ESD RECORD COPY
RETURN TO
SCIENTIFIC & TECHNICAL INFORMATION DIVISION
(ESTI), BUILDING 1211

ESTI FILE COPY

ESD-TR-69-121

ESD ACCESSION LIST
ESTI Call No. 66424
Copy No. 1 of 2 cys.

FINAL REPORT SD-265
(PROJECT TACT)

1 October 1968

DIRECTORATE OF PLANNING AND TECHNOLOGY
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
L. G. Hanscom Field, Bedford, Massachusetts

P/C

Sponsored by: Advanced Research Projects Agency
Washington, D. C.

ARPA Order No. 952

This document has been approved for public release and sale; its distribution is unlimited.

(Prepared under Contract No. F19628-68-C-0300 by Harvard University, Cambridge, Massachusetts)

AD0692782

LEGAL NOTICE

When U. S. Government drawings, specifications or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

OTHER NOTICES

Do not return this copy. Retain or destroy.

ESD-TR-69-121

FINAL REPORT SD-265
(PROJECT TACT)

1 October 1968

DIRECTORATE OF PLANNING AND TECHNOLOGY
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
L. G. Hanscom Field, Bedford, Massachusetts

Sponsored by: Advanced Research Projects Agency
Washington, D. C.

ARPA Order No. 952

This document has been approved for public release and sale; its distribution is unlimited.

(Prepared under Contract No. F19628-68-C-0300 by Harvard University, Cambridge, Massachusetts)

FOREWORD

This report describes the work completed under Contract SD-265 from July 1964 through March 1968. This project was concerned with development of information processing techniques which could aid cognitive processes.

The major portion of the work conducted under this contract was directed towards design and development of an advanced computer-aided teaching system named the BRAIN. This report is devoted primarily to a description of the structure and function of the BRAIN. Additional work on the BRAIN is being conducted under ARPA sponsorship with contract F19628-68-C-0300.

The second part of contract SD-265, which started in the last few months of the contract, was concerned with computer graphics and networking. This effort is only briefly touched upon in this report. ARPA is providing continued support of that work under contract F19628-68-C-0379.

Professor Anthony G. Oettinger of Harvard University has been the principal investigator on this project. Dr. Lawrence Roberts was the ARPA director. Mr. James S. Duva, Mr. Keith Handsaker and Lt John P. McLean have provided technical guidance.

This technical report has been reviewed and is approved.

Sylvia R. Mayer
SYLVIA R. MAYER
ESD/ARPA Agent

William F. Heisler
WILLIAM F. HEISLER, Colonel, USAF
Chief, Command Systems Division
Directorate of Planning and Technology

Robert W. Taylor
Robert W. Taylor
Director for
Information Processing Techniques

ABSTRACT

The project objective has been to determine what creative thought processes can best take advantage of new technology in computer hardware and software. The plan has been to acquire or develop on-line computer systems of significant mathematical power, and to explore their use in vivo in teaching and research situations. The main product of project research is THE BRAIN (The Harvard Experimental Basic Reckoning And Instructional Network), an interactive computing system which operates on a standard IBM 360 Model 50 under the standard IBM operating system OS. This working system is easy and flexible to use in mathematical and engineering investigations as well as in teaching, and has been engineered for straightforward exportation from Harvard.

TABLE OF CONTENTS

	Page
I. Highlights of THE BRAIN Design	1
1.1 Introduction	1
1.2 Inside-out and Outside-in Facilities	2
1.3 Naming Facilities	5
1.4 Random Variables	10
1.5 Composite Entities	10
1.6 Subroutines, Functions and Branching	11
1.7 Display Formal Control	12
1.8 Aids to Memory	13
II. Exploratory Applications	14
Appendix I - Contributing Staff	27
Appendix II - Publications	28
Appendix III - Project Reports	30
Appendix IV - Project History	31

Section I

Highlights of THE BRAIN Design

1.1 Introduction

In keeping with the project's major aim of determining how THE BRAIN, together with other technological devices, may contribute most effectively to creative teaching processes and to scientific and engineering investigations, our design objective for THE BRAIN was to combine a balanced selection of desirable features into a coherent system with careful attention to human engineering and to the exportability of a system easy to use by people interested in computers only as tools for their own work.

