The present status of interactive graphic displays in the application environment is reviewed.

Although graphics data processing is still largely experimental, several applications have come into productive use—especially in the areas of data and design analysis. The primary benefit from enhancing such applications with graphic displays is the savings in calendar time.

Before surveying several application areas, user aspects and application characteristics are discussed.

INTERACTIVE GRAPHICS IN DATA PROCESSING

Implementation and usage

by C. W. Day and L. L. Zimmerman

The use of displays in an interactive mode is a strong departure from the conventional typewriter-oriented device. Research into the support of multiple terminals and the use of display terminals in problem solving is still young. The techniques for basic graphic information display and the manipulation thereof have been developed and proved in the university and laboratory environment, as discussed by C. I. Johnson in the first paper of this issue. Industry is now taking on the task of implementation, keeping an eye on the economics of graphic displays in man-machine interaction. The purpose of this paper is to report on the current implementation state of the art.

The graphic display allows a large variety of ways in which its user can form images and interpret his action. This versatility, and thus complexity,[1] has made it necessary for most of today's operational programs in graphics to have their own specific graphics sections. Although the concept of utilizing generalized graphics functions for applications is now gaining acceptance,[2] all programs discussed here exhibit the former structure.

user aspects

On graphic display consoles, the data is presented in its natural form for both input and output purposes. Adding the analyst to the processing loop helps eliminate unnecessary processing. The user can halt the run if he sees that further computation will be fruitless. He can infer from several attempts of the design what other attempts might be less fruitful and thereby steer a more direct path to the eventual satisfactory design.

Many decisions can be made at the display console, thus allowing several separate trials during a single session; the net result is that turnaround time is sharply decreased. Often the thinking done during the design state is such that the definition of the phenomena to be analyzed is incomplete. This is quite similar to a laboratory environment. Engineering analysis done in such an environment is most easily done on-line, allowing definition and solution efforts to take place in parallel.

How much can a man's thinking be enhanced by working at the display console? Can one attach a value to the fact that he does his work faster or to the fact that he works several iterations without interruption? It may be of value that, at this faster speed, he can follow a train of thought farther down the line prior to becoming exhausted. Also, a graphic image in motion, manipulated by the user or the program, often provides added insight by suggesting phenomena not apparent in a static picture.

application characteristics

Some early uses of graphics were too costly to be of any use beyond that of a technical curiosity. Others, however, have had enough merit to point toward several specific characteristics that should be considered as a guide in selecting sound application programs in graphics. In general, many applications have benefited from graphics support because this support can:

- Immediately produce a plot of results for inspection
- Reduce turnaround time between analyses
- Allow more to be accomplished within a fixed time
- Facilitate manipulation of representative design models
- Facilitate selection of the appropriate solutions
- Allow definition and solution efforts to occur in parallel (e.g., providing partial solutions of incompletely defined problems)
- Stimulate insight into problems

Many applications are noticeably enhanced even if only one of these improvements is achieved when using graphics support. For instance, when using the graphics console for a quick look at output scheduled for a plotter, emphasis is on saving calendar time rather than providing more flexibility in manipulation and selection.

Most easily justified for conversion to graphics are programs of an iterative nature, used repeatedly in approaching the solution of a problem. In this case, graphics considerably reduces the turnaround time between iterations. But the display also provides the user with great flexibility in manipulation and selection. Examples of this sort of application are curve fitting and simulation runs.

An example of an evolving utility involves the linking of programs. As an engineering library of programs increases in size, attempts are usually made to tie the programs together in such a manner that the data output of one program serves as input to the next program. Since data formats are usually not compatible, this process is quite involved. The more difficult problem, however, is the one of programming the decisions that must be made between

program runs regarding the data being used. These decisions usually require not so much the user's time as his quick attention and judgment. Providing the user with a graphics terminal to view the data between runs facilitates the amalgamation of many programs. When reducing data, for example, the iterations involve trying one of several methods of data reduction and then passing judgment on the results. Thus, while at the display console, the user desires dynamic access to several types of data reduction programs. This access can be accomplished in one of two ways:

- Generation of programs by scheduling at the graphics terminal. In scheduling a series of jobs, this type of terminal scheduling involves the use of the IBM SYSTEM/360 Operating System and its associated job control language, much as one would use a card reader. On the display console, however, the user is not required to select the next job until he sees the result of the previous job.
- Composite program generation. This type of overall program usually involves some sort of dynamic linking among specialized load modules. Though harder to program, this approach has the advantage of speed at the graphics terminal.

