbol to illustrate the memory requirements of a 'conventional' versus 'path direction method' as the size (resolution) increases. The number of dots for an n x n box would be 4(n-1) while the total number of dots would be n x n. Therefore, the percent fill would be: $F(\%) = \frac{4(n-1) \times 100}{n^2}$

The graph of Figure 3.17 illustrates the point at which the new path direction method becomes more efficient for this box symbol at an estimated 2.5 bits/dot of memory over the conventional 1 bit/dot method. Therefore, memory savings should occur if the symbol has a resolution of greater than nine by nine dots. A circle has less fill and would require even less resolution to be competitive with the 1 bit/dot method.

To determine the relative merits of the path direction method other techniques of encoding Bliss Symbols were investigated.

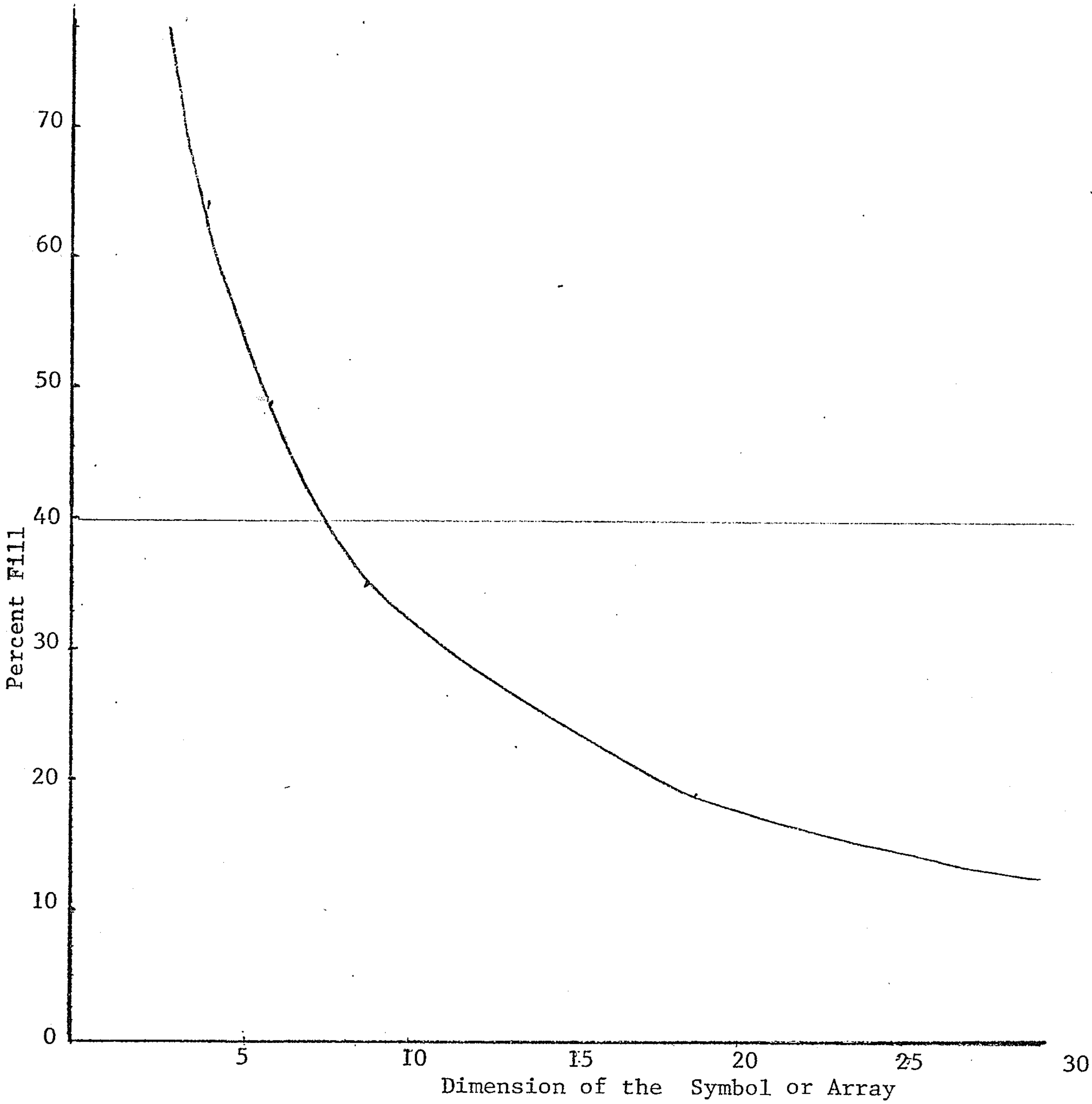


Fig. 3.17 The Dimension of the Array Vs. % Fill

3.7 TWO OTHER METHODS

3.7.1 Run Length

The run length method encodes a symbol by recording the length of horizontal runs of ones or zeroes. Each time a transition occurs from one to zero or vice versa, the length of the run is determined. This method exhibits the most merit with long runs. As an example of this method consider the heart symbol of Figure 3.18. The longest run is 23 which implies a 5 bit block code. Since 92 runs occured, a total of 460 bits of memory are required.

To determine the average memory requirements per symbol, twenty five typical Bliss Symbols were chosen. For a vertical resolution of 20 grid lines the number of transitions were tabulated for each symbol. Considering the more complex symbols, the maximum run could easily reach 64; therefore six bits were used to block code each run. The memory requirements are summarized in Table 3.1 along with the path direction method. As can be seen, the run length method requires almost three times the memory to code the symbols.

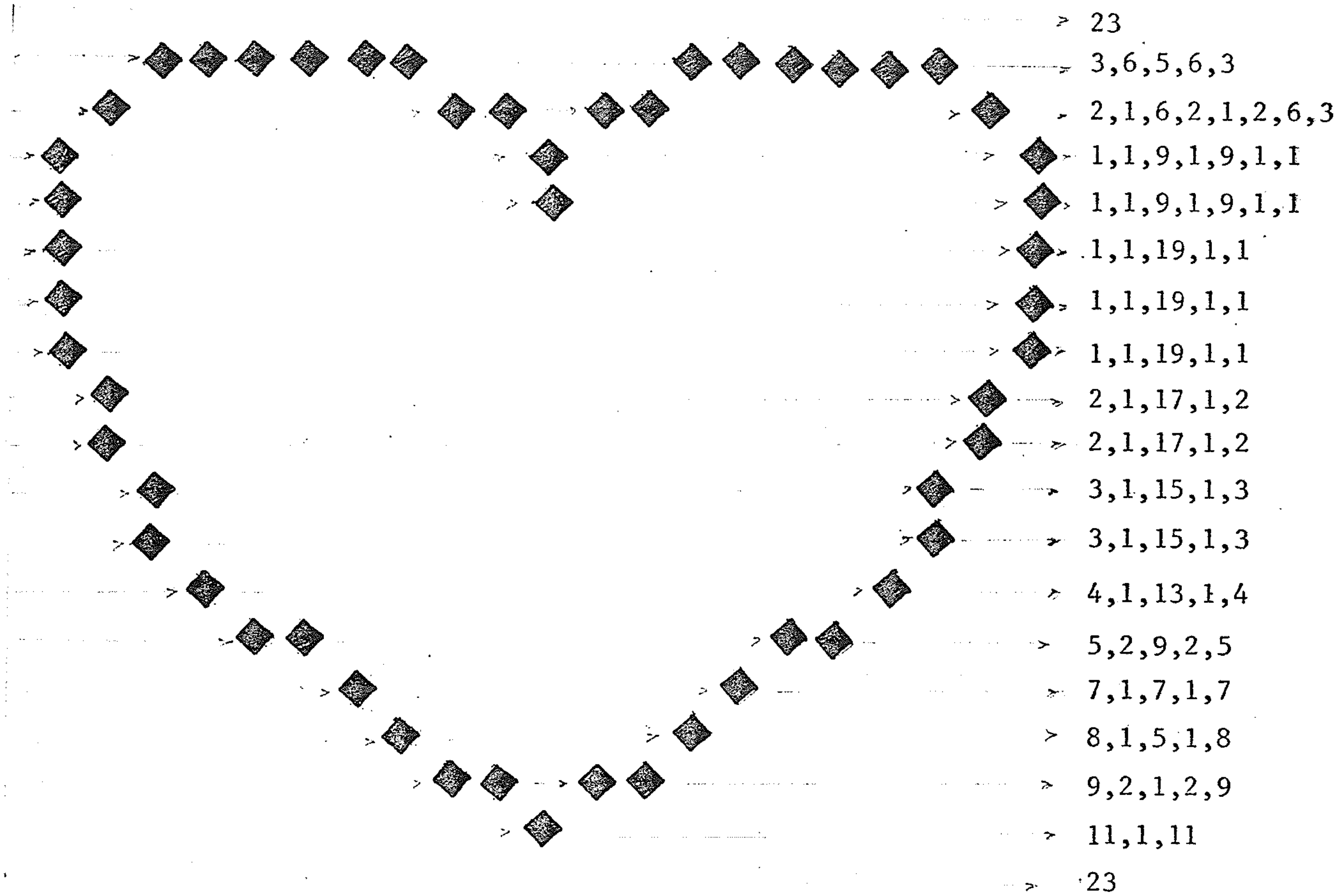


Fig. 3.18 Run Length

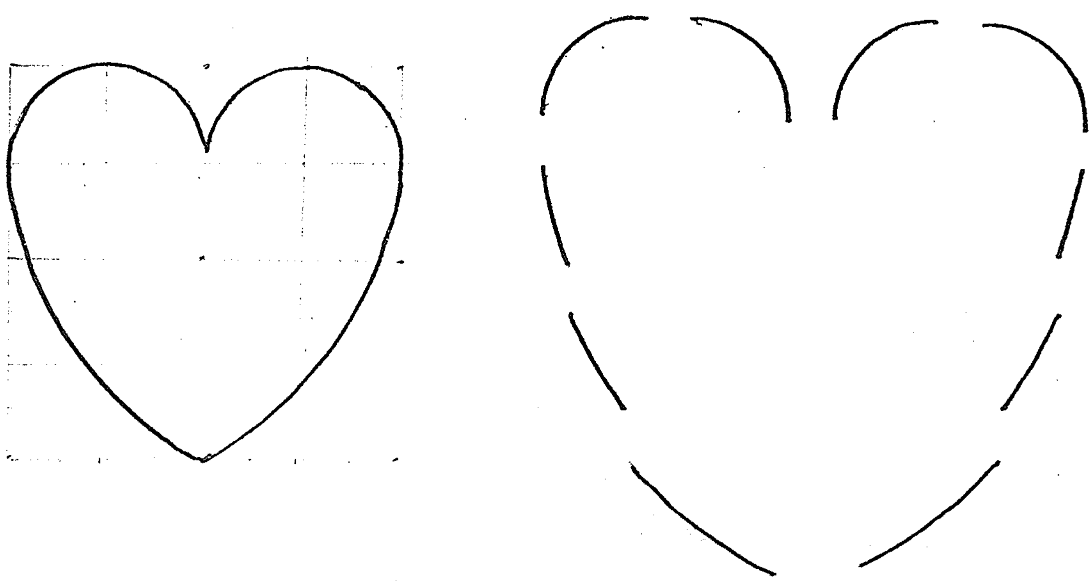


Fig. 3.19 Partial Symbol

3.7.2 Partial Symbol Method

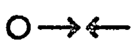
Storey broke the symbols down into sixty-four basic shapes or units, Figure 3.19 illustrates how a heart can be dismantled into the partial symbols.

Each unit or partial symbol requires eight bits, six bits for identification and two bits for overlay information. An additonal byte is required to position the unit vertically and horizontally. Table 3.1 indicates the amount of memory required to code the 25 Bliss Symbols. The partial symbol method appears to be an effective method to display the symbols.

However, the partial symbol method does have some disadvantages. A new symbol added to the existing set may not have the proper symbol shape(s) to fabricate it properly. Initial coding of the symbols requires tedious selection and fitting of the proper partial segments. For the above reasons and also because the path direction method is original, experiments in Chapter IV concentrate on its development. Some of the results of these experiments are shown in Figure 3.20.

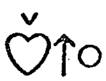
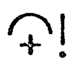
Table 3.1 Comparing Methods of Coding

(Resolution = 20 vertical grid lines)

SYMBOL	PATH DIRECTION (bytes)	RUN LENGTH (transitions)	(bytes)	PARTIAL SYMBOL (parts)	(bytes)
 friend	47	158	118	25	50
 day	20	80	60	14	28
 week-end	38	25	19	22	44
 car	33	106	80	24	48
 hello, goodbye	46	92	69	23	46
 afraid	60	209	156	26	52
 wheelchair	23	86	64	15	30
 angry	48	196	147	22	44

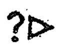
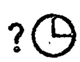
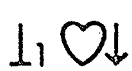
cont.

Table 3.1 (continued)

SYMBOL	PATH DIRECTION (bytes)	RUN LENGTH (transitions)	(bytes)	PARTIAL SYMBOL (parts)	(bytes)
 birthday	49	168	126	26	52
 question	16	58	43	8	16
 thanks	44	168	126	18	36
 funny	50	180	135	21	42
 good	37	100	75	17	34
 wash	19	27	20	11	22
 happy	40	120	80	16	32
 want	34	130	97	14	28
 teacher	48	176	132	29	58

cont.

Table 3.1 (continued)

SYMBOL	PATH DIRECTION (bytes)	RUN LENGTH (transitions)	(bytes)	PARTIAL SYMBOL (parts)	(bytes)
 think	17	40	30	6	12
 how	19	92	69	10	20
 why	19	80	60	10	20
 mouth	10	60	45	8	16
 help	19	92	69	13	26
 when	23	70	67	10	20
 please	26	78	58	13	26
 I'm sorry	48	198	148	24	48
TOTAL (bytes)	833		2093		850
BYTES/SYMBOL	33.3		83.7		34.0

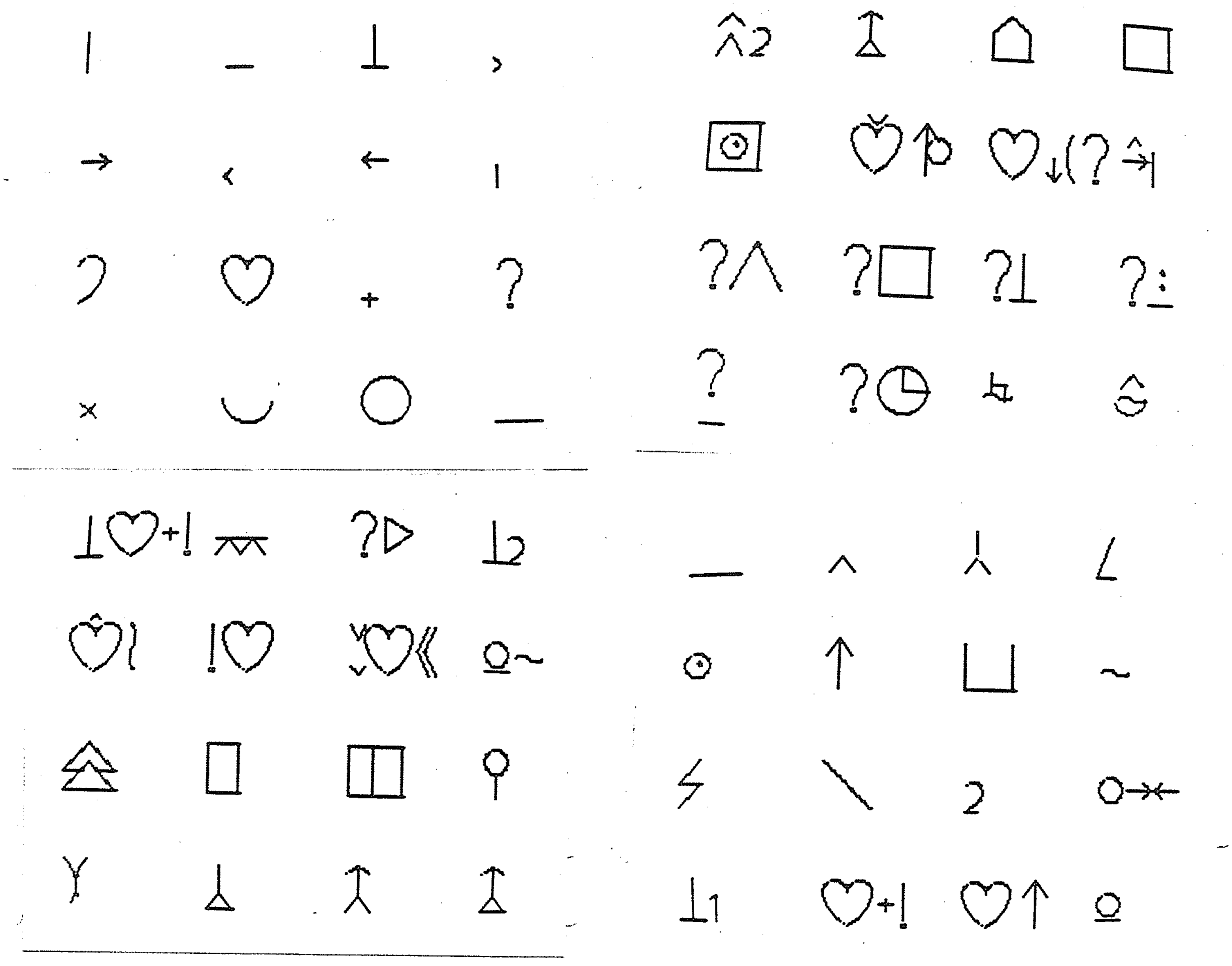


Fig. 3.20 Bliss Symbols as Displayed on Television (Vertical Resolution = 20)

CHAPTER IV

TEST PROGRAMS

4.1 Test Hardware

The test equipment (Fig. 4.1) consisted of an Imsai 8080 microprocessor with a Merlin Graphics Display and I/O (input/output) devices to communicate with the computer. The technical details and specifications are given in Appendix B.