Detailed descriptions of THE BRAIN from the user's point of view are given in our Primer and User's Reference Manual (see Appendix 3) which are available on demand. This report highlights those design features of the system which enhance its transparency to the user. These features are intended to reduce as far as possible the user's conscious awareness of the interposition of a tool between thought and action. Our point of departure was the User's Manual for the Culler-Fried On-Line Computer[1], which was prepared with our collaboration at that stage in our project's history (see Appendix 4) when we were operating terminals connected to Glen Culler's installation in Santa Barbara through Western Union's dial-up microwave network. Continuing cooperation has led to some measure of feedback as reflected, for example, in Fried's "On the User's Point of View[2]". But, in the main, the features of THE BRAIN -- like its actual implementation as a task under OS 360 -- have diverged significantly from those of its ancestors.

Some divergencies stem from the recognition of certain irreducible differences of style among users, differences which must therefore be accommodated and not ignored by forcing every user into a single mold. Even a single user will, according to our experience, change his style depending on the nature of the problem of current concern to him and depending even on the stage of development of his work on a problem.

---

[1] Culler, G. "User's Manual for an On-Line System" in On-line Computing, W. J. Karplus, ed. McGraw-Hill 1967, pp. 303-324.

[2] Fried, B. D. "On the User's Point of View" in Interactive Systems for Experimental Applied Mathematics, M. Klerer and J. Reinfelds, eds. Academic Press 1968, pp. 11-21.

1.2 Inside-out and Outside-in facilities

The way in which a problem may affect a user's style is illustrated by a distinction between inside-out and outside-in problems which we have found very useful and which has therefore influenced the design of THE BRAIN.

The inside-out/outside-in terminology is suggested by two different ways of viewing a mathematical formula. In one case, which might also be described as synthetic, the fragments of a formula (corresponding to terms enclosed in inner parentheses) present themselves as individual building blocks to be developed each in turn and combined step by step into a structure of increasing complexity. In the other case, which might be viewed as analytic, a formula presents itself as a whole to be used directly or analyzed into components. Attention is focussed on the explicitly present or implicit set of outermost parentheses.

Typically, the inside-out view is germane to an exploratory situation in which a new problem is being explored. Formulas are not available but are to be constructed; it may be desirable to understand the behavior of each component as a prelude to grasping how it combines with other components. The outside-in situation, on the other hand, corresponds to remembering or finding in a handbook a whole formula to be evaluated without concern for its components. These two approaches are not altogether disjoint since, in the course of an exploratory investigation, one component may well present itself as a complete and well-known formula.

The Culler On-Line System was constructed exclusively from the inside-out point of view. THE BRAIN, which permits an inside-out approach similar to Culler's, also operates in an outside-in mode more reminiscent of such facilities as JOSS, BASIC, or the on-line versions of FORTRAN. Indeed, THE BRAIN permits easy passage from the one mode to the other according to the style of the user and the nature of his problem or subproblem.

The foregoing may be illustrated concretely with reference to the well-known Law of Cosines formula:

$$(1) \quad \alpha(t) = a(t)^2 + b(t)^2 - 2a(t)\, b(t) \cos C\,(t)\ .$$

While THE BRAIN can operate on scalars, it is a more interesting tool for the manipulation of functions. The elements of (1) have therefore been expressed as functions of a parameter t so that the behavior of α may be examined as a function of the values of t in some range.

The following expression illustrates the sequence of keys to be pushed in order to evaluate α. Each underlined expression stands for a single keypush:

(2) <u>LOAD</u> a <u>SQ</u> = t1 <u>LOAD</u> b <u>SQ</u> + t1 = t1 <u>LOAD</u> 2 · a · b

= t2 <u>LOAD</u> C <u>COS</u> · t2 <u>NEG</u> + t1 = α .

Each underlined expression stands for a single keypush. Thus <u>LOAD</u> places an item in a working register; <u>SQ</u> squares whatever is in the working register; <u>COS</u> replaces the contents of the working register by the cosine of the corresponding elements, while <u>NEG</u> replaces each element of the working register by its additive inverse.

Now consider (2) in the spirit of someone trying to remember or perhaps to construct (1). He loads a and squares it, pausing perhaps to give a display instruction to examine the behavior of a^2. Remembering now -- or perhaps just discovering -- the need for a term b^2, he must temporarily put away a^2 under the name t1, then create b^2, perhaps examining it on the display screen, and then combining it with the previous partial results stored under the name t1. Realizing once again that an additional term is necessary, he puts the partial sum away under the temporary label t1. Next, the coefficient 2ab is formed but cannot be combined with the term cosS since the latter has not yet been formed. The coefficient 2ab is therefore stored under the temporary label t2 while the cosine term is formed, then multiplied by t2, the whole combination negated and finally added to the previously obtained sum of the two squares. Once again, partial results could be examined at intermediate points.