In general, if graphics support is provided for an already useful application program, early payback can be expected and the task of justifying the result is simplified because the output has already been determined to be of value.

application similarity

When surveying the implementation and use of interactive graphics for industrial data processing purposes, it becomes obvious that display console applications are predominantly of the analysis type. In one way or another, the program analyzes a problem and displays the result on the console screen, so that the user can pass judgment on it. The individual engineering analysis applications discussed here are similar in two aspects:

- They arrive at their solutions in one or more iterative processes. An example is thermodynamic analysis. Given the boundary values of temperatures as a function of time around a structure, one can calculate and display internal isotherms for given points in time. In a similar manner, electrical charge or physical stress —both dynamic and static—can be calculated and displayed.
- They frequently use data of a geometrical or topological nature. Examples are plots of two or more variables in two or three dimensions, topological representations of schematics, geometric representations of cross sections of structures, line drawings of large structures, representations of curved surfaces, statistical data plots, and fully dimensioned engineering drawings.

optical design

The use of graphics in optical design enjoyed an early success. The functional characteristics of an optical system can be determined by calculation of the paths of light rays through the system by the digital computer. The input to the calculation includes the geometry of the optical system and the physical charac-

teristics of the optical elements. Among the parameters describing the function of such a system of interest to the designer are the location of focal points and the sensitivity of this location as a function of system parameters. A graphics version of this program utilizes the display console for entering and/or editing the description of the system, as well as for viewing the ray positions anywhere in the system, using the console to simulate a ground glass. The ability to observe more alternatives and the drastic reduction of design time are the benefits derived from this version of the program.

trajectory analysis

Space flight trajectory analysis is being done in two modes: one is the plotting of telemetry data from actual flights, and the other is the plotting of a theoretical flight path based on the analysis of design parameters. The first mode is similar to the one of data reduction discussed earlier. Viewing the data before and after reduction allows the analyst working at the display console to apply his own experience to the problem of quickly choosing the best method. Within any given method, he can also exercise judgment as to which points should be disregarded due to data error. Bringing such expertise to bear makes feasible the use of less expensive telemetering equipment in the actual flights. Since parametric analysis is a highly iterative procedure, the interactive graphics approach provides a considerable savings of elapsed calendar time. Combining these two modes, one can plot the actual versus the theoretical curves and make judgments as to the validity of the model used in the analysis of design parameters. The program could allow for possible later change of the theory upon which the program is based, dynamically creating a better tool for future design purposes.

automated circuit design

Today's primary tool for the automated design of electronic circuits is some form of a network simulator. One such program is IBM's Electronic Circuit Analysis Program (ECAP). Originally, the program was written for a small desk-size computer, the IBM 1620. Watching the results of the analysis appear on the console typewriter, the designer was able to exercise a fair degree of interaction with the program. Many users found the size of the problem they were able to handle with this version of ECAP inadequate for their purposes. Thus, larger versions were written for larger machines at the expense of interactivity. An available graphics version of this program[3] puts the designer back in his interactive role, even for complex problems. Users with a library of models for frequently-used transistors, diodes, and so forth, can use graphics economically for simulating circuit manipulation and oscilloscope viewing, thus saving the time and materials otherwise used to breadboard the circuit. The graphics version of ECAP provides immediate access to the analysis results, thus allowing a quick decision on any necessary input alterations. Having the input in the form of a schematic aids in this decision-making process and also simplifies the procedures involved in modifying the model. Figure 1 shows an application of an early version of graphics ECAP. A typical output of a graphics ECAP-designed circuit is shown in Figure 2.

