We make deep impression moldings of the mouth by using two to three slabs of dental wax (Kerr impression compound), which are fitted onto a 3/16-in. aluminum plate biteboard.

6. It would be possible to record (in noncontingent situations) the EPS off-line on a multichannel FM instrumentation tape recorder and subsequently play the signals back to the computer through A-D channels for analysis. One FM channel would be needed for each EPS input, and several channels would be needed for recording event markers, calibration pulses, and so forth.

7. The assistance of David H. VanDercar in designing and building the biteboard holder and attached XYZ movement is gratefully acknowledged.

8. The success of the biteboard apparatus in restricting head movements is readily determined by requesting the S to attempt head movements while maintaining a fixation point and observing correlated changes in the EPS.

9. This will be the case when the tape (or disk) control utilizes a cycle-stealing data-break facility.

10. If the sampling rate is so high that more than N words will be output in an interval during which a single physical block is output onto the mass storage device, more than $2N$ words of

buffer storage will be required. In this case, the necessary modifications of RECORD are straightforward.

11. These can be averaged or otherwise smoothed over the sequence of samples taken during the fixation interval.

12. The thresholds are most readily determined in units of necessary increase or decrease in X- and Y-eye-position voltages.

13. The instantaneous velocity is estimated by comparing two or more successive EPS samples in terms of angular displacement information from the calibration matrix.

14. The speed at which displays can be changed also depends on persistence characteristics of the phosphors used. In our display, low-persistence P-31 phosphors are used. Once again, however, the same tradeoff appears: low-persistence phosphors must be refreshed more frequently to avoid flicker in the display.

15. If a standard 8-bit ASCII code is used to represent characters, even a 9,600-baud rate permits a maximum of 1,200 characters/sec, which limits the rate at which a display can be changed to a little more than one new character every millisecond.

16. The Biometrics circuitry (with filter "in"), for example, induces a signal delay of approximately 20 msec.

Software character generation*

CLIFFORD B. GILLMAN and PAUL BUCKLEY

University of Wisconsin, Madison, Wisconsin 53706

Programming considerations are described for computer-controlled visual displays. Software character generation is emphasized.

The cathode ray terminal (CRT) is an excellent general-purpose device for the display of visual stimuli in psychological research. This paper is concerned with the programming considerations involved in computer-controlled visual displays. Criteria for choice of a CRT have been discussed by Sperling (1971) and Van Gelder (1972). Visual display routines may be used to present strings of alphanumeric characters, text, geometric forms, dot patterns—any visual stimuli. Character generation facilities and display algorithms must be chosen on the basis of the stimuli to be presented. The optimal strategy is ultimately determined by the stimulus material.

The primary decision must be made between hardware and software character generation. Hardware character generators allow very rapid presentation of material, since only a stimulus code must be transmitted from the computer. The character generator is wired to transform this code to the actual stimulus representation. Frequently, additional electronics for automatic refreshment are included, so that central processor involvement is minimized. Speed and automatic refreshment are usually achieved by limiting the stimulus set to the alphanumeric characters. Such CRTs have been called "electronic typewriters." It is impractical, if not impossible, to expand the stimulus set beyond the standard 96 ASCII characters. If this

constraint is acceptable, hardware character generation is optimal. However, generality requires that we discuss the possibility of displaying any visual stimuli that can be drawn, and to do this requires that the character generator be programmable. Software character generation allows maximum flexibility in choice of stimuli at the cost of presenting fewer characters per second. For programmed character generation, a cathode ray oscilloscope (CRO), with suitable interface to a laboratory computer, is necessary.

An additional consideration is the availability of a storage oscilloscope. Nonstorage oscilloscopes require the E to choose a combination of refreshment rate and phosphor decay so that the desired stimulus array is presented without flicker. This commits the central processor to maintaining the visual display. This problem does not exist for storage oscilloscopes, but the storage process produces other potential problems. Erasure of a storage oscilloscope is performed by flooding the entire screen with light, which requires approximately 500 msec to subside. This severely limits the E's ability to present several stimuli in rapid succession and makes dynamic displays impossible. It is also impossible to erase a portion of the display selectively. Although technological developments promise to alleviate these problems while improving the brightness and resolution of storage oscilloscope displays, generality and flexibility limit this discussion to the standard nonstorage oscilloscope.

The problem is to present and refresh a number of characters as rapidly as possible while maintaining acceptable character detail. However, speed and resolution are but two of many criteria for design of software character generation subroutines. It is also desirable that the E be able to define new stimuli without changing the existing subroutines. The subroutines must be easily called and must occupy as

*This work was supported by Grant MH 19006 from the United States Public Health Service.

little memory as possible. For compatibility among programmers and computers, the alphanumeric character codes should correspond to the ASCII standard.