Our experience, like Culler's, has demonstrated to designers and users alike that where it is desirable to construct an expression from the inside-out or inductively, it is most useful to have transparent facilities for building components up piece by piece, examining each as required, and finally combining them into some desired result possibly after several alterations of unsatisfactory partial results. The inside-out mode provides such a facility.

On the other hand, the inside-out procedure is most unattractive, exasperating and opaque to someone who has (1) in front of him and wishes merely to evaluate it or to combine it <u>in toto</u> with some other expression. Under these circumstances the sequence of actions described in (2) seems exasperatingly time consuming and burdened with such logically irrelevant operations as the creation of temporary labels (equivalent to transfers to temporary storage locations). In fact, the process of converting (1) to (2) is essentially what the first stage of a conventional compiler does. The user, thus constrained to play compiler, is made quite uncomfortable. If he is unfamiliar with the compilation process, he is unable to pinpoint the source of his discomfort but merely rejects the system.

These problems become even more acute if the system is not operated in the manual mode where each keypush leads to immediate action and partial results may therefore be displayed and observed but

operated rather in the mode where a composite operator (or subroutine) is being constructed for future execution out of elementary operators or other available composite operators. In such cases an experienced user might well wish to design the composite operator as compactly as possible, perhaps in the following manner:

(3) LOAD a SQ = t1 LOAD b SQ + t1

= t1 LOAD C COS · 2 · a · b NEG + t1 = α .

The use of the temporary t2 in (2) has been eliminated by interchanging the order of computing the cosine term and its coefficients. The mental gymnastics necessary to make this interchange distract the user from his mathematical purpose and, at this stage of development of the computer art, they are an unnecessary nuisance since the expression to be evaluated is already known.

Effects equivalent to those of (2) or (3) may be obtained on THE BRAIN by pushing the keys described by the following expression:

(4) (α = a SQ + b SQ - 2 a b COS C) .

Except for the enclosing parentheses, this expression is almost a direct transcription of (1), including the use of implicit multiplication operators and the assumption of conventional precedence relations. A parser invisible to the user performs the necessary analysis and internal code generation. THE BRAIN is therefore as convenient to use for an outside-in problem as for an inside-out problem.

Experience having shown that users cannot, in general, predict at which stage of what problem they will be more comfortable with which of these two modes, we have provided for an easy alternation between the two. Suppose, for example, that the user is engaged in evaluating β while having in mind a sequence somewhat like:

(5) LOAD x + z = β .

He might have pushed LOAD and x but then recalled that z had not yet been computed according to some simple formula either in his mind or on paper before him. By enclosing this formula in parentheses, he can substitute it for z right on the spot and push the following keys:

(6) LOAD x + (2 - Y SQ) = β .

In general an outside-in expression in parentheses is accepted by THE BRAIN wherever a single data name may appear. Moreover, should the user have some interest in the partial expression, perhaps to save it for later calculations or perhaps to display it in isolation,

he has the option of combining the evaluation with an assignment statement as illustrated in:

(7) LOAD X + (ꞩ = 2 - YSQ) = β .

Once the program represented by (7) has been executed, both β and ꞩ have appropriate values assigned to them and available for further manipulation. The conventional display operator for functions of a real variable "DISPLAY Y X" which displays Y as a function of X may be invoked whenever Y and X have previously been computed. If, however, each of these arguments depends in turn on others, the computation can be made implicit in the display operator according to the general convention that a parenthesized outside-in expression may be substituted for any argument. Thus, if t ranges between 0 and 2π, the expression

(8) DISPLAY (COS t) (SIN t)

will lead directly to the display of the unit circle.

1.3 Naming facilities

Another aspect of THE BRAIN'S design strongly influenced by our experience with varying user styles and the varying demands of different problems at different stages of their solution is the matter of naming. There is an attractive sense of economy in Culler's concept of "one-name/one-key". The difficulty, of course, is that the number of available names is thereby limited to something less than the number of keys on the keyboard. The number of names may therefore be extended only by extending the keyboard, a process that quickly reaches the limit of practicability, or else by establishing a one-to-many correspondence between keys and names, a process requiring something like multiple case shifts to maintain the necessary one-to-one correspondence between key strokes and named entities. Our experience with Culler's system convinced us of both the great value of the one-name/one-key concept for certain users at certain times and of its discouragingly severe limitations also for some users at some times.

