THE DESIGN OF INTERACTIVE SYSTEMS FOR THE CREATION OF PRECISION GRAPHIC PRIMITIVES

W. D. Hoskins
G. E. McMaster
Departments of Computer Science,U.of Manitoba,Brandon U. respectively.

ABSTRACT

The widespread availability of inexpensive microcomputers with a medium resolution graphics capability, enables the user to imagine the creation of program systems which have the capability of using large sets of non-standard characters such as the Greek alphabet, Cyrillic or Gothic character sets, to display data, text or program results on the graphics screen.

An early approach to obtaining alternative character sets via modification of standard hardware is outlined.

The subsequent design, implementation and use of a program system is examined and the appropriate interaction and state transition descriptions of the systems are detailed.

KEYWORDS: Graphics Shapes, Cree, Arabic

1. Introduction

Early in 1981, we received a request from the Native Studies department at Brandon University, to create the Cree alphabet on a microprocessor for communication with native studies students and subsequent printing of text. Cree is the Muskagean language of the Cree Indian, and it consists basically of 42 alphabetic-like characters and 13 punctuation-like characters (Figure 1). Using an EPROM blaster, an EPROM chip was created to replace the character generator chip on a PET 2001 microprocessor and give the Cree dictionary through the use of the upper case character set. The remaining Pet characters were left unmodified for normal communication.

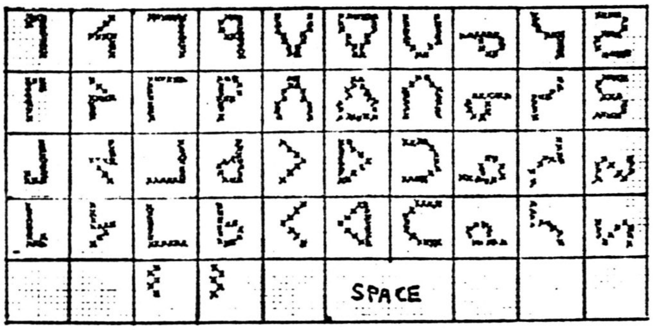

Figure 1.1
Cree Characters: initial design form before transferral of the bit patterns to the Character generator chip.

The modification was a success in that a workable Cree alphabet was generated. Because of the limitations of the small 8 x 8 bit matrix used to represent each character (and which must include space between characters) a design was not achieved that was completely satisfactory to the native studies department. The symbol ~ for example, had to be smaller than the symbol S. In other words, there was little room for the design and creation of the shapes that best represented the Cree symbols in a clear and unambiguous fashion and with the necessary pleasing artistic interpretive implementation. The problem of designing a suitable font is a difficult task and should be done interactively, where changes can be made and immediately observed. This problem is amplified if one is attempting to create the Arabic character set, which additionally is cursive in nature. This means that a great deal of precision is required so that the characters can flow smoothly together.

One solution to the problem is to design graphic primitives and then simply manipulate a page of text as a graphic entity. This creates the need for the designer to be able to create or specify an appropriate set of graphic primitives so that a high level language or text editor can consider the primitives as a set of basis elements for the graphics space considered. This space will vary from application to application but is not likely to consist of merely straight lines and circles, and will comprise only the set necessary for that users application. An additional constraint placed on the actual set of primitives is that it is unlikely that freehand sketching of their form is adequate (as is the case with the Arabic character set) and it is much more likely that the parts of the primitives have a precision arrangement with each other. This has, of course, implications for the actual system used to construct the primitives as does the observation that it must be possible for the primitives to be **visible** both at a 1:1 scale and at an enlarged scale so that precise pixel to pixel arrangements and relationships can be established. This latter point, of course, forces the need for implementation of windowing in the system for producing the precision primitives, if they are so large, that at the scale selected for creation, more space is necessary (This essentially was the problem that we encountered with the Cree alphabet).

The ability to create graphic primitives is offered by the Minnesota Educational Computing Consortium [5] in the shape table Editor facility. The design of shapes (for example, a character set) is done free hand, or if more accuracy is required, then it is suggested (i) that the design be printed on the screen and then the specific program tables and its associated commands are used to follow the pattern or (ii) that a graph transfer be carried out, that is, you use graph paper to design the shape, and then follow the squares with the appropriate commands. This does not permit ease of design nor does it facilitate the requirement that precise relationships be established with one graphic primitive relative to the others.

An additional point to be made is that the sudden availability of low-cost dot addressable graphics printers has meant that these visual displays could be used to create actual hard copy versions of the graphics, and hence character fonts etc. Other possible application areas would involve teachers demonstrating circuit design and elementary animation techniques to students. Both of these latter problems additionally require a need for permanent copies of the graphics screen.

To implement the design of precision graphic primitives, we used the following window (Figure 1.2) with the actual size appearing to the right.