All available character generators stem from a common approach. The generation routine accepts as an argument a code which specifies the character to be generated. By a table look-up procedure, this code is transformed to a series of coordinates or line segments. For ease of discussion, these will be called coordinates, but strokes or vectors may be substituted without loss of generality. Character display routines transmit the coordinates to the display device and repeatedly refresh the display to maintain a constant image. The character generation algorithm must be chosen to minimize the memory requirements of the generation subroutine, any necessary storage buffers, and the display subroutines. Control of location and scaling gives the E complete command of stimulus format. The major differences among algorithms are in the way the coordinates are used.

One class of generation algorithms stores a character as a pattern of ones and zeros at the intersections of a grid. A dot is stored as a one and its absence as a zero. To store a character defined as a 6 by 6 matrix requires only three 12-bit words. By reassembling the points in a fixed order, one can generate or recognize a character. This technique is very common in optical character recognition of English letters and numerals (Uhr & Vossler, 1963). However, complicated characters may require large quantities of memory for adequate resolution; for most characters, this approach is adequate.

One straightforward software character generation procedure has been programmed by L. T. Gell for the PDP-8 (DECUS 5/8-23B). This program generates alphanumeric characters by illuminating points in a 5 by 7 matrix. Thirty-five bits of information are required, each bit specifying whether or not one particular dot is to be illuminated. Three 12-bit words hold all the necessary information for the presentation of one character. A description of this algorithm is also presented by Haber (1971).

Three entry points to this routine are available, to allow presentation of text strings, character pairs, and single characters. In all cases, the x- and y-coordinates for the lower left corner of the first character to be presented must be set before the routine is called. In text mode, the routine is passed the address of the first word of a buffer containing the text in packed, stripped ASCII. The buffer is terminated by the delimiter 0001. String presentation involves successively reading pairs of ASCII characters and passing them to the character pair presentation routine. This routine, in turn, merely unpacks the ASCII code and passes the resulting stripped ASCII for single character presentation.

Character display is accomplished by two sets of interlocking loops. The first of these selects the appropriate stimulus representation using the stripped

ASCII to access the appropriate three-word buffer of bits. Each word is then rotated to interrogate the bit string 1 bit at a time. The second set of loops maintains beam position in the 5 by 7 array. Initially, two pointers are set to the x- and y-coordinates for the lower left dot in the matrix. As bit strings are rotated in the accumulator, these pointers are incremented proportional amounts, moving from bottom to top and from left to right. After all points in a character have been interrogated, the y-coordinate is reset to its initial value and the x-coordinate is increased so that, if these values are not reset, the next call to the subroutine will produce a character on the same line, but immediately following the one presented. In other words, successive calls will present characters in succession along a line. Besides the end-of-string special character, four others are provided to space, perform a carriage return, perform a line feed, and perform both of the latter two. Optionally, the E may control the size of the characters generated.

Several basic problems arise in connection with this routine. Only one presentation of a character is made per call to the routine, so that repeated calls must be made to the routine to maintain the symbols on the screen. Since coordinates for presentation are computed each time the routine is called, a single character requires approximately 1.5 msec for presentation. Thus, displaying large amounts of text without flicker requires a long-persistence phosphor or a storage oscilloscope. The limited character set may be overcome simply by addition to the bit string buffer.

The routine developed in Professor W. K. Estes's laboratory is basically an improvement of the Gell programs. Thirty-six bits of information for characters in a 6 by 6 array are stored in three 12-bit strings. However, the appropriate bit string is accessed and decoded only once. A call is made to a buffer-building routine, which transforms a stripped ASCII text string to coordinates and stores them. For actual display of the characters, this buffer is repeatedly read serially and the coordinate pairs presented. The display remains on until some action is taken to terminate it.

Although some optimizing is done, this set of routines sacrifices efficiency in the use of memory to maximize the speed of character display and minimize flicker. Gell's program stores only packed, stripped ASCII, while the Estes program maintains two large buffers. Each word of the Estes text buffer contains the stripped ASCII for only one character, thus doubling the size of the Gell text string buffer. The Estes coordinate buffer takes advantage of column-by-column decoding to set an x-coordinate once per column. A y-coordinate is set for each dot displayed. The advantage of this system is evident in characters with several vertical lines, such as "H," "M," or "N." In spite of this, approximately 20 words of memory are required in the coordinate buffer for each word in the text string buffer. Thus, 40 times as much memory is required for the Estes algorithm.