The TOCS system which we implemented on Project MAC's CTSS during an early stage of this project (see Appendix 4) reaffirmed the value of the more conventional multi-key naming systems. Under many circumstances, the need to strike several keys per name is offset by the enhanced mnemonic value of names so constructed and by the avoidance of frequent and confusing case shifts.

Accordingly, THE BRAIN permits operation in either a "single-button" mode or a "typewriter" mode. Single-button mode corresponds to Culler's naming system where each key stroke invokes some data entity (operator, operand, etc.) without spaces or other delimiters being required between successive names.

In typewriter mode, names may be constructed of eight characters or less, but, of course, each name must be terminated by a space or other delimiting character.

Passage from one mode to the other at the whim of user style or according to problem type has been facilitated by embedding the single-button mode within the typewriter mode in the following sense. Every defined single-button name is available also as a typewriter name when the same single-button push is followed by a delimiter character. Moreover, every standard operator assigned to a single button on the standard upper keyboard of THE BRAIN is given a canonical typewriter name. Thus, in typewriter mode, the sine operator may be executed either by pushing key #4 on the upper keyboard (Figure 1.1) or by typing "SIN" on the lower typewriter keyboard. These canonical names are printed on the standard overlay shown partially removed from the upper keyboard in Figure 1.2.

Additional naming flexibility is provided by the SYNONYM operator. Pushing the synonym key (key 46, Figure 1.1) or, alternatively, typing "SYNONYM" followed by the arguments "oldname" and "newname" establishes "newname" as a synonym of "oldname". Thus the user who inveterately prefers "sine" to "SIN" may synonym the former to the latter and, thereafter, type his customary four characters to execute the sine operator.

A standard display is associated with each key; these standard display "characters" are engraved on the keys of the lower keyboard or printed on the standard overlay for the upper keyboard. The display value of a key is not altered by the synonym operator since, at any given time, several synonyms may be associated with a particular key. The user does however have the option of constructing an arbitrary display value of his choice and associating it with a key by using the "BUILD" and "STORE" operators as described in the User's Manual. He may then also wish to change the value shown on the overlay. Thus the user is provided with a basic standard naming system which our experience has shown to be flexible enough to be serviceable, at least initially, for most users. As the user grows in experience and his programs in complexity, he may alter the naming scheme as informally or as formally as he wishes.

We envisage that creating synonyms and corresponding special overlays will prove particularly useful in certain teaching applications. For instance, in demanding a response from a student a composite operator (subroutine) within a program might display instructions to the student on the screen and conclude with the INPUT operator which restores the system to manual control. Following the execution of the manual operations required of him, the student is then required to activate the "RESUME" operator (key 58, Figure 1.1) which returns the system to program control. It seems easier to change the display