The Merlin interface displays a dot on the television screen for every corresponding dot in memory; requiring 8K bytes (8,000 bytes of 8 bits) of memory. Merlin and the microprocessor share the memory and in this manner, the microprocessor can alter what is displayed.

This equipment was used to experiment with the first three methods as described in Chapter III. The primary purpose of this chapter is to verify the theory discussed earlier and perhaps identify unforseen areas of difficulty.

All programs were written in North Star Basic except the code decompiler of the path-direction method, which was written in 8080 ASSEMBLER.

Fig. 4.1 Experimental Equipment

4.2 Equations Describe the Symbol

'BLISS' was a program written to experiment with symbols by inputting equation parameters to construct and display Bliss Symbols with a large degree of flexibility.

The program first initializes pertinent data and variables and then requests the desired position of the symbol to be constructed. The program then requests equation parameters and displays all sample points. The sample point coordinates are stored in x,y, arrays and on the completion of a symbol are shifted until the x and y error is less than a predetermined value. The coordinates are then matched with a dot and a dot is designated if more than 'T' samples are assigned to the dot. This is equivalent to the smoothing technique described in Chapter III. After all the dots are displayed, the operator can scrutinize the symbol and record the equation parameters for posterity. The program then reinitializes and solicits input for another symbol. Refer to Figure 4.2, the flowchart, and Appendix C, the program listing, for further details.

One unforseen problem became apparent when the first circle was presented and it appeared egg-shaped. An aspect ratio factor 'F' was introduced to correct for this distortion. Also the 'C' and 'S' variables which determined the sample point density for circle and line segments respectively, required some trial and error in conjunction with the threshold

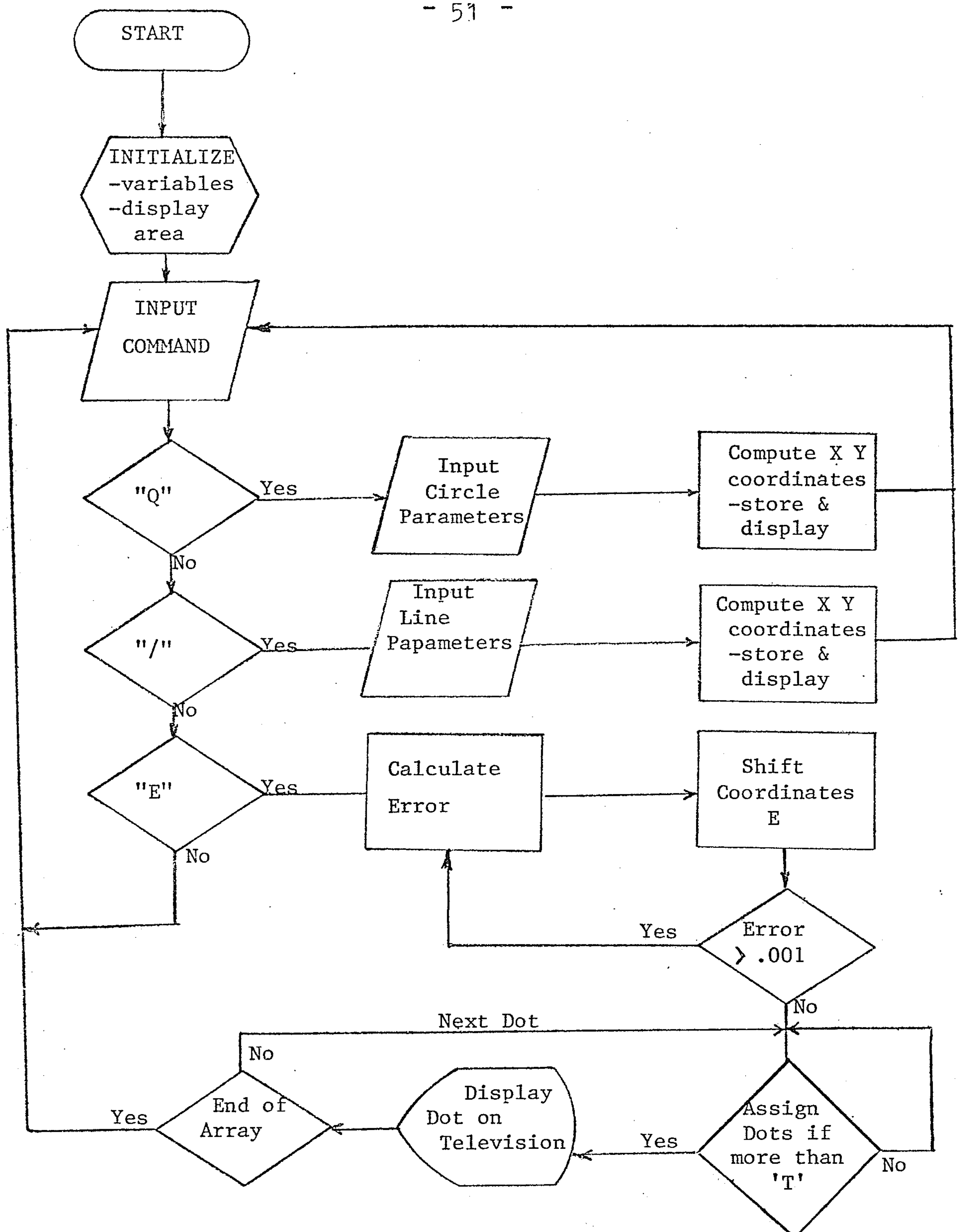


Fig. 4.2 Flow Chart of Bliss

variable 'T' to yield a satisfactory display of dots. Another small problem in the error routine occurred when the total error would not converge to less than the preset threshold and the computer would, therefore, loop in this program indefinitely. This was remedied by putting a limit of ten iterations on the error correction routine. The remainder of the program was straight forward.

'BLISS' was an ideal program to experiment with in varying parameters and variables. Before 'BLISS' was attempted, it was the author's belief that the equation parameters could be stored in a file and called upon to display the symbol. However, an unreasonable amount of time, measured in minutes, was necessary to compute and display each symbol, making it impractical. A solution to this dilemma would be to compute the x,y coordinates beforehand, thus alleviating the computer of a heavy workload at the time of presentation. This was the course of action pursued.

4.3 'BLSLD' - Constructing a Data File of Equation Parameters

The first requirement was to assemble a file of equation parameters for each symbol. The program that did this was 'BLSLD' and it loaded a string of data in the following format:

SYMBOL C x x x x C x x x C x x x x x E

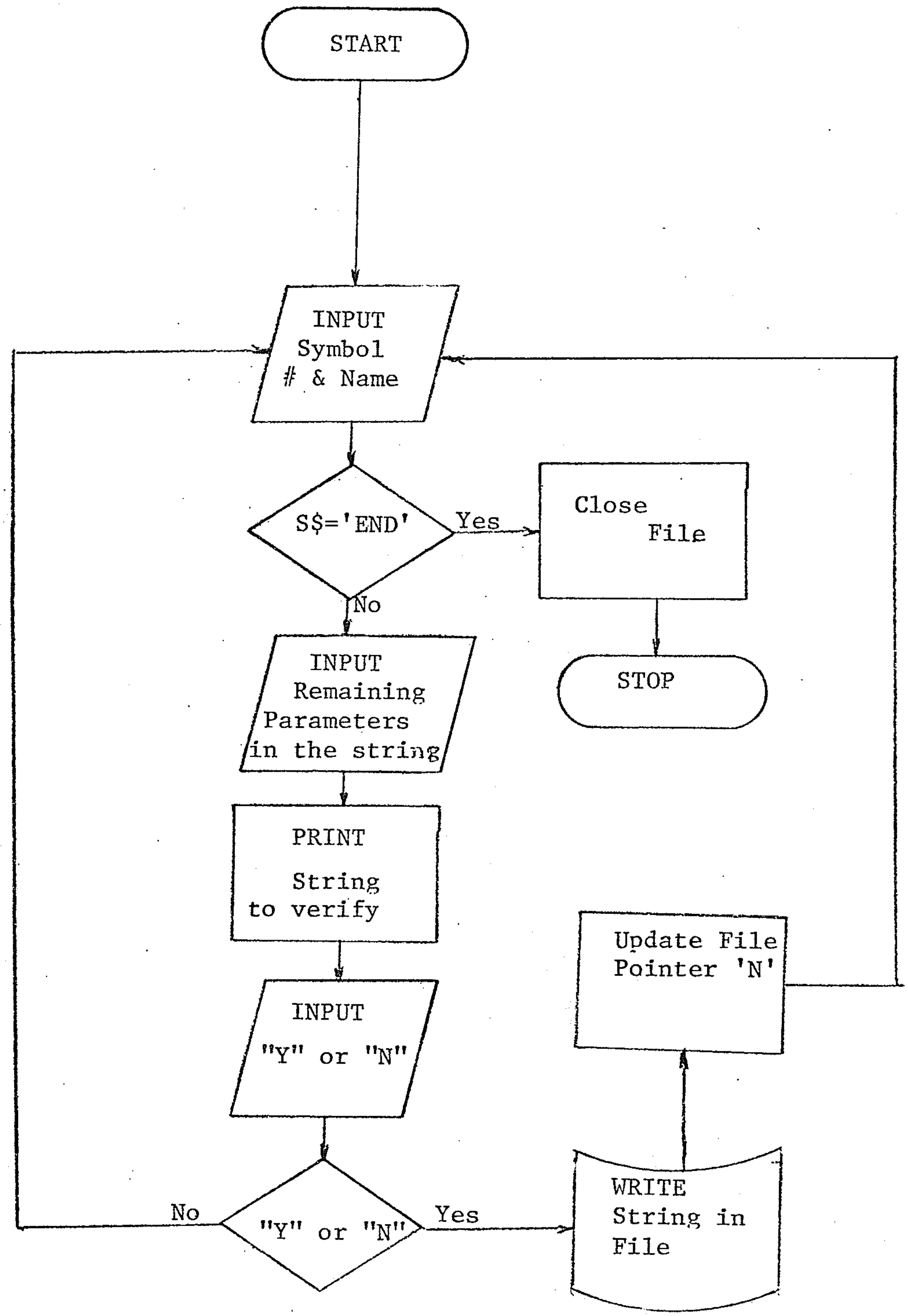


Fig. 4.3 Flow Chart of BLSLD

where: # - symbol number, used to identify a symbol when scanning

C - command

/ - line with four trailing arguments xo yo x1 y1

Q - circle, five arguments x y r A B

: - previous symbol, 3 arguments # x y

E - end delimiter

All the individual components of the string of dots were delimited with spaces.

A new command ':' meaning previous symbol was introduced here. The line and circle command, along with their respective arguments were previously discussed. The ':' has three succeeding arguments (# x y) the first of which was the symbol number that had previously been processed. The x and y were translation factors and they indicated how much the symbol would be translated in the x and y directions. This technique saved file space and computer time in not having to reprocess similar symbol shapes. This technique was a recommendation of Chapter III and it proved to be very satisfactory.

BLSLD solicited input which was entered as a long string. The data file it created was BLSDATA. Provisions were also made for correcting this BLSDATA file with a program called BLSEDIT. The flowchart for BLSEDIT is shown in Figure 4.4 and the program listing is in Appendix E.

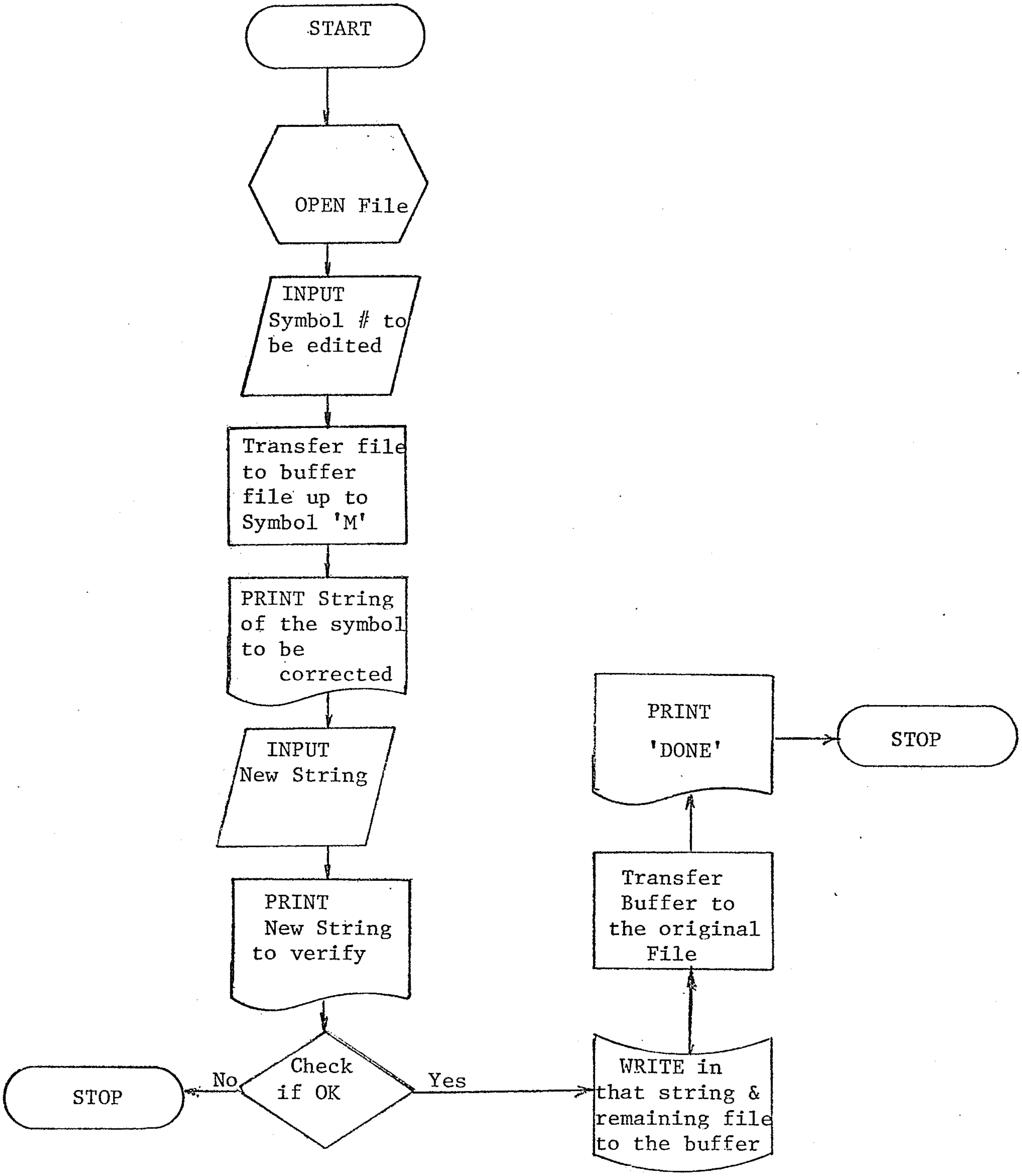


Fig. 4.4 Flow Chart of BLSEDIT

4.4 'BLSXY' - Computing the Dots Off-Line

BLSXY is a program very similar to BLISS. The difference lies in the input. BLSXY reads the BLSDATA file for input instead of entering it manually. Also the computed coordinates of the dots were stored in a file called BLSDXY instead of being presented on television. This file is then utilized in real time presentatation of the symbols. BLSXY was intended to be run independent of manual input, reading the data from one file and computing coordinates for another. A typical execution time of four or five hours is required to compute the coordinates for one hundred symbols with a height of twenty dot rows. The flowchart and program listing are in Figure 4.5 and Appendix F respectively.

