Although some Medieval and Renaissance writers extolled the intellectual and spiritual benefits of copying manuscripts, such an attitude is less common now. Today's musicologists often face the task of copying passages of early music in its original notation, but cannot resist wishing for a deus ex machina to liberate them from such toil. Buxton et al. (1980) cite major projects in automated music editing, to which may now be added Rumery (1981). Reproducing good quality notation is a complex problem, so expensive hardware, elaborate programs, or sophisticated encoding systems characterize this kind of work. In most cases, the musicologists' needs mentioned above do not seem to justify such investments of time and resources; our common need is merely to reproduce Gregorian chant, incipit catalogues, and illustrative excerpts in diplomatic form. Gregory's Scribe was conceived as an experimental step toward satisfying this need.

The musical input employs interactive graphics, a process perhaps more expedient and accurate than alpha-numeric code. The program operates on modest equipment: an Apple II Plus, video monitor, graphics tablet (bit pad), one 5 1/4 inch disk drive, and a Malibu graphics printer. The program is stored on one disk, and enough storage remains on that disk for about 20 staves of music. Storage can be increased by adding disks.

Music notation before 1600 is well suited to dot matrix graphics since this notation, unlike modern notation, consists largely of straight lines rather than curves and circles. One simple way to reduce crudities in the graphics is to make the notes large, then use photo-reduction. In interactive graphics, however, the usefulness of photo-reduction is limited; larger note shapes complicate editing because fewer notes can appear on the screen at one time. Striking an appropriate compromise between the user's convenience and the quality of output is a well-known problem.

Another compromise involves the nature of the input. Should the user choose from a vast array of character sets, or should there be perhaps only five choices: a straight line, a square, a diamond, a draw-function, and the inevitable eraser? Gould (1967) analyzed components of the Solesmes chant notation, and the reduction of early music notation to common basic forms is carried further in planning Gregory's Scribe. In order to enhance the user's efficiency and also the attractiveness of consistency in the output, Gregory's Scribe contains algorithms for prefabricated common designs and spacings. However, algorithms are not
attempted for all possible needs, because Gregory's Scribe aspires to reproduce most forms of Medieval and Renaissance monophonic and polyphonic notation. Those notation systems were handdrawn and not always standardized. Interactive graphics can preserve elements of handcraftsmanship, thereby retaining the versatility enjoyed by the original scribes.

Operating the Program

This is a layman's program, intended for scholars and students lacking backgrounds in computing. In order to help simplify learning the program, operations are designed in imitation of common file editing processes. Upon loading the disk, the user is automatically entered into file mode. Commands in this mode permit the user to list the named notation files already created on the disk, to create or delete files, to display filed music on the monitor or printer, to stop, or to enter edit mode for a particular file.

Edit mode permits copying one page (a frame, or screen, with two staves) of a particular file to a new page; deleting, hardcopying, or displaying a single page; renumbering file pages; and returning to file mode. Two other commands allow the user to alter or insert a page; these two commands switch the user to graphics mode.

The file control directives described above are given at the keyboard, but most of the commands in graphics mode are entered with the graphics pen and tablet. These commands appear as a menu at the top of an acetate overlay. This overlay, which also bears two staves corresponding to the two on the screen, is taped to the tablet. The menu consists of two rows of command boxes. Some of the boxes enact functions as described below.

Clear  Clears the entire page.
Erase  Changes pen color to black.
Reset  Returns pen color to white.
Dots   Makes a single dot.
Draw   Activates freehand drawing.
Erase Box Erases an area defined by its diagonal corners.
4-line  Makes two 4-line staves.
5-line  Makes two 5-line staves.
Caption Activates ASCII characters (for titles or text overlay).
Stop   Saves the page and returns the user to edit mode.

Other boxes contain the character sets given in Illustration I at the top of the next page. Using the graphics pen, the user presses the appropriate box for a character set or a function. If the box contains a character set, the user then presses the desired position on the staff. Self-correcting error checking rounds off the vertical axis to the nearest space or line and adds ledger lines if needed. In order to speed
up the encoding, a set, once selected from the menu, remains active for repeated notes until a different one is chosen. The three notes preceding the barline in Illustration 1 are actually one character set. In plainchant and early polyphony, descending scales are quite common melodic patterns, and they are often notated as a long followed by descending lozenge shapes. Five different character sets were developed in order to encode, with one command, descending scales of two to six degrees.