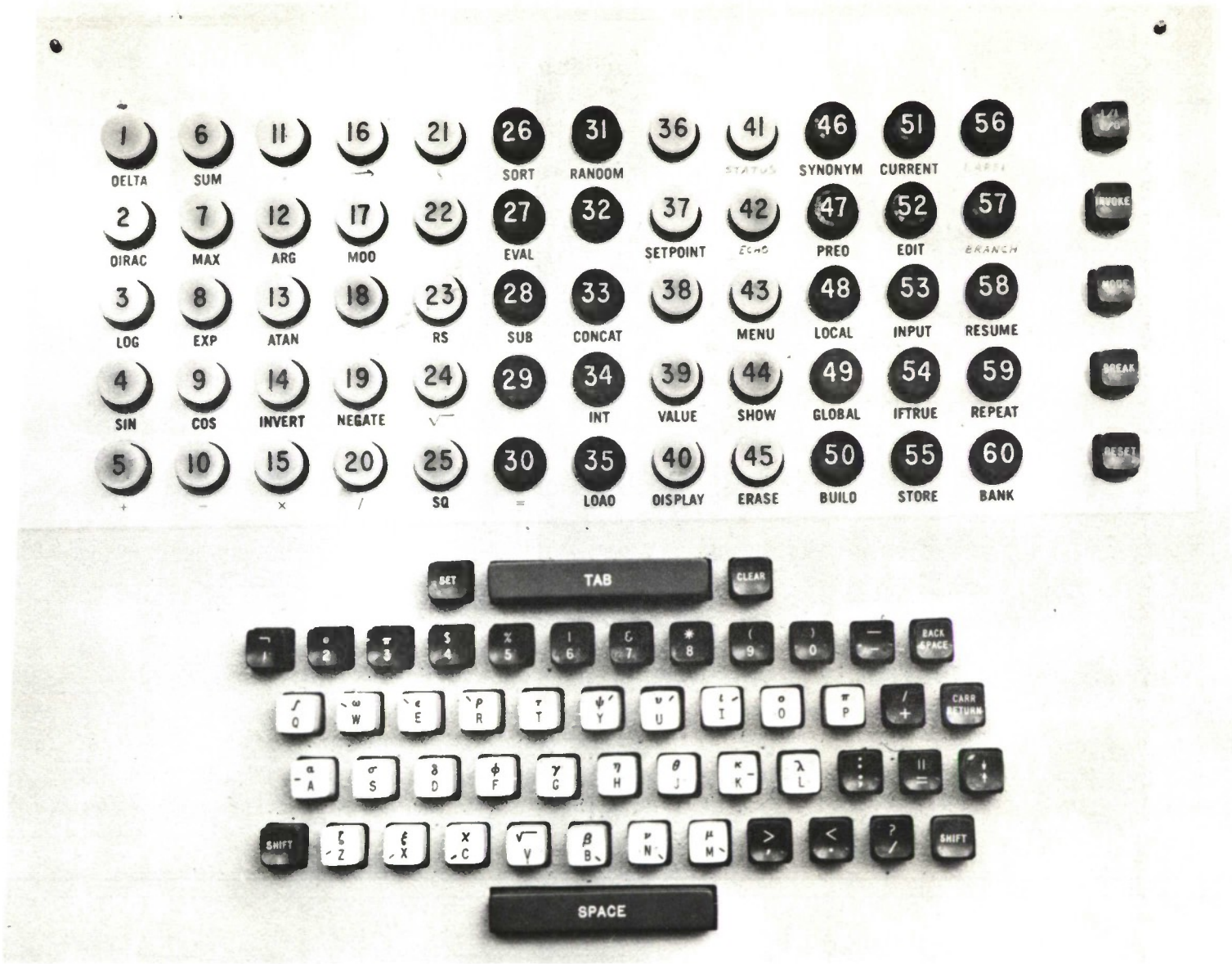


Figure 1.1

THE BRAIN Keyboard

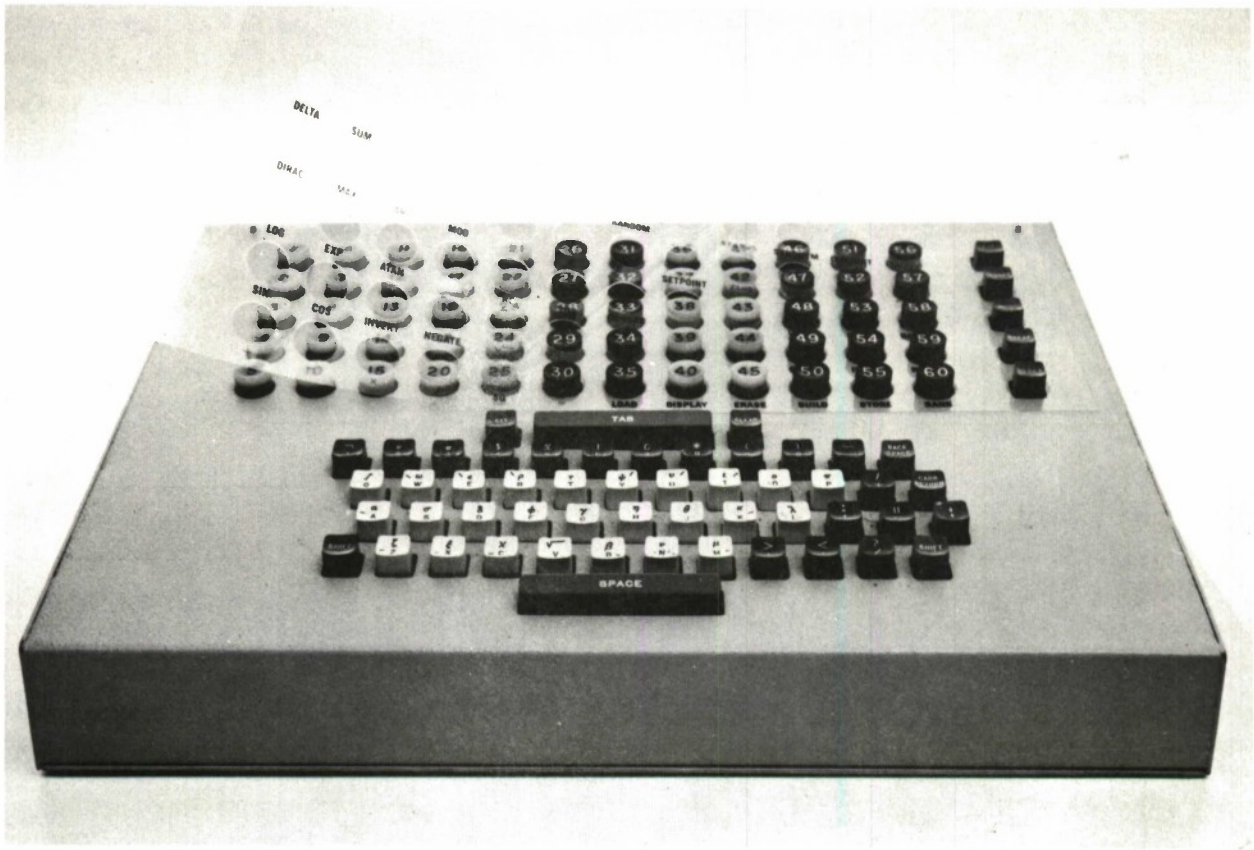


Figure 1.2

Upper Keyboard Overlay

value and the labeling of key 58 to something like "ANSWER" than to provide the student with a technical explanation of the uses of the "RESUME" operator. A routine named "XYINT" by its creator might be synonymed to something like "Case 2" for student use in a particular context without confusing either the creator or the student.

This flexibility of naming and name display dovetails with the ability to attach any standard or composite operator to any arbitrary single-button or typewriter name. Consequently, the user need not be limited by the configuration of the standard keyboard. Any of up to 60 operators may be attached to the keys on the upper keyboard in any desired array and provided with a corresponding set of display values on a suitably engraved overlay. Anytime an operator which has replaced one of those shown in Figure 1.1 is deleted, the standard operator shown in Figure 1.1 automatically becomes available once again.

Specially constructed keyboard configurations are themselves treated as data entities. They can be named, stored in a permanent file, and reactivated at will. Operators not available on a particular upper keyboard are accessible at all times through their typewriter names. The construction of a special keyboard is therefore a matter of economy and perspicuity, not of necessity. Typically one might strive to have those operators most frequently used directly available as keys on the upper keyboard with all others remaining available through their typewriter names. The standard systems operators "SAVE" and "RESTORE" which create and retrieve disc files are not represented on the keyboard of Figure 1.1. Any user, who has frequent occasion to use these operators can substitute them for any pair on the keyboard which he rarely uses.

These flexible naming and keyboard construction facilities of THE BRAIN have superseded the "level" conventions of TOCS and its ancestors. The user may also freely move back and forth between the real and the complex plane without the level shifts of earlier systems. Naturally this has entailed the labeling of data by types and some testing for data compatibility by arithmetic operators like "+" and others. The resulting slight cost in storage and in operating time seems well compensated for by the increase in transparency and ease of use.