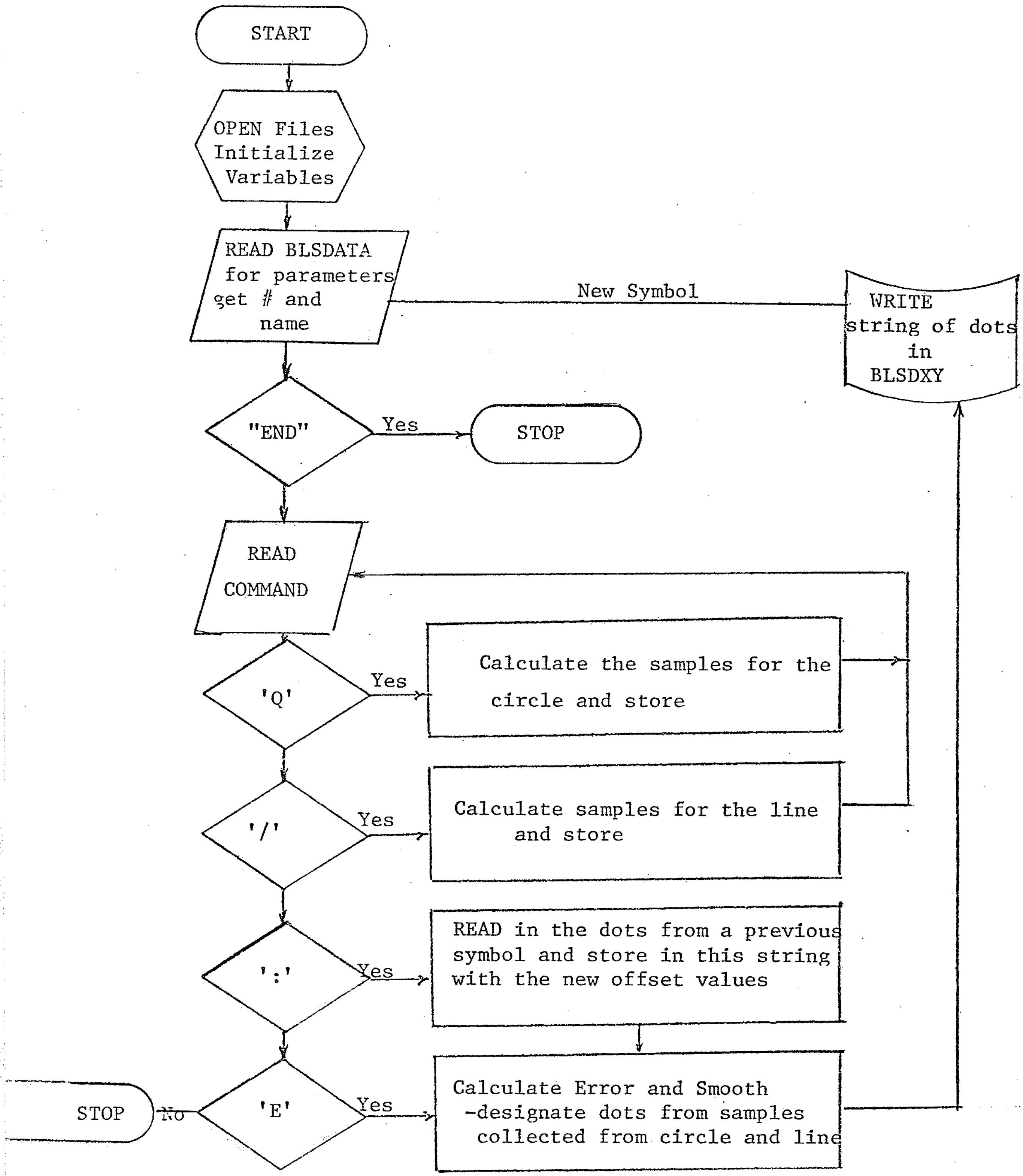


Fig. 4.5 Flow Chart of BLSXY

4.5 'DISPLAY - Read the File of Coordinates and Present the Symbol

DISPLAY is the real-time counterpart of BLSXY and it displays the coordinates which BLSXY computed and stored in BLSDXY. A symbol still requires a few seconds to be presented with the calculation of the width of the symbol occupying much of this time. Refer to Figure 4.6 and Appendix G for the flowchart and program of DISPLAY. The width could be added to the existing file to save some time but then this again requires more memory.

One can begin to see here the conflict between time and memory requirements. Undue delays in presenting the symbols lead to frustration, yet additional memory is a direct hardware expense. A compromise to partially satisfy both would be necessary unless another means can be devised to satisfy both. Method Three is an attempt to satisfy both of these requirements.

4.6 'BLSCODE' - Compute Code for Path Direction Method

BLSCODE compiles a code for the path direction method as described in Chapter III. For details refer to the flowchart in Figure 4.7 and the listing in Appendix H. The code does not require extensive memory and it is stored in core memory.

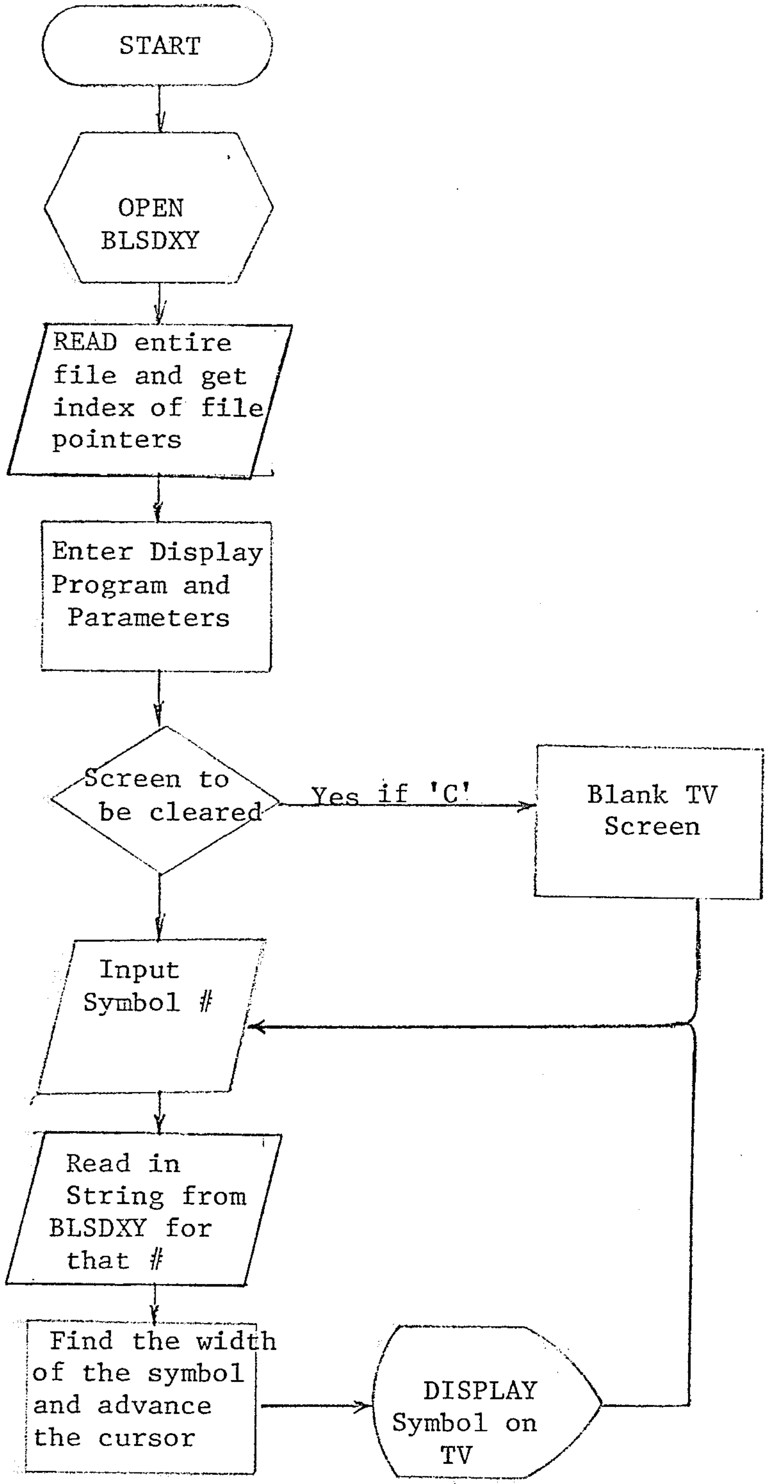


Fig. 4.6 Flow Chart of DISPLAY

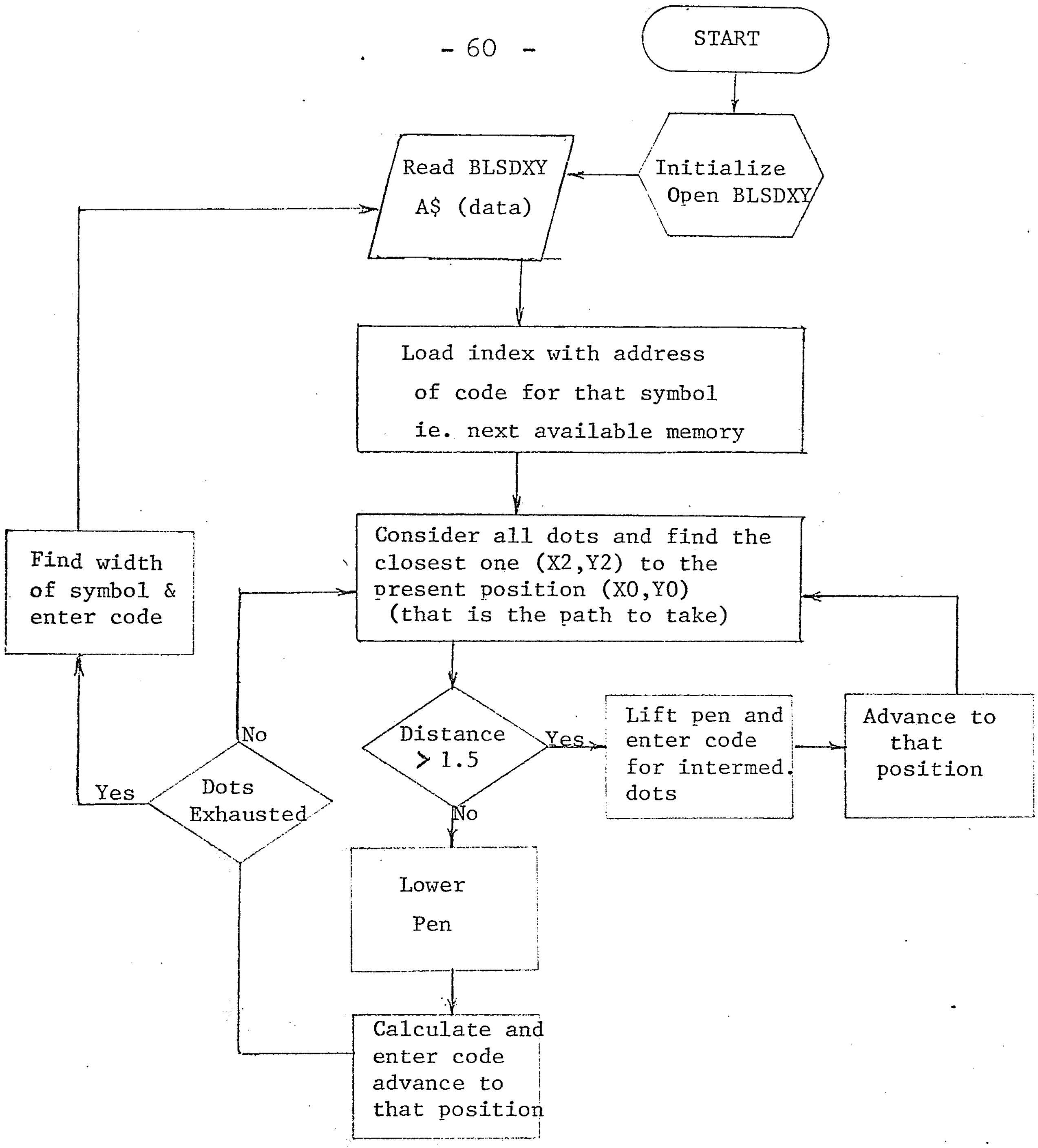


Fig. 4.7 Flow Chart of BLSCODE

A few notes on the program will clarify some points. For convenience of decompiling the width is added at the end of the code; this was not done previously in BLSXY. Also notice that an index is compiled, indicating the absolute memory location of the code for each symbol. The symbol number determines the offset in the index table which in turn contains addresses of the code. The 2-bit code is entered into a byte from most to least significant side. When a byte contains four codes the next byte is started. A new symbol always starts on a new byte irregardless of whether there was any remaining space left in the previous byte.

Again, this program was designed to be run in an off-line mode, taking hours to execute a set of one hundred symbols. The program reads the dot coordinates from the BLSDXY file and enters the code in an unused portion of core memory.

4.7 Path Direction Decompiler

The objective of method three was to minimize the memory requirements, and a method was devised to shrink the requirements for the symbol data, but the program to manipulate this code also requires memory. This program will be an essential element of the Bliss Symbol Communicator. The off-line program requirements of memory and time were not critical.

These resources were only initially required to develop the software for this system. However, the length of program used for decompiling the code and presenting it is important. For this reason, this program was written in assembler language. Although writing a computer program in assembler language is more difficult to both write and understand; it is the best means to minimize memory requirements.

The program first initializes variables, sets and clears the display area and then scans the eight output light emitting diodes (output port FF) located on the front panel of the computer. The LEDs simulate a scan board, each light representing a different symbol. Touching any key on the keyboard freezes the number currently in the scan register and the program gets the address (SYMB) of the first code from TABLE (OFOO -). The flow chart and program listing are in Figure 4.8 and Appendix I respectively.

The program then goes to DIREC and gets two bits (BITS), decodes them and executes them according to the command they contain.

Figure 4.9 has an example to help clarify the operations of the program. The code is as follows:

Hex	Binary
CF	11001111
FF	11111111
2F	00101111

A2	10100010
F0	11110000
05	00000101

During initialization X is set to minus one and Y to zero with the direction register set to zero. The first two bits (11) mean straight ahead, so the cursor moves to coordinates (0,0), but since the pen is up, nothing is displayed. The next two bits (0,0) indicate that secondary bits are required and they being (11) for a combined value of 0011 mean 'lower the pen'. The next five 11's display a horizontal base line with the direction register all this time being zero. The next 0010 adds two to the current value of the direction register. The next 0010 adds two to the direction to turn 90 left (counterclockwise) or in this case up, and the dot is displayed. The next two sets of 11's display two additional dots vertically and then the two 10's shift the direction left. The direction register is now horizontal, pointing left, but the next 0010 brings it to a downward direction for the remainder of the symbol. The three sets of zeroes (00,00,00) indicate the end of the symbol in binary; with the remaining six bits representing the width of the symbol in binary; which in this case is five.

A few of Merlin's operating systems subroutines were used to display the dot, clear the screen, etc. All of these subroutines are located at Cxxx (Hex) in Merlin's MEI rom.