However, presentation time per character is only about 250 microsec. A message containing 128 characters would require 64 12-bit words of memory for Gell's program, but occupies 2,560 words of core for Estes's. A single presentation of this message requires about 200 msec using the Gell program and 32 msec for the Estes routine.

The disadvantages of the Estes approach may be easily corrected. One approach is that of Yang and Yang (1969), who developed a universal graphic character generator for all languages. Each character is a single independent graph, decomposed into line segments within a 16 by 16 grid for nonalphabetic languages and a 5 by 8 grid for alphabetic languages. A character is represented by coordinates which indicate the beginning, turning points, and end point of each line segment. For example, the letter "M" is expressed by five pairs of coordinates. Each coordinate requires no more than 4 bits, so that the entire upper- and lowercase English alphabet can be stored in 440 12-bit words. All starting points, turning points, and end points must be at the intersections of the grid. The use of turning points was developed to avoid storing the end point coordinates a second time as the beginning point of a connected line segment. The corresponding display routine simply draws straight lines between coordinates. The electron beam is turned on at a starting point and remains on until an end point is reached. Yang and Yang (1969) specified the scale to the display routine in order to minimize the coordinate buffer size, and this limits display speed. Using this algorithm, they were able to store all 10,000 Chinese characters in a 65K computer and generate Chinese manuscript on a Calcomp plotter. Drawing line segments avoids storage of intermediate coordinates along a straight line in any orientation.

There are advantages and disadvantages to each of the approaches described. The Gell routine minimizes memory requirements at the expense of speed. The Estes approach maximizes speed at the expense of memory. The Yang algorithm allows display of complex characters with acceptable resolution by optional expansion of the size of the character grid. The Yang storage scheme is efficient at the expense of display speed, although presentation rate may still be acceptable. All three routines allow expansion of the character set to nonalphabetic characters, and all are sufficiently general for most experimental purposes. The Yang routines were written in FORTRAN and may be used on either oscilloscopes or plotters. By substituting appropriate oscilloscope instructions for those which control the plotter pen, other graphics routines may be adapted for the display of visual material in the experimental laboratory. However, plotting routines generally optimize memory requirements at the expense of presentation rate.

The demands of particular sets of stimuli may dictate use of different generation schemes for different experiments. It is not sufficient to develop a single display routine and allow its properties to dictate the research questions to be investigated. A general flexible character generation package should be available in every experimental laboratory. Such a collection of routines would have the following properties: (1) The E specifies the stimulus set prior to the experiment. Character codes are assigned and coordinates or vectors are generated at this time. Provision for light pen input and dynamic character cleanup are desirable features. (2) The E specifies the desired size of his memory buffer for stored coordinates, the display size, and refreshment rate. The optimal character generation scheme would then be selected from among several alternative algorithms. (3) Matrix size is determined at compile time, but provision must be made for dynamic rescaling of individual characters at execution and simultaneous checking for overflow and wraparound. (4) Stimulus grid size must be variable, to optimize speed and clarity.

A stimulus display package of this sort could present text material efficiently at one stage of the experiment and later display complicated geometric forms, masking stimuli, or graphs. The implications are immense for research in visual psychophysics, cognitive psychology, and information processing. Such a character generation package is currently being developed in our laboratory. The usefulness of this new display package may be short-lived. Research in machine-character recognition promises better generation algorithms. For example, feature analysis algorithms may identify salient features for character construction. Further, optimization theory has much to contribute to reduced compilation and execution time. Finally, new two-dimensional languages have been developed to ease communication with the computer (Wells & Morris, 1972). These languages may revolutionize the way we conceptualize the material to be presented.

REFERENCES

- Haber, R. N. An alphanumeric display program for a PDP-8. *Behavior Research Methods & Instrumentation*, 1971, 3, 141-145.
- Sperling, G. Flicker in computer-generated visual displays: Selecting a CRO phosphor and other problems. *Behavior Research Methods & Instrumentation*, 1971, 3, 151-153.
- Uhr, L., & Vossler, C. A pattern-recognition program that generates, evaluates, and adjusts its own operators. In E. A. Feigenbaum and J. A. Feldman (Eds.), *Computers and thought*. New York: McGraw-Hill, 1963.
- Van Gelder, P. CRT displays in the experimental psychology laboratory. *Behavior Research Methods & Instrumentation*, 1972, 4, 102-103.
- Wells, M. B., & Morris, J. B. (Eds.) *Proceedings of a symposium on two-dimensional man-machine communication*. SIGPLAN Notices, 1972, 7(10).
- Yang, S. C., & Yang, C. W. A universal graphic character writer. Paper presented at the Third International Conference on Computational Linguistics, Stockholm, September 1969.