There is no rescuing a user who uses the same name for two distinct operators or two distinct data arrays. It is, after all, trivial to invent a new name when in doubt. However, the user should not be penalized for using the same name for both an operator and a data array since, in fact, this practice is common in mathematics where a function (viewed as an operator) and its value (viewed as that element of the range of an operator which is obtained when the operator is applied to an element of its domain) are often named alike. In THE BRAIN a single name may therefore be applied simultaneously to a data array and to an operator. There is no difficulty, when analyzing

a string of commands of either the inside-out or the outside-in form, in distinguishing when an operator is required and when a data array. Disambiguation of this kind is simple in THE BRAIN's syntax. The user who chooses to call an operator "Y" and corresponding values "y" gains some clarity, but he would not be in trouble should he use either character for both entities.

Indeed a single name may be applied simultaneously to yet a third object chosen among the categories of keyboards, format specifications, or files. For example, a format specification describing how a data object is to be displayed on the screen may be given the same name as the data object itself and as the operator which creates that object.

1.4 Random Variables

With a little care in programming, THE BRAIN is made indifferent as to whether it is dealing with functions of deterministic or random variables or, independently, of real or complex variables. For instance, the very simple composite operator LOAD X SIN = Y will define Y as real or complex, deterministic or random depending on how X has been defined prior to the execution of this string of standard operators. The operator "INT A B C" creates a linear array of C components consisting of equally spaced values in the interval (A, B). If either A or B is complex, the resulting complex array defines a line joining A to B in the complex plane. The complex number x + i y is written as x, y. Thus the string

(9) INT 0, 0 0, 2π N EXP = Y DISPLAY Y

will compute and display N equally spaced points of the unit circle in the complex plane.