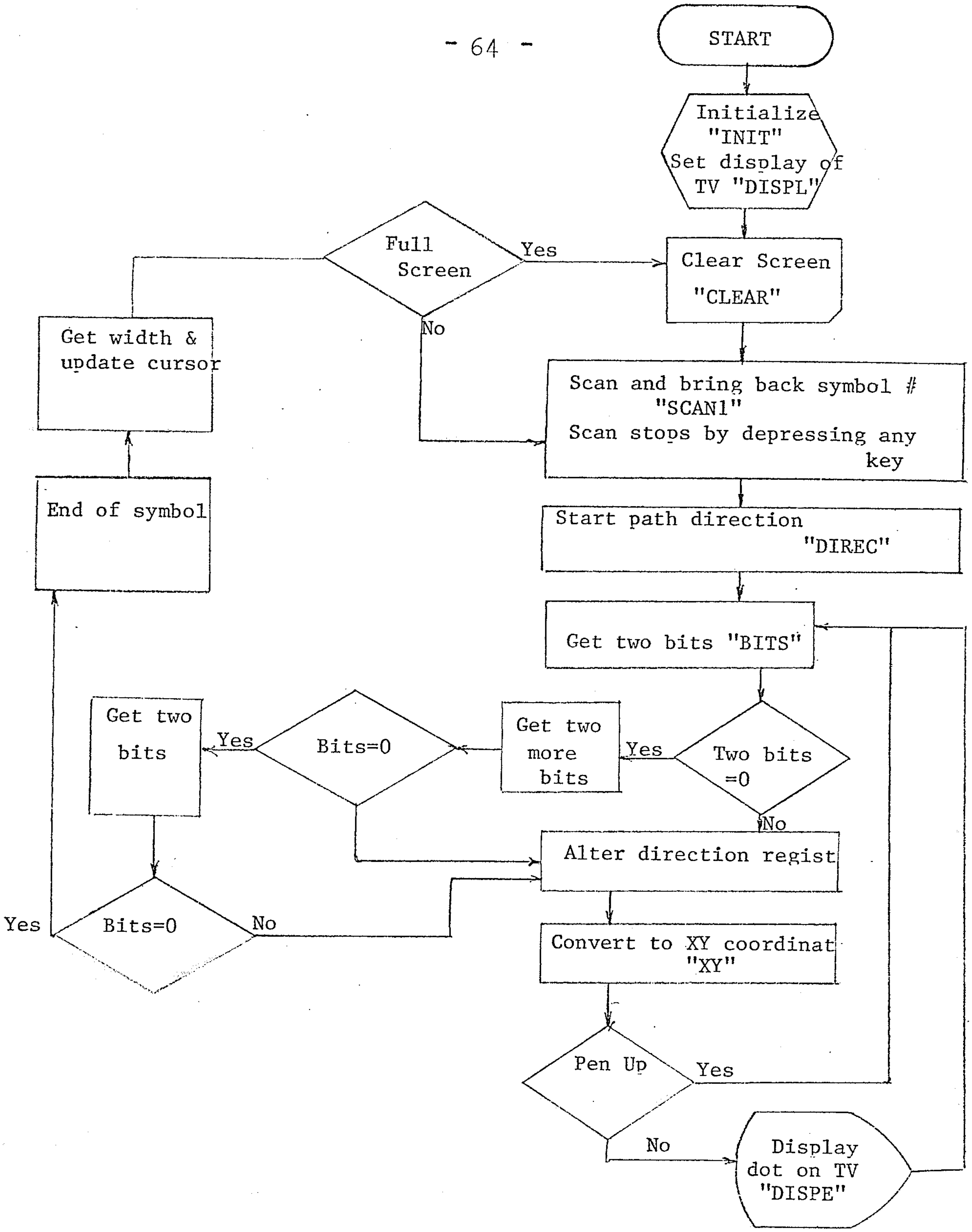


Fig. 4.8 Flow Chart of Path Direction Decompiler

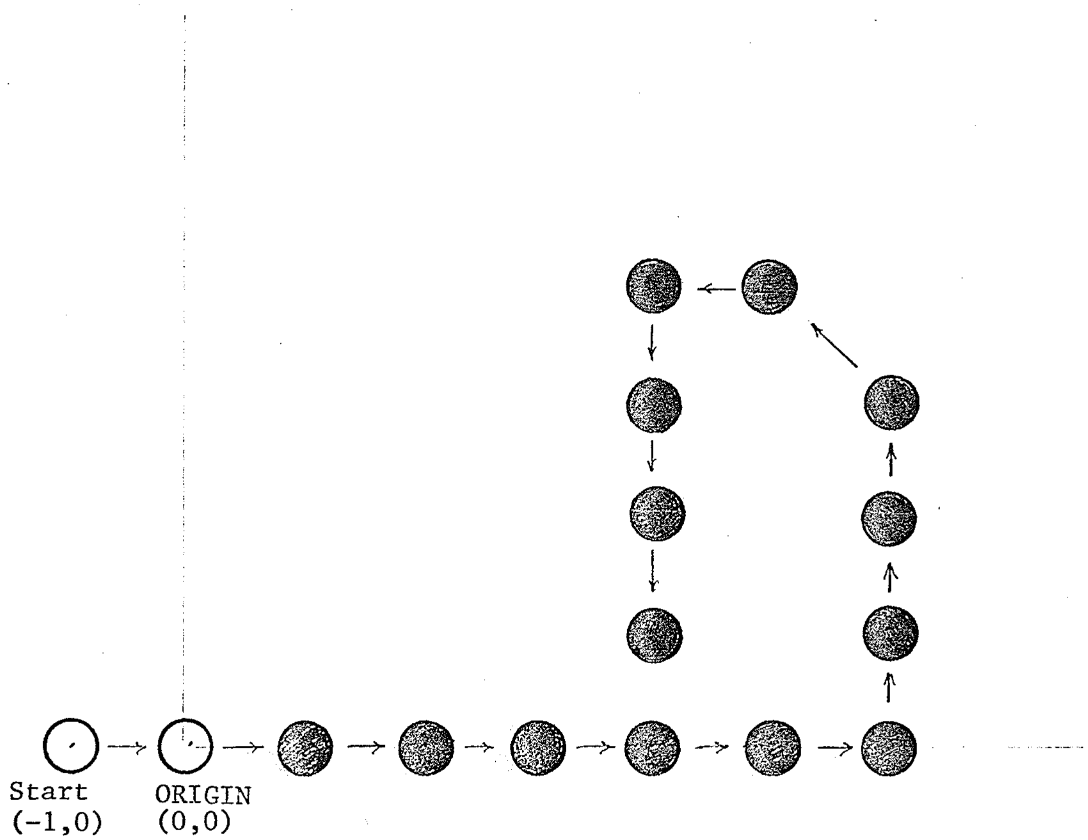


Fig. 4.9 Example of Path Direction Method

The path direction method code was developed empirically. Given the knowledge of the probabilities of the commands, one can code this optimally as suggested by Huffman. Optimallity refers to the average number of bits per code required. Thus the probabilities of the different codes were calculated and the Huffman code was determined. The results are shown below:

Meaning	Authors's Code	Percent Occurrence	Relative Size	Huffman Code	Relative Size
Straight	11	68.5	137	1	.685
Left One	10	10.9	21.8	000	.327
Right One	01	8.1	16.2	001	.243
Pen Down	0011	3.3	13.2	0101	.132
Left Two	0010	2.7	10.8	0110	.108
Right Two	0001	.8	3.2	010010	.048
End	000000	2.4	14.4	0111	.096
Pen Up	000011	1.4	8.4	010000	.084
Right Three	000001	.9	5.4	010001	.054
Left Three	000010	.7	4.2	010011	.042
			2.346 bits/code		1.819 bits/code

By implementing the Huffman code, a 20 percent saving in memory storage could be realized. However, since this code would be more difficult to compile and decompile, no attempt was made to implement it. Future endeavours should consider it.

4.8 Review and Results

Figure 4.10 presents in chronological sequence a review of the major developments of the experiments. Initially equation parameters were tabulated and they became an excellent standard in displaying Bliss Symbols but they were inadequate for displaying the symbols directly because of time limitations. A second approach was attempted in computing the coordinates of the dots to be displayed but still it required a fair amount of time and memory to present the symbol. Then the path direction method was attempted and it solved these problems. Quantitative results of memory and time for the three methods are given in tables 4.1 and 4.2 respectively. From these tables it can be concluded that the path direction method is a feasible technique to display Bliss Symbols.

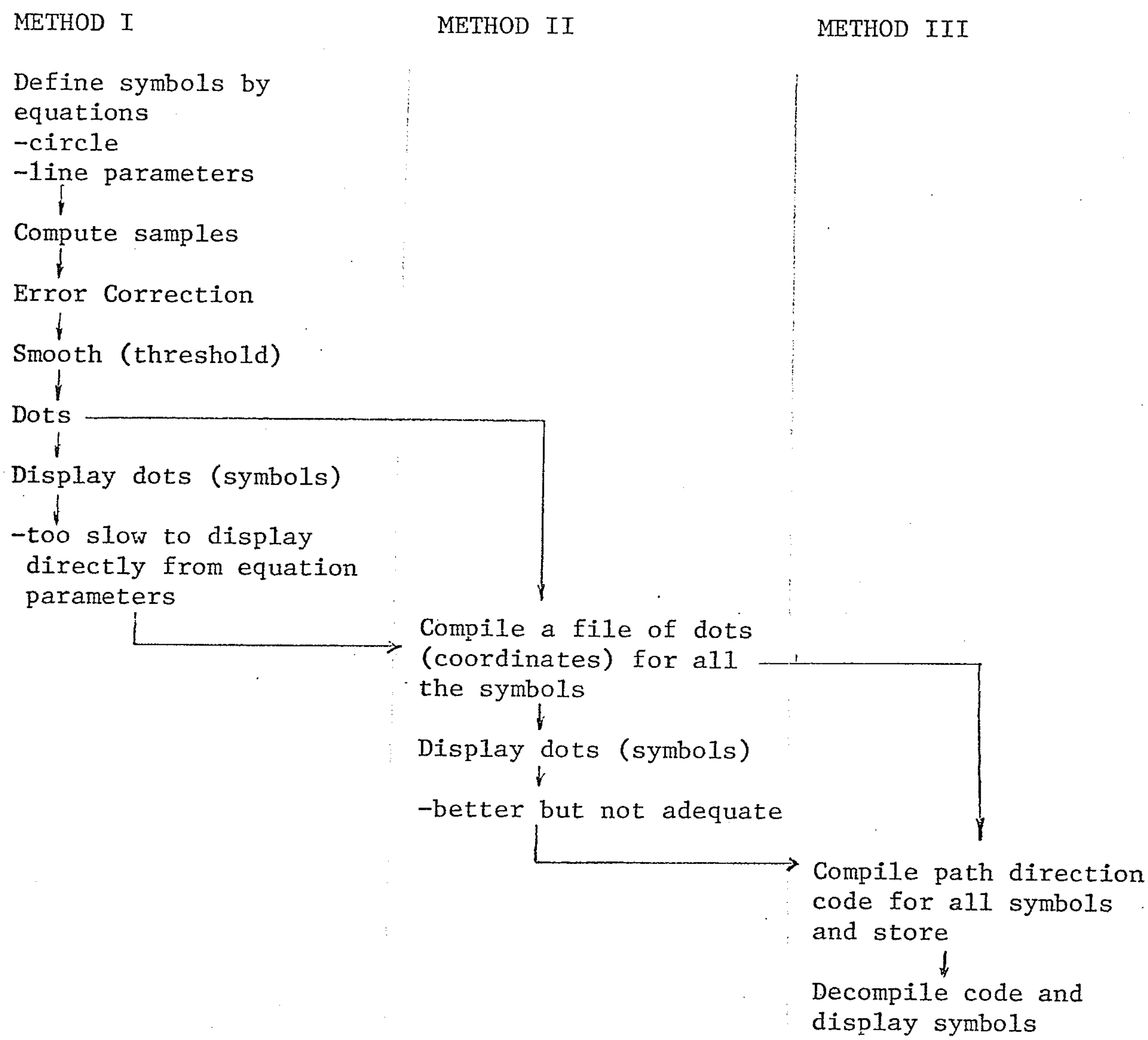


Fig. 4.10 Review of Sequential Events

Method	Basic Language	Program	Variables	Code	Total
1. Equations Directly	11.5 K (bytes)	2.2 K	10 K	6.2 K	29.9 K
2. Reading the Coordinates	11.5 K	1.9 K	1.1 K	35.4 K	49.9 K
3. Path Direction	______	.75 K	0 K	3.2 K	4 K

Table 4.1 Memory Comparison

Method	Off-Line Execution Time	Display Time
1. Equations Directly	0	30-100 sec.
2. Reading the Coordinates	4 - 5 hrs./ 100 symb.	3 sec./ symb.
3. Path Direction	4 - 5 hrs/100 symbols	.3 sec./symb.

Table 4.2 Time Comparison

CHAPTER V

CONCLUSIONS AND RECOMMENDATIONS

This thesis has been concerned with finding efficient techniques to display Bliss Symbols on television. The results of Chapter III indicated that the path direction method was the most favourable. Chapter IV presented programs to transform the Bliss Symbols from equation descriptors to the path direction code. These investigations and experiments demonstrated the feasibility of displaying Bliss Symbols on television.

Original contributions of this thesis in the author's opinion are:

1. The standardization of Bliss Symbols employing equation descriptors.

2. Error minimization to represent a symbol on a discrete array.

3. Path direction method to code Bliss Symbols efficiently.

Some topics that warrant further investigation are:

1. It may be obvious that the path direction method is intended to present a path that is one dot wide. To have a path wider than one dot, as would be required for easy viewing of large symbols, would be awkward with this method directly because of the continuous turning to set the dots.

A wider line could be attained by displaying two dots simultaneously making the line thicker with no additional code required. This should be investigated further.

2. Try the path direction method with other graphics, handwriting, etc.

3. Try other codes, possibly allowing only counter clockwise turns; such as:

1 - straight ahead

01 - counterclockwise turn

001 - change pen mode

Appendix A

TABLE OF BLISS SYMBOLS

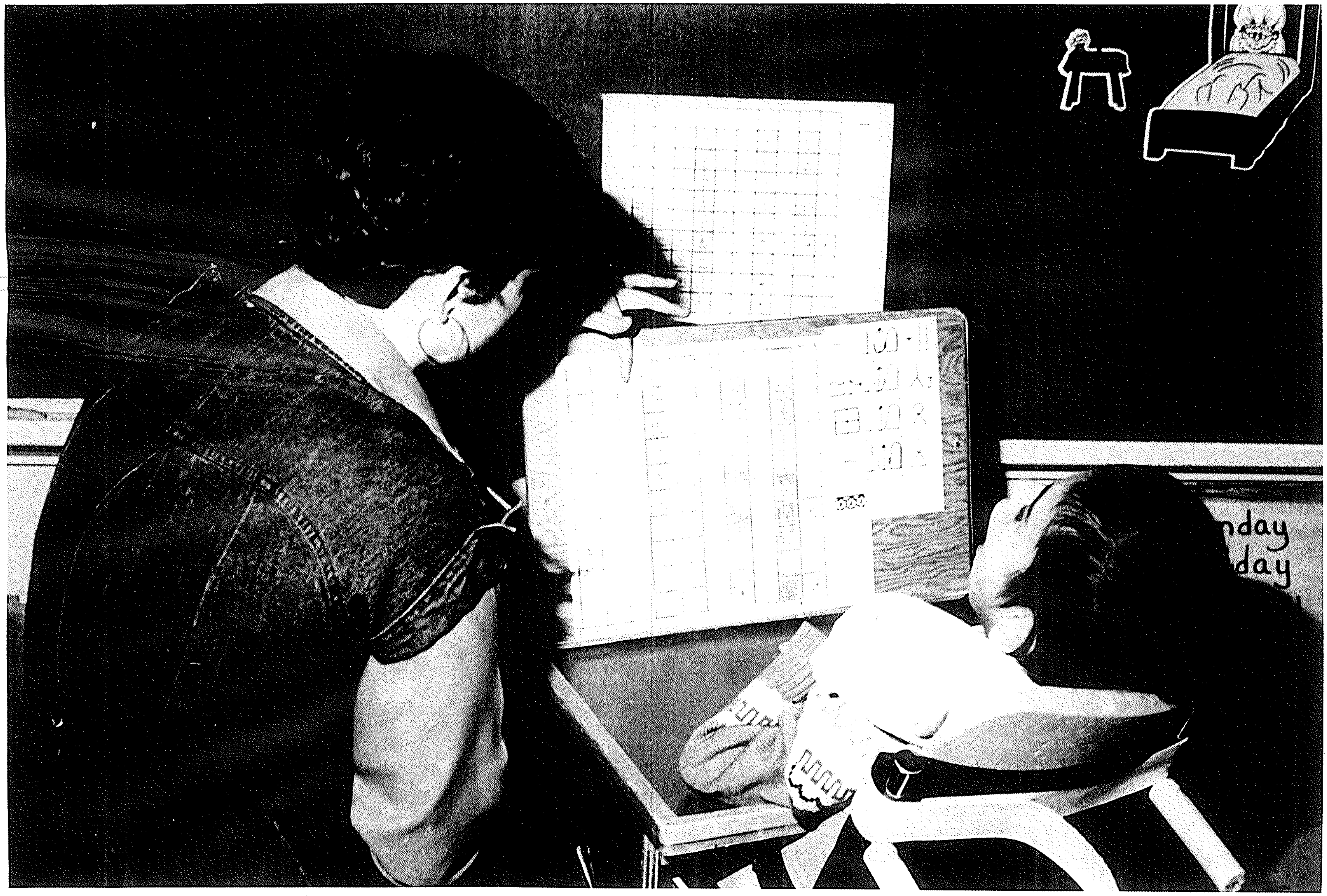
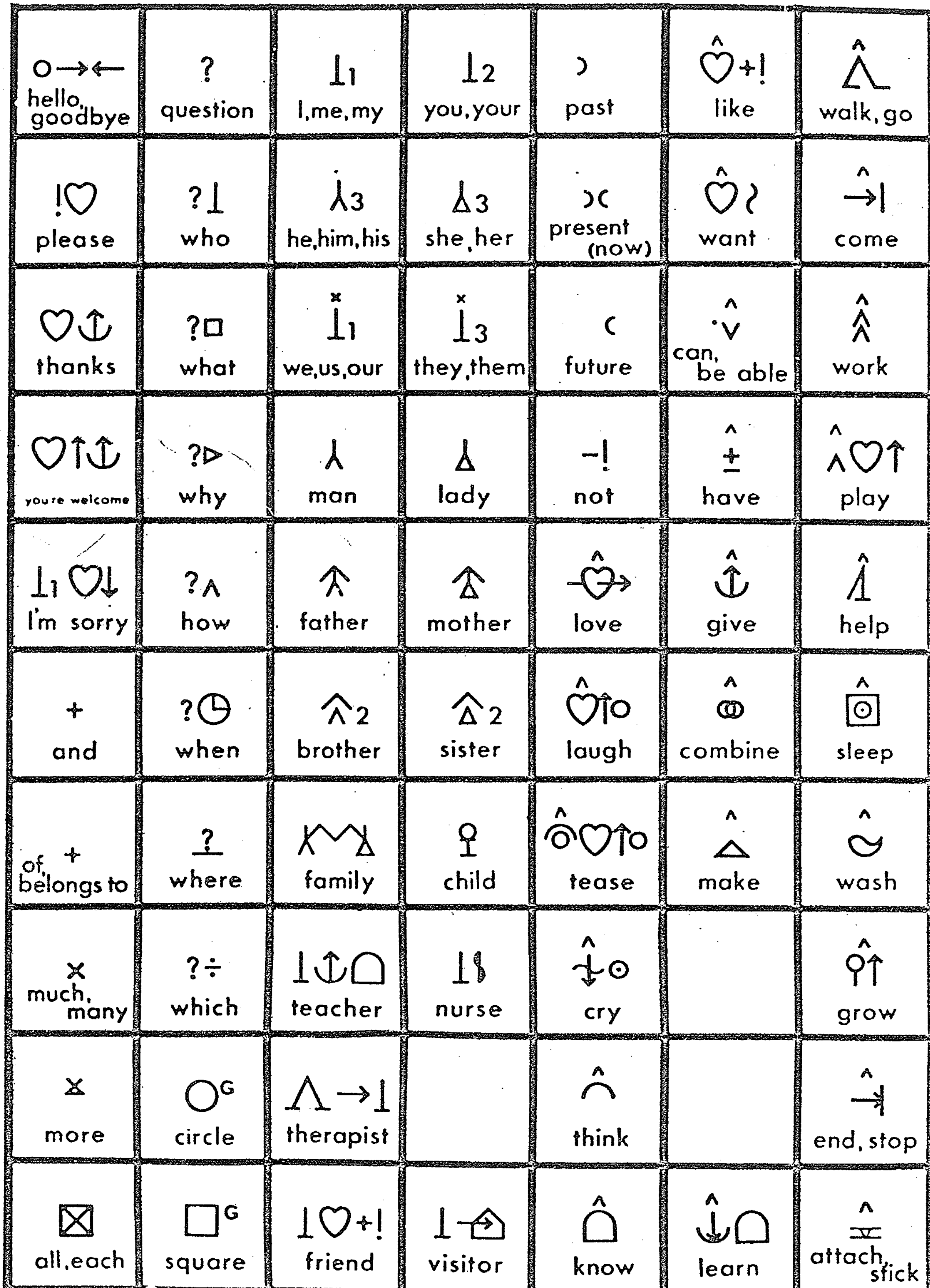