Illustration 1.

The character sets from the barline to the end of Illustration 1 are variable in size, needing the definition of two points (lengths of lines or sizes of intervals within compound neumes). These are accomplished by two pen presses on the staff. The barline and doublebar are of variable length, but an algorithm for error-checking assures that they will be drawn vertically. The horizontal line following the barline also corrects its axis and varies in length. The next line varies in length, but it connects two points regardless of the axis they form. The angle of the thick oblique neume is self-corrected to 45 degrees in order to enhance its appearance (except for the interval of a second, which requires a unique shape table).

Three other items in graphics mode involve the Apple keyboard. Pressing "B" or "W" sets the program to deliver solid black or white (hollow) note shapes. Another keyboard function concerns the spacebar; depressing it "n" times moves the next symbol "n-1" spaces to the right of the preceding one. If the spacebar is not depressed, a symbol will be drawn wherever the pen is pressed on the tablet. Finally, texts for captions (automatically centered if desired) and text underlay are typed at the keyboard. Using the cursor, upper case and lower case ASCII characters can be precisely placed anywhere on the screen, and text underscoring is also available. If future needs require additional character sets, a simple solution will be to redefine an unused ASCII character.

419
Some Technical Aspects

The computer possesses 64K of memory, 48K being RAM. Additional storage of 140,000 bytes is provided by the single density, single-sided 5 1/4 inch floppy disk. The program is written in an enhanced floating point Basic, APPLESOFT, because this language provides easy access to graphics capabilities. The graphics tablet and its interface contain the necessary machine language routines to read the pen position and place a cursor on the memory mapped 280 x 192 pixel screen. The printer and its interface provide machine language routines to produce hardcopy from the graphics screen.

The program is organized sequentially. The initial mode, file mode, handles all commands pertaining to files (a series of frames). For each file, the program stores a directory containing the number of frames in the file and the number of each frame as stored on the disk. Storing the number of each frame allows nondynamic page numbering. Therefore, inserting and deleting a page does not automatically resumer the frames. Storing the number of frames in a file allows simple chaining onto additional disks when large files are required.

Edit mode is a subset of file mode. All commands in this mode pertain to the one file being edited. In graphics mode, however, commands operate on one specific frame in a file. Most note shapes are not drawn by the program, but are stored in a table and plotted directly on the screen by a command within the program, such as, "draw 5 at 10,10". Other notes, especially two-part ones, use a hybrid method. The invariant portions of the note are stored in a table and the remaining portions are drawn by the program. This latter method is almost as fast for the machine and it simplifies programming, since many pieces of one note shape also can be used for others. The full ASCII character set for captions and text underlay is also stored in a table. Another machine language routine plots characters from this table to the graphics screen.

Conclusion

Although the program is only several months old, four users at Michigan now work with Gregory's Scribe. The projects involve different repertories (trouvere notation, early plainchant notation, and Renaissance music theory), but the program's versatility is passing these tests. Although the users are experienced music copyists, they agree that Gregory's Scribe is faster, neater, and more congenial than the time-honored ink pen.

At the present, Gregory's Scribe is a copyst and nothing more, although plans may develop to write a translator for converting the graphics data into a form which invites programs for analyzing the music. Gregory's Scribe is a stand-alone program, but the Apple is interfaced to
the university's mainframe for transmitting alpha-numeric and graphics
data; this hookup provides quick access to additional storage,
programming languages, and Calcomp plotting. In the meantime, we have
learned one way to take advantage of the efficiency and simplicity of
interactive graphics, users find the software to be convivial, and, while
working with modestly priced hardware, we are producing notation good
enough in quality to meet many musicological needs.

Illustration 2.

GREGORY'S SCRIBE
ME SCRIPSIT.

References

"The Evolution of the SSSP Score Editing Tools." Computer Music
Journal 3(4):14-25, 60.

Elektronische Datenverarbeitung in der Musikwissenschaft, ed. Harald
Heckman. Regensburg: Bosse.


421