The operator "RANDOM A B C" creates C random numbers rectangularly distributed between A and B. In the complex plane these points are distributed within the square whose diagonal runs from A to B, since the values of the real and imaginary components are picked independently. When A and B are real, the RANDOM operator, together with a SORT operator which arranges the elements of a linear array in increasing order, provides a handy means for applying conventional mappings to obtain samples from any distribution previously defined by an appropriate cumulative distribution function.

1.5 Composite Entities

Composite operators, composite characters (as for a new display value) or display format control statements and also strings to be attached as narrative descriptions to operators, data arrays or format controls are all created through the use of the BUILD operator.

Executing BUILD shifts the system's operation from immediate interpretation and execution to a string construction mode. A built string may subsequently be attached to a name by using STORE followed by the desired name. Both BUILD and STORE may take an argument defining the kind of object that is being built. If no such argument is given, the system's default option assumes that a composite operator is intended. Conventional editing facilities are available to correct errors or otherwise to alter things constructed under control of the "BUILD" operator.

1.6 Subroutines, Functions and Branching

The problem of avoiding confusion between the names of working variables used only within the confines of a particular composite operator and nowhere else and the names of data elements used throughout a program is handled by the LOCAL operator which, when followed by a string of names enclosed within a pair of parentheses, identifies these names as active only within the confines of the composite operator within which the LOCAL operator has been executed. The beginner can easily pass arguments from one composite operator to another by using global variables, that is variables which have not been declared local and that therefore retain the same significance wherever they may be invoked.

A composite operator may, however, begin with the DUMMY operator whose arguments in turn define the arguments with which the newly defined composite operator is to be called. Names used as arguments within the DUMMY operator beginning a composite operator are automatically declared local to that composite operator.

Thus in the operator GEORGE constructed by the following sequence - all five variables A, B, C, D, and E are local to GEORGE:

(10) BUILD DUMMY (A B C) LOCAL (D E) LOAD . . .
STORE GEORGE.

To execute GEORGE, a sequence like GEORGE U V W must be used. When GEORGE begins execution, A, B, and C are declared local and taken as alternative names for whatever may be stored under the names U, V, and W respectively. Occurrences of A, B, and C anywhere in the GEORGE operator point to the data under U, V, and W respectively for the duration of GEORGE's execution. Should GEORGE store any data under A, B, or C, the same data become available under the corresponding names U, V, or W. Thus the arguments used to call a composite operator need not have been previously defined since there will be an inherent "functional return". However, when GEORGE is finished, the local D and E and whatever data may have been stored under those names disappear altogether. If global D or E exist, they remain completely unaffected by GEORGE and become available again precisely as they were before GEORGE began.

Conditional and unconditional branching facilities in THE BRAIN have been designed with particular attention to the need for easy n-way switching, as in choosing alternative paths in response to answers to a question. A full complement of Boolean operators is provided to enable easy construction of the logical decision sequences leading to a branch.

The operator LABEL A has no effect when encountered in the course of normal program execution; it is simply passed over. It serves as a place marker for use by the BRANCH (unconditional) and IFTRUE (conditional) branching operators. Branching from one place in a composite operator to another place within the same operator is done by using BRANCH A. The argument of BRANCH is compared from left to right with the argument of any labels within the composite operator string, and, if a match is found, execution resumes with the operator occurring immediately after the argument of LABEL. If no argument match is found, the composite program is ended and control returned to whatever invoked the composite operator in question. The operation of IFTRUE A B is similar to that of BRANCH B excepting that the branching will or will not take place according to the value of A. An n-way switch is therefore realized by constructing a composite operator with n labels, each followed by a composite operator defining the appropriate continuing action.

The condition A in IFTRUE A B may be the result of Boolean operations on numerical arrays which themselves may result from extensive computation. Thus A might be calculated so as to produce branching according to the degree of agreement between a proposed solution to a differential equation and a reference solution within an interval defined by a pair of functions bounding the reference solution from above and from below. Branching control in such a system is therefore not limited to multiple choice questions, but rather definable by the much wider range of decision procedures that can be constructed in terms of the arithmetic and Boolean operators available on THE BRAIN.

1.7 Display Format Control

The provision of a very simple format control language readily enables the user of THE BRAIN to control where on the screen and how displays are presented. Control options include the possibility of specifying a viewport, that is, a physical area on the physical screen. A window may be independently defined as a rectangle within the abstract xy plane. In both of these cases, as for all other options, simple default options are provided which enable the novice user to operate with the least amount of difficulty. Thus, in the standard system format, the viewport is the whole screen and the window is defined by the extreme values of the x and y coordinates of the curve being displayed.