Fig. A.1 Teacher and Child using Bliss Symbols


hello, goodbye
question
I, me, my
you, your
past
like
walk, go
please
who
he, him, his
she, her
present (now)
want
come
thanks
what
we, us, our
they, them
future
can, be able
work
you're welcome
why
man
lady
not
have
play
I'm sorry
how
father
mother
love
give
help
and
when
brother
sister
laugh
combine
sleep
of, belongs to
where
family
child
tease
make
wash
much, many
which
teacher
nurse
cry
grow
more
circle
therapist
think
end, stop
all, each
square
friend
visitor
know
learn
attach, stick

symbol	happy	angry	0	1	2	3
part	excited	afraid	do, make action	eye	colour	ABC
like, similar	funny	lucky	food	mouth	thing	pen, pencil
opposite	good	special	drink	legs	tool	paper, page
in	big	young, new	clothing	hand	container	book
on	long	full	toilet	ear	strap, string	letter
to	heavy	hot	pain	nose	vegetable	story
up	dirty	soft	make-believe	teeth	bread	news
before	quick		broken	head	fruit	word
here	open	difficult	dead	body	milk	name

4	5	6	7	8	9	
wheelchair	chair	home	GOD	night	day	U u
car	table	school	animal	sun	yesterday	V v
bus	bed	show	bird	weather	today	W w
airplane	door	hospital	fish	rain	tomorrow	X x
boat	window	street	flower	snow	time	Y y
money	room	store	tree	earth	morning	Z z
gym	floor	light	country	sky	afternoon	
music	wall	telephone	outing	water	week-end	
art	cupboard	television	gathering	fire	birthday	

Appendix B

SPECIFICATIONS ON MERLIN AND OTHER COMPONENTS

Appendix B

Technical Details of Experimental Equipment

Mainframe - IMSAI [1]
- Front Panel - 44 LED's, 22 switches
- Motherboard - 22 slot, S100
- Power Supply - 18 amps ±12 volts and +5 volts

Mainframe Cards
- Microprocessor or Central Processor Unit
  - IMSAI
  - 8080A
  - 2 phase, 2 Megahertz clock

- RAM 4A
  - IMSAI
  - 3 x 4 K byte boards

- RAM Godbout [2]
  - 3 x 8 K byte boards

- Serial I/O
  - RS232 output ports 2 and 3 for VUCOM I terminal, and printer
  - I/O for Disk Operating System, Basic, Assembler, etc.
  - IMSAI

- Merlin - Double Card
  - Miniterm Associates Inc. [3]
  - Graphics Interface for 12" B & W portable Fleetwood Television
  - Options - Superdense 320 x 200
    - MBI, MEI

- Floppy Disk Card - North Star [4]
  - Interfaces Floppy

Minifloppy - North Star
- With power supply
- 100 K bytes/disk
- Used North Star Basic and Disk Operating System

Keyboard - Radio Shack [5]
- ASCII Encoder Keyboard #277=117
- Plugs into Merlin Board to operate Merlin's Operating System.

1. IMSAI - IMS ASSOCIATES, INC.
   14860 Wicks Boulevard
   San Leandro, Ca. 94577

2. GODBOUT - BILL GODBOUT ELECTRONICS
   Box 2355
   Oakland Airport, Ca. 94614

3. MERLIN - MINITERM ASSOCIATES, INC.
   Dundee Park
   Andover, Ma. 01810

4. NORTH STAR COMPUTERS INC.
   2547 Ninth Street
   Berkeley, Ca. 94710

5. RADIO SHACK - A DIVISION OF TANDY CORPORATION
   Barrie, Ont. Fort Worth, Texas
   Canada L4M 4W5 U.S.A. 76102

Appendix C

BLISS LISTING

```
15 REM PROGRAM TO EXPERIMENT WITH BLISS SYMBOLS
20 DIM X(500),Y(500)\REM BLISS FEB 5
30 N=53361
40 FOR I=0 TO 8
50 READ A
60 FILL N,A
70 N=N+1\NEXT I
71 IF N<53361 THEN 90
72 N=53300
73 RESTORE 85\GOTO 40
80 DATA 58,121,208,235,182,119,235,201
85 DATA 0,0,0,0,0,0,0,10,17
90 Z=CALL(50099)\REM FLIP SCREEN
100 F=1.4\REM ASPECT RATIO
110 T=1\REM THRESHOLD
120 H=22
130 C=2\REM # OF DEGREES/SAMPLE
140 S=2.7
150 INPUT "SYMBOL COORDINATES? X0,Y0: ";X0,Y0
160 N=0
170 INPUT "COMMAND /QEC ";C$
180 IF C$="C" THEN 990
190 IF C$="Q" THEN GOSUB 530
200 IF C$="/" THEN GOSUB 630
210 IF C$="E" THEN GOSUB 1030 ELSE GOTO 170
220 GOTO 150\REM START A NEW SYMBOL
530 REM CIRCLE ROUTINE
540 INPUT "X Y RADIUS START OF ARC END OF ARC:",X1,Y1,R,A,B
550 A=A*6.282/360
560 B=B*6.282/360
570 X=(X1+R*COS(A))*H/10*F
580 Y=(Y1+R*SIN(A))*H/10
590 A=A+C*6.282/360*5/R
600 GOSUB 890
610 IF A<B THEN 570
620 GOTO 170
630 REM LINE ROUTINE
640 INPUT"X0 Y0 X1 Y1: ",X3,Y3,X4,Y4
650 IF X3=X4 THEN 830
660 IF X3<X4 THEN 730
670 X7=X3\X3=X4\X4=X7
700 Y7=Y3\Y3=Y4\Y4=Y7
730 M=(Y4-Y3)/(X4-X3)
740 X5=SQRT(1/S↑2/(1+M↑2))*10/H
750 B=Y3-M*X3
760 IF M=0 THEN B=Y3
770 Y=(M*X3+B)*H/10
780 X=X3*F*H/10
790 GOSUB 890
800 X3=X3+X5
810 IF X3<X4 THEN 770
820 GOTO 170
830 X=X3*F*H/10
840 Y=Y3*H/10
850 GOSUB 890
```

```
860 Y3=Y3+10/(H*S)
870 IF Y3<Y4 THEN 840
880 GOTO 170
890 N=N+1\Y(N)=Y\X(N)=X\REM OUTPUT CIRCLE-LINE
900 X=INT(X+X0)\Y=INT(Y+Y0)\REM NORMAL OUTPUT
910 A1=INT(X/8)+INT(199.5-Y)*40
920 B1=X-INT(X/8)*8
930 FOR J=0 TO 7
940 IF INT(B1)=J THEN C1=2↑J
950 NEXT J
960 FILL 53369,C1
970 Z=CALL(53361,A1)
980 RETURN
990 FOR A=0 TO 4096\REM CLEAR SCREEN ROUTINE
1000 FILL A,0
1010 NEXT A
1020 GOTO 150
1030 REM ERROR ROUTINE
1040 E0=0\E1=0
1050 FOR K=0 TO N
1060 E0=E0+X(K)-INT(X(K)+.5)
1070 E1=E1+Y(K)-INT(Y(K)+.5)
1080 NEXT K
1090 FOR K=0 TO N
1100 X(K)=X(K)-E0/N
1110 Y(K)=Y(K)-E1/N
1120 NEXT K
1130 PRINT "EX,EY: ";E0/N;E1/N
1140 IF ABS(E0/N)+ABS(E1/N)>.0015 THEN 1030
1150 PRINT "# OF SAMPLES IN THIS SYMBOL: ";N
1170 FOR K=0 TO N\REM THRESHOLD ROUTINE
1175 X=X(K)\Y=Y(K)-H-10\GOSUB 900
1180 X(K)=1000*INT(X(K)+.5)+INT(Y(K)+.5)
1190 NEXT K
1200 N1=N\N=-1
1210 FOR L=0 TO N1
1220 IF X(L)<>INT(X(L)) THEN 1300
1230 FOR K=L+1 TO N1
1240 IF INT (X(L))<>X(K) THEN 1280
1250 X(L)=X(L)+.01
1260 X(K)=X(K)+.01
1280 NEXT K
1290 IF X(L)-INT(X(L))>=T*.01 THEN GOSUB 1330
1300 NEXT L
1310 PRINT "# OF DOTS = ";N+1
1320 RETURN
1330 N=N+1\REM PREPARE FOR ERROR CORRECTED DOT OUTPUT
1335 Y(L)=X(L)
1340 X(N)=INT(X(L)/1000)\X=X(N)
1350 Y(N)=INT(Y(L)-1000*X(N))\Y=Y(N)-2*H-20
1360 GOTO 900
1470 RETURN
1500 END
```