Figure 1 Application of an early graphics ECAP

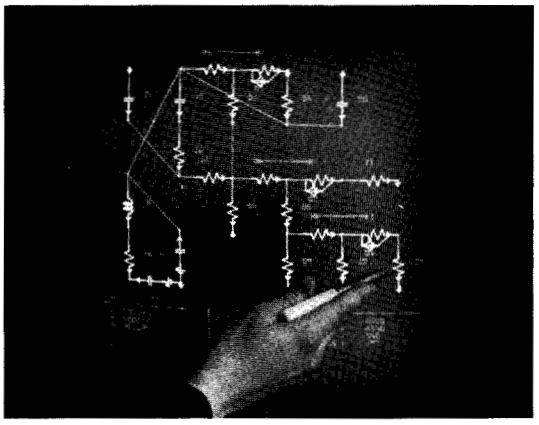
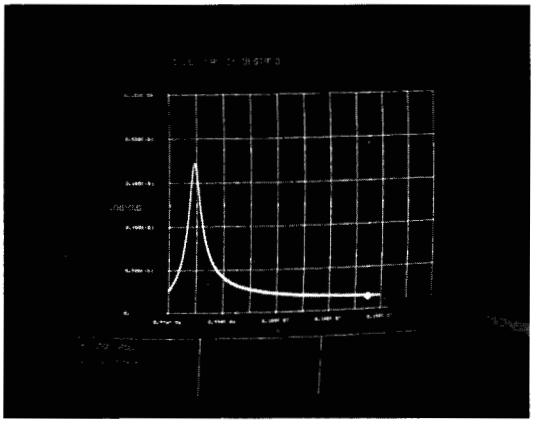


Figure 2 Typical output of a graphics ECAP circuit design



continuous systems modeling

Another simulation program is the Continuous Systems Modeling Program (CSMP).[4,5] Several persons have adapted this program for use in a graphics environment. Just as one would use an analog computer, the program is used as a generalized differential-equation solver, the problem description being in the form of a block diagram familiar to users of analog computers. The use of the Continuous Systems Modeling Program is characterized by many iterations. First, one attempts to model a known phenomena, the output of which is then compared with known data. Once the model has a degree of sophistication necessary to adequately match the data, the model can be manipulated in many ways to try out new ideas for designs for which data is not available. This program has been successfully applied in such areas as control systems analysis, kinematics, process dynamics, thermodynamics, and biomedical studies. As with ECAP, this form of analysis saves considerable time (possibly in the order of weeks or months for a complete study) and materials otherwise used to physically model the system under study. In addition, measurements that would be difficult or impossible in a real model are readily accessible. The available graphics versions of the program have output in the form of curves, the growth of which can be observed during the analysis. Some versions also allow the plotting of parameters after the analysis. Figure 3 shows a graphics CSMP analog input model. An IBM 2250 display of a typical CSMP model analysis is shown in Figure 4.

circuit layout

One graphics program with great potential, now in its embryonic stage, is in the area of circuit layout. Manual design of circuit boards and integrated circuits requires that much time be spent in placing and connecting elements for a given circuit on paper without violating a set of constraints. A computer can be programmed easily to check for design violations and to check that the constraints have been satisfied; but for the most part, programs in