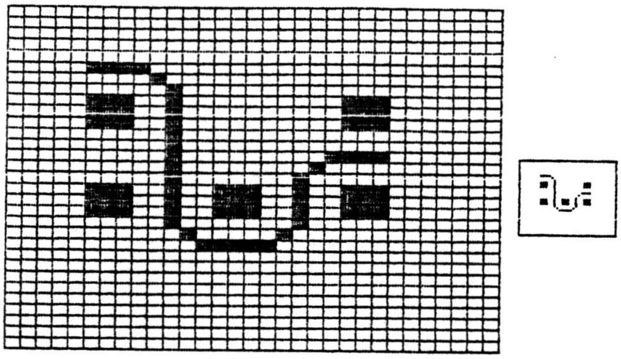

Figure 1.2
Primitive in the Design window
and at a 1-1 scale.

The primitive situated in the window can be added to in any direction with the primitive moving away from the information being inserted. The grid system allows the user to accurately specify the desired primitive which can then be placed in the shape table with its associated position in the grid system.

2. Description of the Shape Table Editor and Graphic Entity Editor.

The data structure appropriate for the representation of the graphic shapes, depends partially on the way that the high level language is to manipulate the primitives. Typically, on both the IBM PC [4] and the Apple-II plus microcomputers, specification of the graphic entities use chain encoding [1], and hence the set of operations that an interactive graphics system has to have so that the user is able to design, edit, display, print and merge files of graphic primitives, must be able to cope with this facility.

Figure 2.1, gives the complete module hierarchy of the application program, that was implemented on the APPLE II Plus, so that the structure of the shape table editor can be understood (Foley and Van Dam [1]). Many of the options displayed are similar to those offerred by the MECC [5] system , but several additional modules are added that enhance particular requirements such as character font creation and later processing by a Graphics Entity Editor.
Figure 2.2 gives the state transition graph for the user interaction with the Shape Table editor.

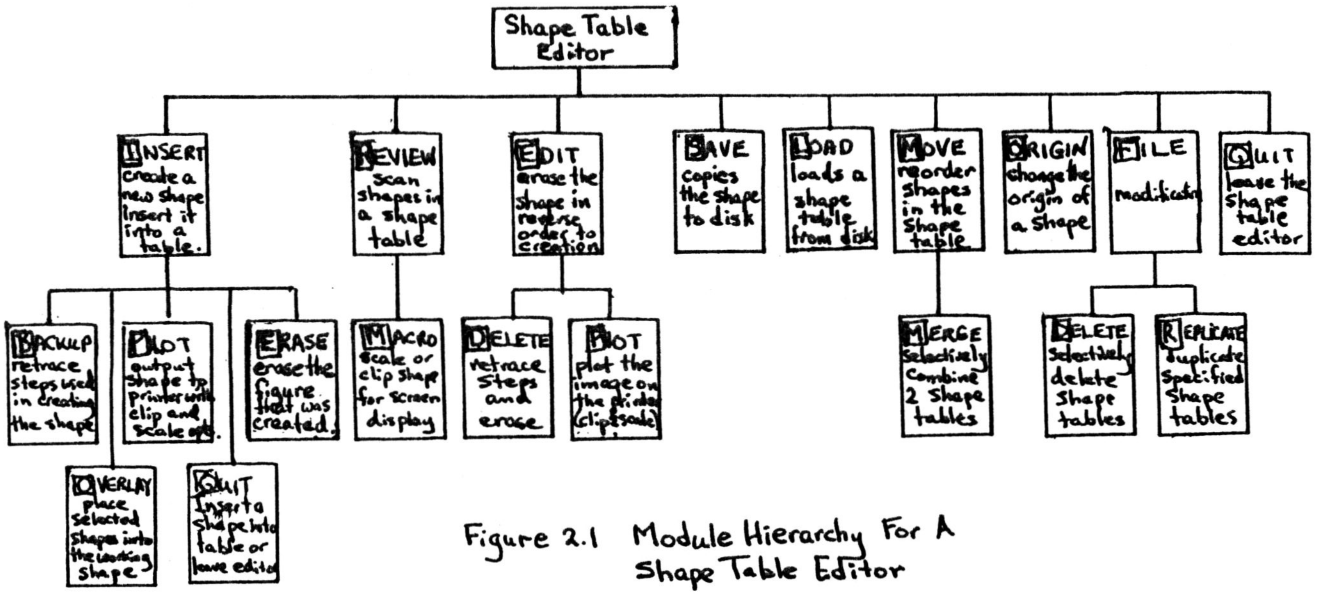
Shape Table Editor
INSERT create a new shape insert it into a table.
REVIEW scan shapes in a shape table
EDIT erase the shape in reverse order to creation
SAVE copies the shape to disk
LOAD loads a shape table from disk
MOVE reorder shapes in the shape table
ORIGIN change the origin of a shape
FILE modificatn
QUIT leave the shape table editor
BACKUP retrace steps used in creating the shape
PLOT output shape to printer with clip and scale opts.
ERASE erase the figure that was created.
MACRO scale or clip shape for screen display
DELETE retrace steps and erase
PLOT plot the image on the printer (clip&scale)
MERGE selectively combine 2 shape tables
DELETE selectively delete shape tables
REPLICATE duplicate specified shape tables
OVERLAY place selected shapes into the working shape
QUIT insert a shape into table or leave editor