Appendix D

BLSLD - BLSDATA

```
10 REM BLSLD DEC 31
20 REM THIS PROGRAM LOADS THE EQUATION PARAMETERS AND
30 REM PUTS IT INTO "BLSDATA"IN FOLLOWING FORMAT
40 REM "# SYMBOL-NAME 0 X Y R A B / X Y X Y :S X Y E "
50 REM ALL VALUES AND CODES ARE PADDED WITH SPACES
60 REM SYMBOL MUST NOT HAVE SPACES IN IT
70 REM MUST END IN AN "E"
80 REM MULTIPLE LINES DETECTED IF E NOT PRESENT
90 REM S MUST BE A SYMBOL ALREADY PROCESSED
95 N=4007 \REM N IS THE FILE POINTER - CAN BE SET TO ANY VALUE
96 REM STARTING WITH ZERO---ALL STRINGS ARE LEN + 2
100 DIM B$(90)
101 DIM S$(30)
102 DIM A$(500)
105 OPEN #0, "BLSDATA"
110 INPUT "SYMBOL #: ", S
120 INPUT "SYMBOL NAME: ", S$ \A$=STR$(S)+" "+S$+" "
125 IF S$="END" THEN 180
130 INPUT B$ \A$=A$+B$
140 IF B$(LEN(B$))="E" THEN 160
150 GOTO 130
160 PRINT A$
162 INPUT "IS THIS OK: Y OR N", N$
165 IF N$="N" THEN 110
167 WRITE #0 %N,A$
168 N=N+LEN(A$)+2
169 PRINT"FILE POINTER ";N \PRINT
170 GOTO 110
180 CLOSE #0
190 STOP
READY
```

```
0 LEFT-HEART Q 2.5 7.5 2.5 0 181 Q 10 8.8 10.1 188 240 E
1 VERT-LINE / 0 0 0 10 E
2 HORZ-LINES / 0 0 5 0 E
3 PERSON : 2 0 0 : 1 2.5 0 E
4 POINT> / 0 0 1.5 1.5 / 1.5 1.5 0 3 E
5 RIGHT : 4 3.5 3.5 : 2 0 5 E
6 POINT< / 1.5 0 0 1.5 / 0 1.5 1.5 3 E
7 LEFT : 6 0 3.5 : 2 0 5 E
8 VERT-LINES / 0 0 0 5 E
9 EXCLAMATION / 0 1.5 0 10 Q 0 .25 .5 0 350 E
10 DOT Q .4 .4 .4 0 350 E
11 POINT↑ / 0 0 1 1 / 1 1 2 0 E
12 POINT-DOWN / 1 0 0 1 / 1 0 2 1 E
13 ACCENT / 0 1.5 1.5 0 / 1.5 0 3 1.5 E
14 DOWN : 13 0 0 : 8 1.5 0 E
15 CAP Q 5 5 5 0 180 E
16 MOUTH Q 2.5 5 2.5 0 355 E
17 EAR Q 2.5 7.5 2.5 -2 180 Q -5 8.8 10.1 -61 -8 E
18 HEART : 17 5 0 : 0 0 0 E
19 AND / 0 1.5 3 1.5 / 1.5 0 1.5 3 E
20 ? Q 2.5 7.5 2.5 -45 170 Q 8.5 1.5 6 135 180 Q 2.5 .3 .4 0 350 E
21 MUCH-MANY / 0 0 3 3 / 0 3 3 0 E
22 CONTAINER Q 5 5 5 180 360 E
23 SUN Q 5 5 5 0 355 E
24 EARTH / 0 0 10 0 E
25 DO-MAKE / 0 0 2.5 3.5 / 2.5 3.5 5 0 E
26 MAN / 0 0 2.5 3.5 / 2.5 3.5 5 0 / 2.5 3.5 2.5 10 E
27 NOSE / 0 0 4 0 / 0 0 4 10 E
28 EYE : 16 0 0 : 10 2.5 5 E
29 UP : 1 2.5 0 : 25 0 7 E
30 OPEN / 0 0 10 0 / 0 0 0 10 / 10 0 10 10 E
31 WATER Q 2 3 2.12 45 135 Q 5 6 2.12 -135 -45 E
32 ELECTRICITY / 0 0 4 5 / 0 5 4 5 / 0 5 4 10 E
33 PEN / 10 0 0 10 E
34 TWO Q 2 4 2 -45 150 / .5 0 3.7 2.7 / .5 0 3 0 E
35 HELLO : 16 0 0 : 5 6 0 : 7 12 0 E
36 I;ME;MY / 7 0 7 6 / 6 5 7 6 : 3 0 0 E
37 LIKE : 18 0 0 : 19 11.5 3.5 : 9 16.5 0 E
38 HAPPY : 18 0 0 : 29 13 0 E
39 FOOD : 2 0 1.5 : 16 0 0 E
40 FRIEND : 3 0 0 : 37 6 0 E
41 ANIMAL : 25 0 0 : 25 5 0 : 24 0 3.5 E
42 WHY / 7 1 7 9 / 7 1 13 5.5 / 7 9 13 5.5 : 20 0 0 E
43 YOU : 3 0 0 : 34 5 0 E
44 WANT Q 8 7 5 -25 40 Q 17 3 5 155 215 : 18 0 0 : 11 4 11 E
45 PLEASE : 9 0 0 : 18 2 0 E
46 ANGRY / 0 10 1.5 7 / 1.5 7 3 10 / 14 5 16.5 0 / 14 5 16.5 10 / 15.5
5 18 0 / 15.5 5 18 10 : 13 0 0 : 18 3 0 E
47 DRINK : 39 0 0 : 31 6 0 E
48 GOD / 0 0 10 0 / 0 0 5 6 / 5 6 10 0 / 0 4 3.4 4 / 6.6 4 10 4 / 0 4 5
10 / 5 10 10 4 E
49 PAGE / 0 0 0 10 / 0 0 6 0 / 0 10 6 10 / 6 0 6 10 E
50 BOOK / 0 0 5 0 / 0 0 0 10 / 0 10 5 10 : 49 5 0 E
51 FLOWER : 16 0 2.5 : 8 2.5 0 E
52 BIRD Q 11 4.2 8.5 136 201 Q -6 4.2 8.5 -31 44 E
53 WOMAN / 0 0 5 0 : 26 0 0 E
54 FATHER / 0 8.5 2.5 10 / 2.5 10 5 8.5 : 26 0 0 E
55 MOTHER / 0 0 5 0 : 54 0 0 E
56 BROTHER / 0 0 2.5 5 / 2.5 5 5 0 / 0 7 2.5 10 / 2.5 10 5 7 : 34 6.5 0
E
```

```
57 SISTER / 0 0 5 0 : 54 0 0 E
58 HOUSE / 0 0 0 6 / 0 0 8 0 / 8 0 8 6 / 0 6 4 10 / 8 6 4 10 E
59 BOX / 0 0 10 0 / 0 0 0 10 / 10 0 10 10 / 0 10 10 10 E
60 SLEEP : 59 0 0 : 28 2.5 0 E
61 FUNNY / 3 12 5 10 / 5 10 7 12 : 38 0 0 : 16 15.5 0 E
62 AFRAID Q 25.5 5 9.25 150 215 : 18 0 0 : 14 12 0 : 20 20 0 E
63 COME / 6.5 0 6.5 7.5 / 1 8 2.5 10 / 2.5 10 4 8 : 5 0 0 E
64 HOW / 6 0 11 10 / 11 10 16 0 : 20 0 0 E
66 WHAT? : 20 0 0 : 59 7 0 E
65 WHO? : 20 0 0 : 3 5.5 0 E
67 WHICH? : 20 0 0 : 2 6 0 : 10 8.5 3.5 : 10 8.5 6.5 E
68 WHERE? : 2 0 -5 : 20 0 1.5 E
69 WHEN? / 12.5 5 12.5 10 / 12.5 5 17.5 5 : 20 0 0 : 23 7.5 0 E
70 TOILET / 2 5 2 10 / 5 3.5 5 7 / 2 7 5 7 : 31 0 0 E
71 WASH / 1 7 3.5 10 / 3.5 10 6 7 Q 3.25 4.5 3.4 192 359 : 31 0 0 E
72 HOW-MUCH / 8 0 18 10 / 8 10 18 0 : 20 0 0 E
73 THINK / 2.5 6 5 8.5 / 5 8.5 7.5 6 : 15 0 -5 E
74 NAME : 33 0 0 : 16 2.5 0 E
75 WORD : 10 2.5 3.5 : 10 2.5 6.5 : 2 0 0 : 16 9 0 E
76 LIGHT : 23 0 0 : 28 2.5 0 E
77 TABLE / 0 2.5 0 7.5 / 0 7.5 10 7.5 / 10 7.5 10 2.5 E
78 LEGS / 0 0 3 0 / 0 0 3 10 / 3 10 6 0 / 6 0 10 0 E
79 ROOM / 0 0 10 0 / 10 0 10 10 / 0 10 10 10 E
80 SHOW : 58 0 0 : 28 10 1 E
81 WEATHER Q 3 2.5 3.25 52 150 Q 7 7.5 3.25 -150 -52 : 13 5 0 : 1 5 0 :
23 0 0 E
82 DAY : 24 0 0 : 23 0 0 E
83 WEEK-END / .5 7 3.5 7 / 7 6 7.5 7 / 7.5 7 7.5 2 / 4 5 6 5 / 5 4 5 6
Q 9.5 2 7.75 140 180 : 82 0 0 E
84 STREET / 0 0 5 0 : 13 1 0 : 58 5 0 E
87 STORE / 0 0 10 0 / 10 0 10 10 / 10 10 4 10 : 22 11 0 : 16 13.5 -2.5
: 16 13.5 2.5 : 1 16 0 E
86 NEWS : 17 3 0 : 28 0 0 E
85 OUTING / 16 5 25 5 : 78 0 0 : 59 11 0 : 4 23.5 3.5 E
88 FIRE Q 3.25 2.5 3.25 132 230 Q 1.25 7.5 3.25 -48 50 E
89 PAIN / 9 0 13 5 / 13 5 17 0 / 17 0 17 5 : 18 0 0 E
90 HAND / 0 5 5 0 / 5 0 5 10 E
91 HEAD / 2.5 1.5 7.5 1.5 / 5 3 5 7.5 / 2 9 3.5 7 / 6.5 7 8 9 : 23 0 0
E
```

```
 92 HOT / 0 5 3 2 / 0 5 3 8 / 7.5 2 10.5 5 / 7.5 8 10.5 5 / 3.5 11 5 12.
5 / 5 12.5 6.5 11 : 88 4 0 E
 93 DIFFICULT / 10 10 10 6 / 10 4 10 0 : 13 3 8.5 : 5 5 0 : 2 0 0 E
 95 NEW,YOUNG / 2.5 2.5 2.5 0 : 2 0 -5 : 16 0 0 : 13 0 8.5 E
 96 BIG / 1 11 2.5 12.5 / 2.5 12.5 4 11 / 0 10 5 10 : 3 0 0 E
 97 HELP / 0 0 5 10 / 3.5 11 5 12.5 / 5 12.5 6.5 11 : 3 2.5 0 E
 98 MAKE / 0 0 5 6 / 5 6 10 0 / 0 0 10 0 / 3.5 8.5 5 10 / 5 10 6.5 8.5 E
 99 TEACHER Q 21 6 4 0 180 Q 10 5 5 -156 -31 / 17 0 17 6 / 17 0 25 0 / 2
5 0 25 6 / 8.5 8.5 10 10 / 10 10 11.5 8.5 : 1 10 0 : 3 0 0 E
 100 GAME / 0 5 0 10 / 0 5 5 5 / 0 10 5 10 / 5 5 5 10 / 5 0 8 6 / 8 6 11
 0 / 23.5 8.5 25 10 / 25 10 26.5 8.5 : 1 25 0 : 18 11.5 0 E
 101 BIRTHDAY / 14.5 -1.5 14.5 6.5 / 10.5 2.5 18.5 2.5 / 11.5 -.5 18 5.5
 / 11.5 5.5 18 -.5 : 82 0 0 E
 102 BE / 2.5 0 2.5 5 / 2.5 5 0 7.5 / 2.5 5 5 7.5 : 16 0 -2.5 E
 103 HAVE / 0 0 3 0 / 0 6 1.5 7.5 / 1.5 7.5 3 6 : 19 0 1 E
 104 MUSIC / 5 3 5 10 : 16 0 -2.5 : 17 7.5 0 E
 105 THANKS Q 15 5 5 -156 -31 / 15 0 15 10 / 13.5 8.5 15 10 / 15 10 16.5
 8.5 : 18 0 0 E
 106 OPPOSITE / 0 8.5 1.5 10 / 1.5 10 1.5 0 / 1.5 0 3 1.5 E
 107 I'M...SORRY / 23 0 23 10 : 13 21.5 0 : 18 9.5 0 : 36 0 0 E
 108 HOSPITAL / 11.5 0 11.5 10 : 58 0 0 : 88 10.5 0 E
 109 SCHOOL Q 15 5 5 -156 -31 Q 25 6 4 0 180 / 13.5 8.5 15 10 / 15 10 16
.5 8.5 / 21 6 21 0 / 21 0 29 0 / 29 0 29 6 : 58 0 0 : 1 15 0 E
 110 TV : 59 0 0 : 28 11.5 0 : 17 17 0 : 32 23.5 0 E
 111 GOOD / 3.5 10 5 8.5 / 5 8.5 6.5 10 : 15 -3.5 0 : 19 3.5 0 : 9 12.5
0 E
 112 KNOW / 3.5 11 5 12.5 / 5 12.5 6.5 11 / 0 0 8 0 / 0 0 0 6 / 8 0 8 6
Q 4 6 4 0 180 E
 113 GIVE Q 5 5 5 -156 -31 / 3.5 8.5 5 10 / 5 10 6.5 8.5 / 3.5 11 5 12.5
 / 5 12.5 6.5 11 : 1 5 0 E
 114 CAR / 0 5 14 5 / 2.75 .75 6.25 4.25 / 2.75 4.25 6.25 .75 / 7.75 .75
 11.25 4.25 / 7.75 4.25 11.25 .75 : 16 2 -2.5 : 16 7 -2.5 E
 115 WHEELCHAIR / .75 .75 4.25 4.25 / .75 4.25 4.25 .75 : 16 0 -2.5 : 1
0 0 : 2 0 0 : 8 5 0 E
 116 CLOTHING / .5 3.5 3 6 / 3 6 5.5 3.5 / 0 1.5 6 1.5 / 0 4.5 6 4.5 / 1
.5 0 1.5 6 / 4.5 0 4.5 6 E
END
```

Appendix E

BLSEDIT LISTING

```
PAGE  1

 50 OPEN #0,"BLSDATA" \OPEN #1,"BLSDATE"
 60 INPUT "TYPE THE SYMBOL # THAT NEEDS EDITING :   ",M
 STOP
 READY
 BYE
 *JP C000
 READY
 LIST

 10 REM BLSEDIT JAN 6
 20 REM EDITS BLSDATA BY READING BLSDATA AND
 30 REM PUTTING THE CORRECTION IN BLSDATE
 35 DIM C$(80)
 40 DIM B$(255)
 41 DIM A$(255)
 50 OPEN #0,"BLSDATA" \OPEN #1,"BLSDATE"
 60 INPUT "TYPE THE SYMBOL # THAT NEEDS EDITING :   ",M
 70 FOR I=0 TO M+1
 80 READ #0,A$
 90 IF VAL(A$(2))=M THEN EXIT 130
 100 WRITE #1,A$
 110 NEXT I
 120 PRINT "ERROR" \STOP
 130 PRINT A$,LEN(A$)
 140 INPUT B$
 150 IF B$(LEN(B$))="E" THEN 200
 160 INPUT C$
 170 B$=B$+C$
 180 GOTO 150
 200 PRINT B$,LEN(B$)
 202 INPUT "IF OK PUSH ANY CHAR...IF NOT CONT-C  ",X$
 205 WRITE #1,B$
 210 FOR I=M-1 TO 120
 220 READ #0,A$ \WRITE #1,A$
 225 IF A$="END" THEN EXIT 248
 230 NEXT I \REM WILL PROBABLY END WITH A TYPE ERROR
 240 PRINT"ERROR IN DATA FILE--EXCEDED 120 SYMBOLS"
 245 STOP
 248 P=0\REM COPY BLSDATE TO BLSDATA********************
 250 FOR I=0 TO 120
 260 READ #1 %P,A$
 270 WRITE #0 %P,A$
 280 IF A$="END" THEN EXIT 330
 290 P=P+LEN(A$)+2
 300 NEXT I
 310 PRINT "ERROR--DID NOT HAVE 'END' "
 320 STOP
 330 CLOSE #0\CLOSE #1\PRINT "DONE"\STOP
READY
```

Appendix F

BLSXY LISTING

PAGE Q

```
10 REM BLSXY JAN 8
20 REM READS EQUATION PARAMETERS FROM BLSDATA TO MAKE BLSDXY
30 DIM A(120),S$(20)\REM # OF SYMBOLS, MAX LENGTH OF SYMBOL NAME
35 DIM X(300),Y(300),A$(255),F$(850)
36 DIM J$(800)
37 DIM C$(20),H$(20)
40 OPEN #0, "BLSDATA"
50 OPEN #1, "BLSDXY"
60 F=1.4 \REM ASPECT RATIO
70 T=1  \REM THRESHOLD # OF SAMPLES TO MAKE DOT
80 H=20 \REM HEIGHT OF SYMBOL # OF RASTER LINES
90 C=2 \REM # OF DEGREES/SAMPLE
100 S=3  \REM # OF SAMPLES/GRID SPACE
105 PRINT"# SYMBOL                      N    D    A(M9)    LEN(A$)+2"
106 PRINT
107 PRINT"------------------------------------------------------------"
110 READ #0,A$
112 IF A$="END" THEN STOP
113 L9=0\REM LIMIT ON THE NUMBER OF ERROR TRIALS
115 N=0 \REM # OF SAMPLES
120 P=2 \REM POINTER FOR A$
130 GOSUB 2000 \M9=VAL(C$) \REM SYMB #
135 A(M9)=Q\REM FILE POINTER FOR START OF THIS SYMBOL
140 GOSUB 2000 \S$=C$ \REM SYMBOL NAME
150 GOSUB 2000
160 IF C$="Q" THEN 500
170 IF C$="/" THEN 700
180 IF C$=":" THEN 1000
190 IF C$="E" THEN 1500 \REM SYMBOL MADE OF ALL LINES AND CIRCLES
200 PRINT "COMMAND ERROR", C$
210 STOP
500 REM CIRCLE ROUTINE**************************************
510 GOSUB 2000 \X1=VAL(C$)
520 GOSUB 2000 \Y1=VAL(C$)
530 GOSUB 2000 \R=VAL(C$)
540 GOSUB 2000  \A=VAL(C$)*710/(360*113)
550 GOSUB 2000 \B=VAL(C$)*710/(360*113)
560 X(N)=(X1+R*COS(A))*H/10*F
570 Y(N)=(Y1+R*SIN(A))*H/10
580 A=A+C*710/(360*113)*5/R
590 N=N+1
600 IF A<B THEN 560
610 GOTO 150
700 REM LINE ROUTINE*****************************************
710 GOSUB 2000 \X3=VAL(C$)
720 GOSUB 2000 \Y3=VAL(C$)
730 GOSUB 2000 \X4=VAL(C$)
740 GOSUB 2000 \Y4=VAL(C$)
```

PAGE 1

```
750 IF X3=X4 THEN 900
760 IF X3<X4 THEN 790
770 X7=X3 \X3=X4 \X4=X7
780 Y7=Y3 \Y3=Y4 \Y4=Y7
790 M=(Y4-Y3)/(X4-X3)
800 X5=SQRT(1/S↑2/(1+M↑2))*10/H
810 B=Y3-M*X3
820 IF M=0 THEN B=Y3
830 Y(N)=(M*X3+B)*H/10
840 X(N)=X3*F*H/10
850 N=N+1
860 X3=X3+X5
870 IF X3<X4 THEN 830
880 GOTO 150
890 REM VERT. LINE
900 X(N)=X3*F*H/10
910 Y(N)=Y3*H/10 \N=N+1
920 Y3=Y3+10/(H*S)
930 IF Y3<Y4 THEN 900
940 GOTO 150
1000 REM \ OTHER SYMBOL ROUTINE**********************************
1010 GOSUB 3000\REM COMPUTE DOTS FROM SAMPLE POINTS
1020 GOSUB 2000 \S1=VAL(C$)
1030 GOSUB 2000 \X0=VAL(C$)
1040 GOSUB 2000 \Y0=VAL(C$)
1060 READ #1 %A(S1),J$
1100 P1=2
1101 GOSUB 2100 \REM SYMBOL #
1120 GOSUB 2100 \REM SYMBOL NAME
1130 D=D+1
1135 X0=X0*F*H/10
1136 Y0=Y0*H/10
1140 GOSUB 2100\F$=F$+STR$(INT(VAL(H$)+X0+.5))
1150 GOSUB 2100\F$=F$+STR$(INT(VAL(H$)+Y0+.5))
1160 GOSUB 2100
1170 IF H$="E" THEN 1200
1180 D=D+1\F$=F$+STR$(INT(VAL(H$)+X0+.5))
1190 GOTO 1150
1200 GOSUB 2000 \REM GET COMMAND
1210 IF C$=":"THEN 1020
1220 IF C$<>"E" THEN STOP
1300 F$=F$+" E"
1302 IF M9=0 THEN A(M9)=0 \REM A(M9)-FILE POINTER FOR EACH SYMBOL
1304 Q=A(M9)+2+LEN(F$)
1306 IF LEN(F$)>255 THEN Q=Q+1
1310 WRITE #1 %A(M9),F$
1315 PRINT M9," ",S$,TAB(24),N,D,A(M9),LEN(A$)+2
1320 GOTO 110 \REM GO START ANOTHER SYMBOL
1400 WRITE #1,"END" \CLOSE #1
1420 STOP
1500 REM "E"
1510 GOSUB 3000
1520 GOTO 1300
2000 C$=""\REM DECOMPILE SYMBOL STRING
2010 B$=A$(P,P)\P=P+1
2020 IF B$=" " THEN RETURN \REM LOOKING FOR A SPACE
```