Figure 3 A graphics CSMP analog input model

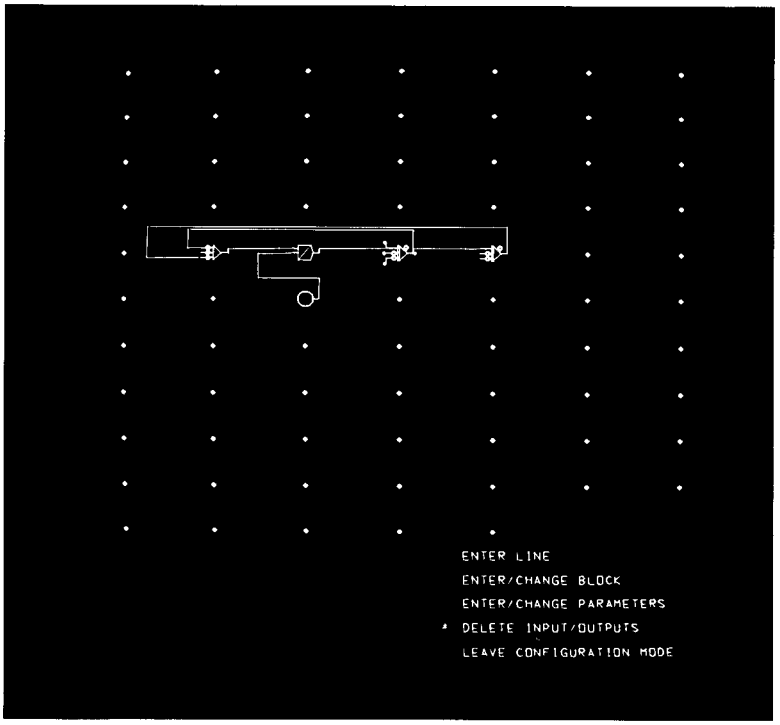
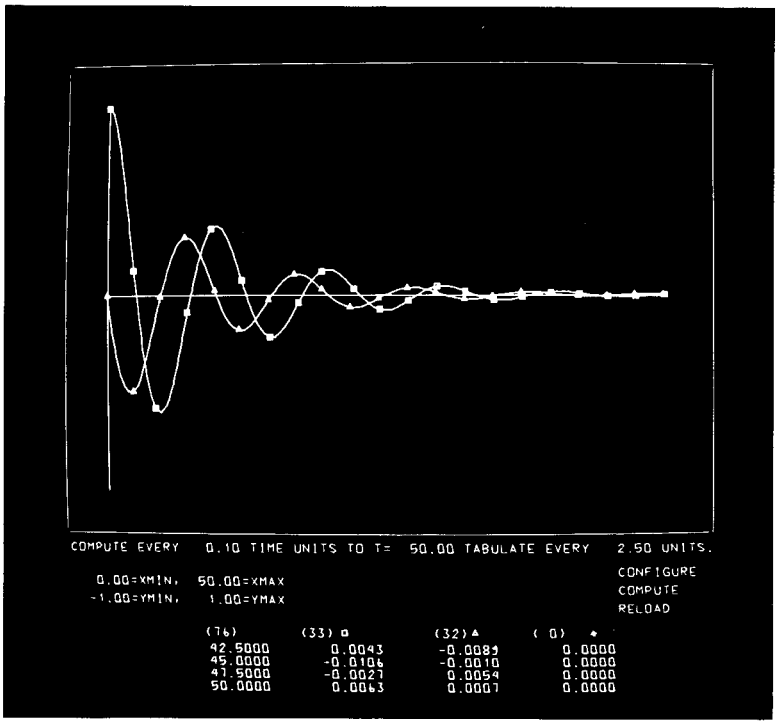


Figure 4 Typical graphics CSMP analysis output



existence today are only partially successful in doing the module placement and the connector routing. For a complete solution to the problem, the difficult portion must still be completed by hand on paper. To be checked for design violations, the results of this manual finishing work must be reentered into the computer. It is considerably faster if the designer can do his work on the display console while being dynamically provided with diagnostic messages. At any point during the design, the efficiency of the design can be calculated and measured—for example—as the percentage of the surface covered by modules and connectors. The results of the design are usually fed to a drafting machine to generate the artwork for etching of the product. The use of interactive graphics for this type of electronics design may exceed the use of a graphics version of network simulators if methods of circuit synthesis become completely automated.

3-D design and analysis

One of today's largest experimental graphics endeavors in data processing is that of three-dimensional design and analysis. The models involved are frequently quite large, and construction and analysis are often done piece-wise rather than on the entire model. Because the many designers involved must be allowed to work on their individual subsets of the total product in an independent, yet coordinated manner, communication of the most current information through data base maintenance is essential. This is in contrast to most other graphics applications now in productive use, in which only a handful of people are responsible for the entire design or analysis, and dependency on a data base is almost nil.

Areas in which three-dimensional graphics is appearing include airframe and auto-body design, three-dimensional pipe and wire routing, and chemical structure research, to mention a few. In a pipe-routing program under development,[6] the designer at the graphics console can specify the geometric and parametric quantities of a piping structure, perform stress analyses on the system, observe the results, and modify the original design for further analysis. This implementation is especially interesting for the three-dimensional cues it provides. To increase the viewer's perception of image depth, stereoscopic images are displayed. Two images, each corresponding to the view of the piping structure as seen by one of the viewer's eyes, are juxtaposed on the screen. Viewed through a stereoscope, the two images are blended into one producing an excellent rendering of the third dimension. In this way, a much more complex system may be displayed without confusing the user. This application is one of several first steps being taken in three-dimensional computer graphics. Methods are still being sought to streamline storage, manipulation, and access associated with three-dimensional graphics applications and their highly complex central data bases.