Figure 2.1 Module Hierarchy For A Shape Table Editor

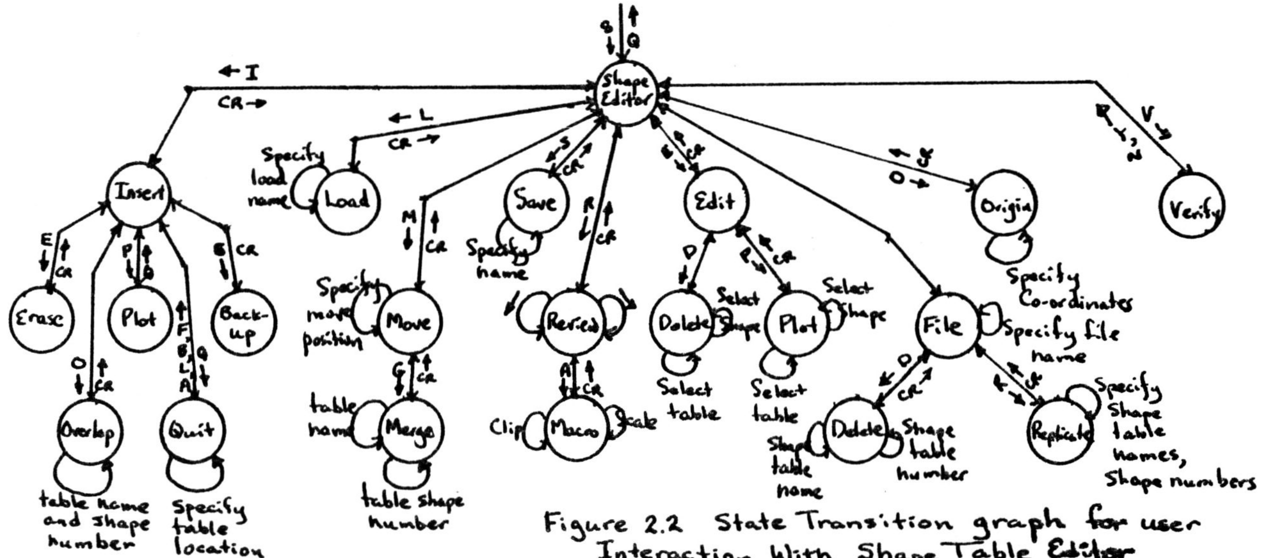

Figure 2.2 State Transition graph ~~for~~ user Interaction With Shape Table ~~Editor~~

Legend: F: first, B: Before, L: last, A: after
→: move forward, ← move back

Once the particular shapes are defined, they can be used by a Basic or Pascal program in the usual manner (MECC [5]). In our particular application of creating character fonts for languages such as Cree or Arabic, the shapes are received by a graphic entity editor. The page is then effectively divided up into the equivalent of tiles dependent on the size of the character shapes, the resultant text is formatted and then placed on a page with the individual characters assuming the tile role.

3. Conclusion

The shape table editor was successfully used with a large first year University computer science class to illustrate animation techniques.

The shape table editor and the graphics entity editor is a viable option for the Native Studies Department for writing and printing text in the particular language chosen. The relatively low cost of the equipment in conjunction with the high quality resolution in the displayed and printed output makes the shape table option very attractive. Additionally, each author can modify the display to effect their interpretation of the form of the characters in the given language. Additionally, labels with the particular character set are easily attached to the keyboard in the usual APL manner to assist the operator.

References

[1] Foley, J. D., Dam, A. V., "Fundamentals of Interactive Computer Graphics", Addison-Wesley, 1982.

[2] Rosenfeld, A., Avinash, C. L., "Digital Picture Processing", Academic Press, 1976.

[3] Apple-soft Basic, Apple Computer Corporation, 1979.

[4] Basic for the IBM Personal Computer, Microsoft, IBM, 1981.

[5] Shape Tables, Volume 1. Minnesota Educational Computing Consortium, MECC publications, June, 1981.

Acknowledgment

Dr. J. Rice,Department of Physics,Brandon University,for the hardware modification of The PET to provide the C ree alphabet.