PAGE 2

```
2030 C$=C$+B$
2035 IF P=LEN(A$) +1 THEN RETURN
2040 GOTO 2010
2100 H$="" \REM DECOMPILES STRING
2110 G$=J$(P1,P1)\P1=P1+1
2120 IF G$=" " THEN RETURN
2130 H$=H$+G$
2135 IF P1=LEN(J$)+1 THEN RETURN
2140 GOTO 2110
3000 REM ERROR ROUTINE-ASSEMBLES SAMPLES INTO DOTS************
3010 REM FROM X(N),Y(N) TABLE
3011 F$=STR$(M9)+" "+S$\REM START ASSEMBLY
3015 IF N=0 THEN RETURN\REM NO SAMPLES TO COMPUTE
3020 E0=0\E1=0\L9=L9+1
3030 FOR K=0 TO N
3040 E0=E0+X(K)-INT(X(K)+.5)
3050 E1=E1+Y(K)-INT(Y(K)+.5)
3060 NEXT K
3070 FOR K=0 TO N
3080 X(K)=X(K)-E0/N
3090 Y(K)=Y(K)-E1/N
3100 NEXT K
3105 IF L9>10 THEN 3120
3110 IF ABS(E0/N)+ABS(E1/N)<.0015 THEN 3020
3120 FOR K=0 TO N \REM THRESHOLD ROUTINE
3130 X(K)=1000*INT(X(K)+.5)+INT(Y(K)+.5)
3140 NEXT K \REM COORDINATES OF SAMPLES NOW IN X(N)
3150 D=-1 \REM # OF DOTS
3160 FOR L=0 TO N
3170 IF X(L)<> INT(X(L)) THEN 3240
3180 FOR K=L+1 TO N
3190 IF INT(X(L))<>X(K) THEN 3220
3200 X(L)=X(L)+.01
3210 X(K)=X(K)+.01
3220 NEXT K
3230 IF X(L)-INT(X(L))>=T*.01 THEN GOSUB 3300
3240 NEXT L
3250 RETURN
3300 D=D+1
3320 F$=F$+STR$(INT(X(L)/1000))+STR$(INT(X(L)-1000*INT(X(L)/1000)))
3340 RETURN
READY
```

Appendix G

DISPLAY LISTING

```
READY
LIST

 10 REM DISPLAY JAN 9
 20 REM DISPLAYS SYMBOLS FORM "BLSDXY" FILE
 30 DIM F$(850),S$(20),S(120)
 35 DIM H$(20)
 100 OPEN#1,"BLSDXY"\P=0\REM COMPILE INDEX OF SYMBOLS
 105 FILL 53308,65\Z=CALL(50099)\REM NO DMA-41
 110 FOR I=0 TO 105
 115 IF TYP(1)<>1 THEN EXIT 160
 120 READ #1,F$\S(I)=P
 130 P=P+LEN(F$)+2
 135 IF LEN(F$)>255 THEN P=P+1
 140 IF F$="END" THEN EXIT 170
 150 NEXT I
 160 PRINT "FILE BLSDXY HAS NO END"
 170 PRINT "FILE HAS ",I," SYMBOLS IN IT"
 180 REM SUB TO " OR" DATA WITH DISPLAY MEMORY
 200 FILL 53361,58\REM 3A LDA D071
 210 FILL 53362,121\REM 79
 220 FILL 53363,208\REM D0
 230 FILL 53364,235\REM EB XCHG
 240 FILL 53365,182\REM B6 ORA,M
 250 FILL 53366,119\REM 77 MOV M,A
 260 FILL 53367,235\REM EB XCHG
 270 FILL 53368,201\REM C9 RETURN
 280 REM SET UP MBI RAM BYTES FOR EDI-Q
 290 REM\ NEW SCREEN AREA,SUPER DENSE
 300 A0=0
 310 FILL 53300,000\FILL 53301,A0\REM D034 TCU
 320 FILL 53302,000\FILL 53303,A0\REM THOME
 330 FILL 53304,000\FILL 53305,A0\REM TSS
 340 FILL 53306,000\FILL 53307,A0+10\REM TEOM
 350 FILL 53308,17\REM TDFMT-SUPER DENSE GRAPHICS
 360 Z=CALL(50099)\REM EDIT-Q
 490 X0=10\Y0=170\REM POSITION OF SYMBOL ON SCREEN
 492 INPUT  "DO YOU WANT THE SCREEN CLEARED: YES OR NO? ",C$
 494 IF C$="NO" THEN 530
 500 FOR A=A0 TO A0+4096
 510 FILL A,0
 520 NEXT A
 530 INPUT1 "#: ",M
 535 Z=CALL(50099)\REM EDIT-Q
 540 READ #1 %S(M),F$ \P1=2
 545 Z=CALL(50099)\REM EDIT-Q
 550 GOSUB 2100
 560 GOSUB 2100\S$=H$\REM SYMBOL NAME
 570 W=0\REM FIND WIDTH AND # OF DOTS
 575 P3=P1
 580 FOR I=0 TO 420
 590 GOSUB 2100\IF H$="E" THEN EXIT 620
 595 IF VAL(H$)>W THEN W=VAL(H$)
 600 GOSUB 2100
 610 NEXT I\STOP
 620 N=I-1\P1=P3
```

PAGE 1

```
640 IF X0>320 THEN 1000
650 FOR I=0 TO N\REM DIG OUT THE DOTS AND DISPLAY
660 GOSUB 2100\X=VAL(H$)+X0
670 GOSUB 2100\Y=VAL(H$)+Y0
680 GOSUB 900\REM DISPLAY DOT
690 NEXT I
695 X0=X0+W+20
700 REM DUMP THE SYMBOL NAME
710 PRINT " ",S$,TAB(20),X0,Y0,W
720  GOTO 530\REM GO START ANOTHER SYMBOL
900 REM DISPLAY X,Y DOT
910 A1=INT(X/8)+INT(199.5-Y)*40
920 B1=X-INT(X/8)*8
930 FOR J=0 TO 7
940 IF INT(B1)=J THEN C1=2↑J
950 NEXT J
960 FILL 53369,C1
970 Z=CALL(53361,A1)
980 RETURN
1000 X0=10\REM START NEXT SCREEN LINE
1010 Y0=Y0-30
1020 IF Y0<100 THEN 490\REM CLEAR SCREEN--START AT TOP
1030 GOTO 650
2100 H$=""
2110 G$=F$(P1,P1)\P1=P1+1
2120 IF G$=" " THEN RETURN
2130 H$=H$+G$
2140 IF P1=LEN(F$)+1 THEN RETURN
2150 GOTO 2110
READY
```

Appendix H

BLSCODE LISTING

```
*GO BASIC
READY
LIST

READY
BYE
*LI
DOS          4   10    0
G/BASIC     14   20    1  2A00
DEMO        34    1    4
DEMO/RUN    35    7    1   400
GB400       42   20    1   400
BLSDATA     62   30    3
BLSDATE     92   30    3
BLSXY      122   15    2
BLSCODE    137   20    2
BLSEDIT    157   10    2
DISPLAY    167   10    2
BLSDXY     177  150    3
*JP 2A04
```

```
READY
LIST

10 REM BLSCODE JAN 11
20 REM COMPILES X-Y COORDINATES FROM BLSDXY TO CODE
30 DIM X(400),Y(400),A$(400),T(3,3),U(14)
35 PRINT "   U    D0    X2    Y2    X1    Y1    Z0    A    B"
40 OPEN #1,"BLSDXY"
50 FOR I=0 TO 3 \FOR J=0 TO 3
60 READ T(I,J)
70 NEXT J \NEXT I
80 DATA 0,0,0,0,64,16,4,1,128,32,8,2,192,48,12,3
90 RESTORE 100
91 FOR J=0 TO 13
92 READ U(J)
93 NEXT J
100 DATA .3,.2,.02,.002,.0101,.003,.01,.1,.1,.01,.0101,.002,.02,.2
110 A=0\REM STORAGE AREA FOR CODE
120 P1=2\REM STRING POINTER
122 IF C$="E" THEN PRINT A$
123 PRINT
125 READ #1,A$
```

PAGE 1

```
 130 N=0 \REM DOT COUNTER
 140 GOSUB 3000 \M=VAL(C$) \REM SYMBOL #
 150 GOSUB 3000 \S$=C$ \REM SYMBOL NAME
 160 GOSUB 3000 \X(N)=VAL(C$)
 170 GOSUB 3000 \Y(N)=VAL(C$)
 180 GOSUB 3000
 190 IF C$="E" THEN 230
 200 N=N+1 \X(N)=VAL(C$)
 210 GOTO 170
 220 REM COORDINATES ARE NOW IN X(N),Y(N)**********************
 230 FILL(3841 +M*2),INT(A/256) \REM INDEX AT 0F00
 240 FILL(3840 +M*2),A-INT(A/256)*256
 250 FILL A,0
 255 A$=S$+" "\REM STRING OF CODES
 260 N3=0 \P=1 \D0=0 \C4=0 \X0=-1 \Y0=0
 270 Z0=100 \REM FIND NEAREST POINT X2,Y2**********************
 280 FOR I=0 TO N
 285 IF X(I)<>INT(X(I)) THEN 330
 290 Z1=SQRT((X(I)-X0)↑2+(Y(I)-Y0)↑2)
 300 IF Z1>=Z0 THEN 330
 310 X2=X(I) \Y2=Y(I) \Z0=Z1
 320 S=I
 330 NEXT I \REM NOW FIND NEW DIRECTION
 340 IF Y2>Y0 THEN D1=SGN(X0-X2)+2
 350 IF Y2<Y0 THEN D1=SGN(X2-X0)+6
 360 IF Y2=Y0 AND X2<X0 THEN D1=4
 370 IF Y2=Y0 AND X2>X0 THEN D1=0
 380 IF Z0>1.5 THEN 2090 \REM LIFT PEN --INTERMEDIATE DOT
 390 GOTO 1480 \REM ADJOINING DOT
1370 B=EXAM(A)+T(V,C4) \REM 2-BIT ROUTINE
1400 FILL A,B
1405 A$=A$+STR$(V)
1406 IF V<>0 THEN A$=A$+" "
1420 C4=C4+1 \IF C4<4 THEN RETURN
1440 C4=0 \A=A+1 \FILL A,0
1470 RETURN
1480 IF P=0 THEN 1515 \REM PEN ALREADY DOWN
1490 P=0\V=.03\GOSUB 2030
1515 GOSUB 1880 \REM GET CODE AND DUMP
1520 X0=X2 \Y0=Y2 \D0=D1
1530 X(S)=X(S)+.1
1550 N3=N3+1
1560 IF N3<=N THEN 270
1565 W=0
1570 FOR I=0 TO N\REM FIND WIDTH
1590 IF X(I)>W THEN W=INT(X(I))
1600 NEXT I
1605 H=20\F=1.4
1610 PRINT "WIDTH= ";W,W*10/(H*F)
1620 FOR I=1 TO 3
1630 V=0 \GOSUB 1370
1640 NEXT I
1670 IF W>31 THEN V0=2
1680 IF W>31 THEN W=W-32
1690 IF W>15 THEN V1=1
1700 IF W>15 THEN W=W-16
```

PAGE 2

```
1710 V=V0+V1
1720 GOSUB 1370
1730 V0=0 \V1=0
1740 IF W>7 THEN V0=2
1750 IF W>7 THEN W=W-8
1760 IF W>3 THEN V1=1
1770 IF W>3 THEN W=W-4
1775 V=V0+V1
1780 GOSUB 1370
1790 V0=0 \V1=0
1800 IF W>1 THEN V0=2
1810 IF W>1 THEN W=W-2
1820 IF W>0 THEN V1=1
1830 V=V0+V1
1840 GOSUB 1370
1850 PRINT M,S$;"NEXT ADDRESS= ",A+1
1860 A=A+1
1870 GOTO 120
1880 I=D1-D0 \REM CALC CODE
1900 IF I<0 THEN I =I+8
1910 U=U(I)
2030 U=U*10 \V=INT(U) \REM OUTPUT
2060 GOSUB 1370
2070 IF U<>INT(U) THEN 2030
2080 RETURN
2090 IF P=1 THEN 2120
2100 P=1 \U=.003
2110 GOSUB 2030
2120 REM CALC X1,Y1
2130 X1=X0
2140 IF X2-X0>=1 THEN X1=X0+1
2150 IF X0-X2>=1 THEN X1=X0-1
2160 Y1=Y0
2170 IF Y2-Y0>=1 THEN Y1=Y0+1
2180 IF Y0-Y2>=1 THEN Y1=Y0-1
2190 GOSUB 1880
2200 X0=X1 \Y0=Y1
2210 D0=D1\GOTO 270
3000 C$=""
3010 B$=A$(P1,P1)\P1=P1+1
3020 IF B$=" " THEN RETURN
3030 C$=C$+B$
3040 IF P1=LEN(A$)+1 THEN RETURN
3050 GOTO 3010
READY
```

Appendix I

ASSEMBLY LISTING - DECOMPILER

Path Direction Method

ÜE.
7
2D.39
DD.