engineering drafting

The problem of engineering drawing or design drafting differs from most graphics applications in that the economic justification is more closely tied to the mechanics of a person working at the display console. We are not comparing two methods of using the

computer but rather two methods of drawing a picture. The contest is not yet settled between the method of drawing pictures on the display console by using a language and that of using a light pen or similar device. There are those who feel that the geometry of a part should be described using a descriptive language, such as APT (Automatically Programmed Tools), and then to use the display console for verification of the result; it is fully expected that users of this method will use the display in a dynamic mode for correcting the image. Others feel that a rough drawing must be made prior to the use of the language anyway; therefore, the display console should be used to make the drawing, bypassing the language step altogether. In either case, one by-product of the application is output for use in operating numerically controlled machine tools. Such an argument can be applied to any area of discipline where currently a language is used to describe a diagram of some sort.

programming terminal

The experienced programmer using the display console as a programming terminal often gains impressive advantages in time, efficiency, and cost because he is able to perform a part of his job on-line. This popular activity of on-line programming and debugging evolved from the simple, often unjustified use of the computer console lights and switches to today's program-aided, terminal-oriented methods and has found its way to the graphics device. And with good reason: speed is a necessary ingredient in successful on-line programming. Conventional devices are limited in this property; only a display console can handle repeated requests for large amounts of data. Thus, the graphics device is a natural choice for a programmer's terminal in which paging through source listings is as common as the dumping of data, object code, and main-storage images. Some programming packages now in use interface the programmer through the IBM 2250 display console to the normal programs used in a batch mode. One of these packages[7] provides a QUIKTRAN-like interaction at the 2250, allowing the programmer to write his program in FORTRAN language. During execution of this program, the programmer retains control and observes the progress while receiving its generated output. At any time, he can debug the program, edit his source statements, and prepare for a rerun. Another programming package, aimed at the assembler language programmer, responds to program interrupts. The error can be located and either corrected or circumvented on-line. In this way, many programming bugs may be remedied during a single run, resulting in fewer reassemblies and reloads as well as marked improvements in program checkout time.

When combining modules or sections of code to form a single program, some type of linking facilities is needed. On-line job control may be carried out through a graphic job processor package[8] which allows the programmer to specify such jobs as the running of the linkage editor at the 2250. The graphic job processor can also be used to execute the results of the link step. At his desk, a programmer is able to work on several programs concurrently, and it may take him several weeks of elapsed time to create any given

working program. However, the time he actually works on one program from his first assembly or compilation to completion is usually a very small percentage of this elapsed time. The use of a graphic programming terminal allows the programmer to execute several man and machine phases of programming end-to-end, speeding up the generation of programs of high priority.

concluding remarks

It should be noted that the use of graphics devices is not at all restricted to technical applications. Information retrieval and file maintenance have also found a tool in graphics. Production line scheduling has been done with graphics. Discrete simulation (such as the General Purpose Systems Simulator) has found its way to graphics, and project control (such as PERT diagrams) has been used experimentally with graphics. As commercial data bases come into practical existence, and as personnel in planning functions are forced to interact with these data bases, graphics devices of various degrees of sophistication can be expected to satisfy their requirements. Thus—although the majority of display console applications are still in an experimental stage—the present trend indicates more productive use of interactive graphics in data processing over the next few years.

CITED REFERENCES AND FOOTNOTES

1. Treated by A. Appel, T. P. Dankowski, and R. L. Dougherty in another paper of this issue.
2. Approaches in this direction are discussed by F. C. Chen and R. L. Dougherty as well as by H. B. Baskin and S. P. Morse in two other papers of this issue.
3. The graphics version of the Electronic Circuit Analysis Program (ECAP) is available from the IBM Program Information Department in Hawthorne, New York, as a Type-III program under number 360D-16.4.002.
4. The SYSTEM/360 CSMP and the IBM 1130 CSMP are discussed by R. D. Brennan and M. Y. Silberberg, "Two continuous systems modeling programs," *IBM Systems Journal* 6, No. 4, 242–266 (1967).
5. For an IBM 1130 CSMP application, see the paper by H. B. Baskin and S. P. Morse in this issue.
6. C. M. Strauss and S. Poley, "3DPDP: a three-dimensional piping design program," to appear in *Proceedings of IFIP Congress* in Edinburgh (1968).
7. This project is discussed by F. Gagliano, H. W. Thombs, and R. E. Cornish in this issue.
8. Discussed by S. Brown in another paper of this issue.