Oct 30/78 - "F3"
Symbols in "F30S" table 0F00

```
                0010 *SYMBOL DEFINITION
                0020 *RON & DAN PALMER
                0030 *AUGUST 1978
                0040 *
                0050 XSCAN   DS     1
                0060 YSCAN   DS     1
                0070 XOUT    DS     1
                0080 YOUT    DS     1
                0090 *WIDTH OF SYMBOL ....  00XXXXXX
                0100 W       DS     1
                0110 *LOCATION WITHIN A SYMBOL
                0120 X       DS     1
                0130 Y       DS     1
                0140 *# OF TWO BIT CODES GONE BY
                0150 CODE    DS     1
                0160 *ADDRESS OF SYMBOL DATA TO BE DISPLAYED
                0170 SYMB    DS     2
                0180 DR      DS     1              ;DIRECTION REGISTER
                0190 PEN     DS     1              ;0=DOWN   1=UP
                0200 X0      DS     2              ;GRAPHIC CURSOR LOC
                0210 Y0      DS     1
                0220 *
                0230 *
                0240 XPOS    DS     3
                0250 *ADDRESS OF THE SYMBOL TABLE
                0260 *
                0270 *
                0280 *    XPOS ..XL
                0290 *         ..XH
                0300 *         ..Y
                0310 *INDEX OF STARTING ADDRESSES OF SYMBOLS
0 0F            0320 TABLE   DW     0F00H          ;RESERVED FOR FUTURE
                0330 *
                0340 *INIT.......INITIALIZE
                0350 *RON AND DAN PALMER
                0360 *AUGUST 1978
                0370 *MUST HAVE TABLE AND SYMB DATA IN BEFOREHAND
1 0A 03         0380 INIT    LXI    H,DR
6 00            0390         MVI    M,00
1 0B 03         0400         LXI    H,PEN
6 01            0410         MVI    M,01           ;UP
3               0420         INX    H
6 00            0430         MVI    M,00
1 0E 03         0440         LXI    H,Y0
6 14            0450         MVI    M,20           ;STRT 20 FROM TOP-SCRN
1 00 00         0460         LXI    H,0000
2 0C 03         0470         SHLD   X0
1 79 D0         0480         LXI    H,0D079H       ;SCMOD
6 01            0490         MVI    M,01           0R
1 00 60         0500         LXI    H,6000H        ;GRAPHICS STR-6K
2 7A D0         0510         SHLD   0D07AH         GDSTART
9               0520         RET
A 7A D0         0530 DISPL   LHLD   0D07AH         GDSTR..SET DIPL ADDR
```

```
AGE  1

E5                       0540            PUSH   H
3E 20                    0550            MVI    A,20H
84                       0560            ADD    H
67                       0570            MOV    H,A
E5                       0580            PUSH   H
21 11 00                 0590            LXI    H,0011H          ;SUPER DENSE MODE
E5                       0600            PUSH   H
C3 50 C1                 0610            JMP    0C150H           ;UPDATE ROUTINE
00                       0620            NOP
2A 7A D0                 0630 CLEAR      LHLD   0D07AH           GDSTR..START ADDR
E5                       0640            PUSH   H
3E 20                    0650            MVI    A,20H
84                       0660            ADD    H
67                       0670            MOV    H,A
E5                       0680            PUSH   H
21 00 00                 0690            LXI    H,0000           ;FILL WITH ZEROES
E5                       0700            PUSH   H
C3 EE C0                 0710            JMP    0C0EEH           ;FILL ROUTINE
00                       0720            NOP
                         0730 *CURRENT ADDRESS OF START OF STORY
                         0740 STORD  DS     2                    ;SEQUENT ADDR OF SYMB
                         0750 STORN  DS     1                    ; CURRENT # OF SYMBOLS
                         0760 * BITS
                         0770 *
                         0780 *
                         0790 *PUTS THE TWO BITS ON THE LEFT SIDE OF
                         0800 *THE ACCUMULATOR ....BBXXXXXX
                         0810 *KEEPS TRACK OF # OF BITS TAKEN OUT
                         0820 *IN CODE AND INCREMENTS SYMB WHEN EXHAUSTED
                         0830 *
2A 08 03                 0840 BITS       LHLD   SYMB
3A 07 03                 0850            LDA    CODE
FE 00                    0860            CPI    00
CA 7F 03                 0870            JZ     COD0
FE 01                    0880            CPI    01
CA 87 03                 0890            JZ     COD1
FE 02                    0900            CPI    02
CA 8D 03                 0910            JZ     COD2
FE 03                    0920            CPI    03
CA 95 03                 0930            JZ     COD3
FE 04                    0940            CPI    04
CA 9F 03                 0950            JZ     NEXT
C3 00 C0                 0960            JMP    0C000H           ;ERROR IN CODE
7E                       0970 COD0       MOV    A,M
E6 C0                    0980            ANI    0C0H             ; PAD 6 RIGHT BITS
21 07 03                 0990            LXI    H,CODE
34                       1000            INR    M
C9                       1010            RET
7E                       1020 COD1       MOV    A,M
07                       1030            RLC
07                       1040            RLC
C3 80 03                 1050            JMP    COD0+1
7E                       1060 COD2       MOV    A,M
07                       1070            RLC
07                       1080            RLC
07                       1090            RLC
```

```
07                1100          RLC
C3 80 03          1110          JMP    COD0+1
7E                1120 COD3     MOV    A,M
07                1130          RLC
07                1140          RLC
07                1150          RLC
07                1160          RLC
07                1170          RLC
07                1180          RLC
C3 80 03          1190          JMP    COD0+1
3E 00             1200 NEXT     MVI    A,00
32 07 03          1210          STA    CODE           ;INITIALIZE CODE
21 08 03          1220          LXI    H,SYMB
                  1230 *SYMB=ADDRESS OF CODE CURRENTLY WORKING ON
34                1240          INR    M
                  1250 *INR SYMB, SINCE USED ALL THE BITS
C3 5D 03          1260          JMP    BITS
                  1270 *CODE MUST=0 AT BEGIN OF EACH SYMBOL
                  1280 *
                  1290 *
                  1300 *
                  1310 *DIREC....RON PALMER  AUGUST 1978
                  1320 *
                  1330 *TAKES CODE FROM BITS AND DOES AS DIRECTED
                  1340 *
CD 5D 03          1350 DIREC    CALL   BITS           ;GET FIRST TWO BITS
07                1360          RLC    PUT            TWO BITS ON RIGHT
07                1370          RLC
                  1380 *
                  1390 *
                  1400 *
                  1410 *
FE 03             1420          CPI    03
CA 5B 04          1430          JZ     XY             ;RETURN IMMEDIATELY
                  1440 *KEEP OLD DIRECTION
FE 01             1450          CPI    01
21 0A 03          1460          LXI    H,DR
CA 47 04          1470          JZ     SUB1           ;SUBT 1 FROM DR
FE 02             1480          CPI    02             ;ADD 1 TO DIREC REG
CA 41 04          1490          JZ     ADD1
CD 5D 03          1500          CALL   BITS           ;MUST HAVE BEEN ZERO
21 0A 03          1510          LXI    H,DR
FE C0             1520          CPI    0C0H           CHANGE PEN
CA 53 04          1530          JZ     CHGPD          ; PEN DOWN
FE 40             1540          CPI    040H
CA 46 04          1550          JZ     SUB2
FE 80             1560          CPI    080H
CA 40 04          1570          JZ     ADD2
CD 5D 03          1580          CALL   BITS           ;MUST HAVE BEEN ZERO
FE C0             1590          CPI    0C0H
21 0A 03          1600          LXI    H,DR
CA 4B 04          1610          JZ     CHGPU          ;PEN UP
FE 40             1620          CPI    040H
CA 45 04          1630          JZ     SUB3
FE 80             1640          CPI    080H
CA 3F 04          1650          JZ     ADD3
```

```
                1660 *MUST BE THREE SETS OF ZEROS..END OF SYMBOL
CD 5D 03        1670 ENDSY  CALL  BITS
0F              1680        RRC
0F              1690        RRC
32 04 03        1700        STA   W                ; W IS STORAGE BYE FOR
SYM
CD 5D 03        1710        CALL  BITS
0F              1720        RRC
0F              1730        RRC
0F              1740        RRC
0F              1750        RRC
21 04 03        1760        LXI   H,W
86              1770        ADD   M
32 04 03        1780        STA   W
CD 5D 03        1790        CALL  BITS
07              1800        RLC
07              1810        RLC
21 04 03        1820        LXI   H,W
86              1830        ADD   M
32 04 03        1840        STA   W
2A 0C 03        1850        LHLD  X0
85              1860        ADD   L                ;WIDTH + X0 >MAX?
DA 1C 04        1870        JC    ENDLN            ;>256 DOTS WIDE
6F              1880        MOV   L,A              ;ON THE TV SCREEN
22 0C 03        1890        SHLD  X0               ;STILL HAVE ROOM
C3 D0 04        1900        JMP   NEWSM            ;GO START A NEW SYMBOL
                1910 *
FE 40           1920 ENDLN  CPI   64               ;MAX DIST TO EDGE
DA 28 04        1930        JC    END              ;64>W+PRES POSIT.
6F              1940        MOV   L,A
22 0C 03        1950        SHLD  X0
C3 D0 04        1960        JMP   NEWSM
                1970 *
21 0C 03        1980 END    LXI   H,X0
36 00           1990        MVI   M,00             ;ZERO X0
23              2000        INX   H
36 00           2010        MVI   M,00
21 0E 03        2020        LXI   H,Y0
3E 1E           2030        MVI   A,30             ;HEIGHT OF SYMBOL
86              2040        ADD   M
FE C8           2050        CPI   200D             ;BOTTOM OF SCREEN
CA C7 04        2060        JZ    START
77              2070        MOV   M,A
C3 D0 04        2080        JMP   NEWSM
                2090 *
34              2100 ADD3   INR   M
34              2110 ADD2   INR   M
34              2120 ADD1   INR   M
C3 5B 04        2130        JMP   XY
                2140 *
35              2150 SUB3   DCR   M                ;M IS NOW DIR REG
35              2160 SUB2   DCR   M
35              2170 SUB1   DCR   M
C3 5B 04        2180        JMP   XY
                2190 *
3E 01           2200 CHGPU  MVI   A,01
32 0B 03        2210        STA   PEN
```

```
C3 E6 04        2220         JMP   CONT
                2230 *
3E 00           2240 CHGPD   MVI   A,00
32 0B 03        2250         STA   PEN
C3 E6 04        2260         JMP   CONT
                2270 *
                2280 *
                2290 *
                2300 * XY  RON PALMER  AUGUST 1978
                2310 *
                2320 *TAKES DIREC REG & STORES IN XPOS..XL/SH/Y
                2330 *X=FF
                2340 *Y=0 MUST BE INITIALIZED FOR EACH NEW SYMBOL
3A 05 03        2350 XY      LDA   X
47              2360         MOV   B,A
3A 06 03        2370         LDA   Y
4F              2380         MOV   C,A
3A 0A 03        2390         LDA   DR
C6 08           2394         ADI   08          ;GET RID OF NEG
E6 07           2396         ANI   07          ;TRUNCATE
FE 07           2400         CPI   07
CA 9F 04        2410         JZ    XY7
FE 06           2420         CPI   06
CA A0 04        2430         JZ    XY6
FE 05           2440         CPI   05
CA 9A 04        2450         JZ    XY5
FE 04           2460         CPI   04
CA 9B 04        2470         JZ    XY4
FE 03           2480         CPI   03
CA 95 04        2490         JZ    XY3
FE 02           2500         CPI   02
CA 96 04        2510         JZ    XY2
FE 01           2520         CPI   01
CA 90 04        2530         JZ    XY1
C3 91 04        2540         JMP   XY0
                2550 *
0D              2560 XY1     DCR   C           ;Y UP-RIGHT
04              2570 XY0     INR   B           ;X RIGHT
C3 A1 04        2580         JMP   XYEND
05              2590 XY3     DCR   B           ;LEFT UP
0D              2600 XY2     DCR   C           ;UP
C3 A1 04        2610         JMP   XYEND
0C              2620 XY5     INR   C           ;LEFT
05              2630 XY4     DCR   B           ;LEFT
C3 A1 04        2640         JMP   XYEND
04              2650 XY7     INR   B           RIGHT-DOWN
0C              2660 XY6     INR   C
78              2670 XYEND   MOV   A,B         ;B CONTAINS X
32 05 03        2680         STA   X
79              2690         MOV   A,C         ;C CONTAINS Y
32 06 03        2700         STA   Y           ;Y POSIT RDY FOR DISPL
21 0E 03        2710         LXI   H,Y0
86              2720         ADD   M
32 6C D0        2730         STA   0D06CH      XPOS+2
48              2740         MOV   C,B
06 00           2750         MVI   B,00
```

```
2A 0C 03       2760          LHLD  X0        ;X+X0 > XPOS
09             2770          DAD   B         ;ADD BC AND HL
22 6A D0       2780          SHLD  0D06AH    XPOS
3A 0B 03       2790          LDA   PEN       ;CHECK PEN STATUS
FE 00          2800          CPI   00
CC B8 CD       2810          CZ    0CDB8H    ;DISPE...DISPLAY
C3 E6 04       2820          JMP   CONT
B8 CD          2830 DISPE    DW    0CDB8H
               2840 *
               2850 *
               2860 *
               2870 * START AUG 78
               2880 *
CD 14 03       2890 START    CALL  INIT
CD 38 03       2900 STAR1    CALL  DISPL
CD 49 03       2910          CALL  CLEAR
CD EA 04       2920 NEWSM    CALL  SCAN1     FETCH SYM ADDR CODE
3E 01          2930          MVI   A,01
32 0B 03       2940          STA   PEN       PEN UP
3D             2950          DCR   A
32 07 03       2960          STA   CODE
32 06 03       2970          STA   Y
32 0A 03       2980          STA   DR
3D             2990          DCR   A         -1
32 05 03       3000          STA   X
C3 AB 03       3010 CONT     JMP   DIREC     DISPLAYS SYMBOL
               3020 TIM      DS    1
3E 00          3030 SCAN1    MVI   A,00
32 00 03       3040          STA   XSCAN
3E 7F          3050          MVI   A,7FH
D3 FF          3060 SCAN2    OUT   0FFH      FRONT PANEL LEDS
F5             3070          PUSH  PSW
06 08          3080          MVI   B,08
3E 7F          3090          MVI   A,07FH
32 E9 04       3100          STA   TIM
CD FC C3       3110 DEL1     CALL  0C3FCH    DIMS..DELAY
CD E8 C1       3120          CALL  0C1E8H    KSTAT-KEYBRD STATUS
C2 19 05       3130          JNZ   TRIG
3A E9 04       3140          LDA   TIM
05             3150          DCR   B
C2 FB 04       3160          JNZ   DEL1
3A 00 03       3170 ADVAN    LDA   XSCAN
3C             3180          INR   A
E6 07          3190          ANI   07        ONLY COUNT TO 8
32 00 03       3200          STA   XSCAN
F1             3210          POP   PSW
0F             3220          RRC
C3 F1 04       3230          JMP   SCAN2
F1             3240 TRIG     POP   PSW
3A 00 03       3250          LDA   XSCAN
FE 07          3254          CPI   07        ;JUMP TP XEK
CA 04 2A       3256          JZ    2A04H     ; ON SCAN OF 8
07             3260          RLC   X2        2 BYTES/ADDR
2A 12 03       3270          LHLD  TABLE
85             3280          ADD   L
6F             3290          MOV   L,A
```

(

```
0528 7E                 MOV   A,M
0529 32 08 03           STA   SYMB
052C 23                 INX   H
052D 7E                 MOV   A,M
052E 32 09 03           STA   SYMB+1
0531 C9                 RET
0532               *
0532               *
```

SYMBOL TABLE

ADD1	0441	ADD2	0440	ADD3	043F	ADVAN	050B	BITS	035D
CHGPD	0453	CHGPU	044B	CLEAR	0349	COD0	037F	COD1	0387
COD2	038D	COD3	0395	CODE	0307	CONT	04E6	DEL1	04FB
DIREC	03AB	DISPE	04C5	DISPL	0338	DR	030A	END	0428
ENDLN	041C	ENDSY	03EC	INIT	0314	NEWSM	04D0	NEXT	039F
PEN	030B	SCAN1	04EA	SCAN2	04F1	STAR1	04CA	START	04C7
STORD	035A	STORN	035C	SUB1	0447	SUB2	0446	SUB3	0445
SYMB	0308	TABLE	0312	TIM	04E9	TRIG	0519	W	0304
X	0305	X0	030C	XOUT	0302	XPOS	030F	XSCAN	0300
XY	045B	XY0	0491	XY1	0490	XY2	0496	XY3	0495
XY4	049B	XY5	049A	XY6	04A0	XY7	049F	XYEND	04A1
Y	0306	Y0	030E	YOUT	0303	YSCAN	0301		

REFERENCES

1. Bliss, C.K., Semantography(Blissymbolics) 1965 Semantography (Blissymbolics) Publications, Sidney, Australia.

2. Bliss, C.K., Mr. Symbol Man produced by National Film Board of Canada and Film Australia.

3. Storey, J.R., "Bliss-Symbol Display Terminal" Communications Technology & Equipment Research Image Communication #76-12 March, 1976, CRC publication. Ottawa, Ontario.

4. Vanderheiden, G.C., "Development of a Bliss Symbol Communication Aid" February 1977, Cerebral Palsy Communication Group, University of Wisconsin-Madison.

5. "Aids to Handicapped", Medical Engineering Section, Ottawa Crippled Children's Treatment Centre National Research Council, Ottawa, Ontario.

6. Luukko, R., "They Use Symbols To Talk", The Regina Leader-Post, June 12, 1978, Regina, Saskatchewan.

7. Halifax (CP) "Symbols Used for Communication", The Regina Leader-Post, Oct.25, 1977, Regina, Saskatchewan.

8. Abramson, N., Information Theory & Coding pp 77 McGraw Hill Book Company, New York. (1963)

9. Bown, H.G., "PDI for the Canadian Videotex System" Communications Canada, Nov.30, 1978, Ottawa, Ontario, Canada.

10. Bown, H.G., et al, A General Discription of Teledon: A Canadian Proposal for Videotex Systems, CRC technical note No. 697-E, Ottawa, Ontario.

11. Sawchuk, W. and Bown, H.G., Interactive Graphics Applied to Symbol Communication CRC, Box 11490 Station H, Ottawa, Ontario, Canada.

12. "Visual Telephone" Could Help Handicapped Communicate JOUR 60 DAYS #12 July 1976, Department of Communications, Ottawa, Ontario, Canada.

13. Information pertaining to Bliss Symbols. Blissymbolics Communication Foundation, 862 Eglinton Ave.E, Toronto, Ontario.

14. Vanderheiden, G. et al, "Cost Reduction in the Development of New Communication Aids Through the Use of Common Control Systems: A Case Example the Blisscom.", University of Wisconsin, Madison, Wisconsin.

15. Newman, W.M. and Sproull, R.F., Principles of Interactive Computer Graphics 1973, McGraw Hill New York.

16. Walker, B.S., Gurd, J.R. and Drawneck, E.A., Interactive Computer Graphics 1975, Crane, Russak & Co. New York.

17. Shwedyk, E., Gordon, J., "Communication Aid for Non-Vocal Handicapped Children" Med. & Biol. Eng. & Comput. 1977, 15 189-194.

18. Molinder, J.I., "Optimal Coding with a Single Standard Run Length" May 1974, IEEE Transactions on Information Theory Vol. 1T-20, No.3.