Other options include page fixed vs. page variable displays. In page fixed formats, THE BRAIN automatically selects the window appropriate for the first display on the new "page" following an ERASE instruction; it then keeps that window for all subsequent displays until a new page is created by either an ERASE command or the invoking of a new format. Under the page variable option, on the other hand, a new window is selected for each curve displayed on a page. In the first case, therefore, all curves on a page are displayed to the same scale and may be visually compared; in the second case, maximum coverage is given to each curve, but visual comparisons of scale may be meaningless.

Other options provide for alternative scaling (linear or logarithmic) independently on both axes, for the location of the x and y axes with respect to the window when the window is automatically selected by the system, and for the scaling of x or y to enable either the preservation of true shape or an adjustment of the x and y scales so that the window completely fills the viewport so that maximum detail may be seen even though the shape of the curve may be distorted.

Figures 2.4 and 2.5 demonstrate some of the useful effects that can easily be achieved through the use of the format control language. A format control string is built, stored, and named like any other string. Normally the system is under control of a standard format which may be superseded by any other defined format through the use of the operator INVOKE followed by the name of the format. Return to the standard system format is effected by INVOKE (SF).

1.8 Aids to Memory

The operators MENU and SHOW are provided to enable the user to keep track of entities he has defined. MENU, followed by appropriate arguments, will cause a list of names to appear on the screen. Thus MENU makes it possible to display lists like that of all currently defined composite operators or the list of all names synonymed to the name A. The SHOW operator will, for example, cause a composite operator or its narrative description to be displayed. Other data entities or their narrative descriptions are similarly accessible through the SHOW operator. Thus, if GEORGE is a composite operator with a narrative description, SHOW GEORGE will cause the narrative description to be displayed. If there is no narrative description, the same command will cause the string of commands defining GEORGE to be displayed. Either type of display may be forced by commanding SHOW (DO) GEORGE or SHOW (0) GEORGE respectively. The SHOW operator is also used to convert the system into a typewriter. SHOW followed by any string in quotes will cause the quoted string to appear on the screen without any other effects. Instructions to users may be transmitted through operators like SHOW "Compute $x^2 + y^2 = R$, then push ANSWER".

Section II

Exploratory Applications

By the end of the contract period, THE BRAIN was working well enough to lend itslef to experimental use, but not yet to full-scale routine operation. The following excerpt from Section 5.5 of the book, Run, Computer, Run: The Mythology of Educational Technology, by A. G. Oettinger with the collaboration of Sema Marks, to be published by Harvard University Press in May 1969, briefly describes some educational experiments. THE BRAIN's promise is contrasted with what has been accomplished with more conventional devices applied to more conventional aims. Future efforts will be bent toward more extensive and rigorous validation of the promise, in the expectation that it can be realized with reasonable operating costs.

o o o o o o o o o o

Were a computer's educational promise limited to page-turning, grand clerking, or gimmicky imitation of Miss Dove, fiscal misgivings might be well-founded at any price. Gargantuan clerical prowess can solve many vexing workaday problems, but it scarcely excites the imagination. However, more subtle qualities make computers capable of profoundly affecting science and, as we shall see, education by stretching human reason and intuition, much as telescopes or microscopes extend human vision. I suspect that the ultimate effects of this stretching will be as far-reaching as the effects of the invention of writing. In their scientific applications computers have been cast in two quite distinct but complementary roles: as instruments and as actors.

The computer's role as an instrument is by far the more clear-cut and firmly established of the two. The advance of science has been marked by a progressive and rapidly accelerating separation of observable phenomena from both common sensory experience and theoretically supported intuition. Anyone can make at least a qualitative comparison of the forces required to break a matchstick and a steel bar. Comparing the force needed to ionize a hydrogen atom with the force that binds the hydrogen nucleus together is much more indirect, because the chain from phenomenon to observation to interpretation is much longer. It is by restoring the immediacy of sensory experience and by sharpening intuition that computers are reshaping experimental analysis.

It is in their other role, however, as active participants in the development of scientific theories, as actors, that computers promise

to have their most profound impact on science. A physical theory expressed in the static language of mathematics often becomes dynamic when it is rewritten as a computer program; one can explore its inner structure, confront it with experimental data and interpret its implications much more easily than when it is in static form. In disciplines where mathematics is not the prevailing mode of expression the language of computer programs serves increasingly as the language of science.

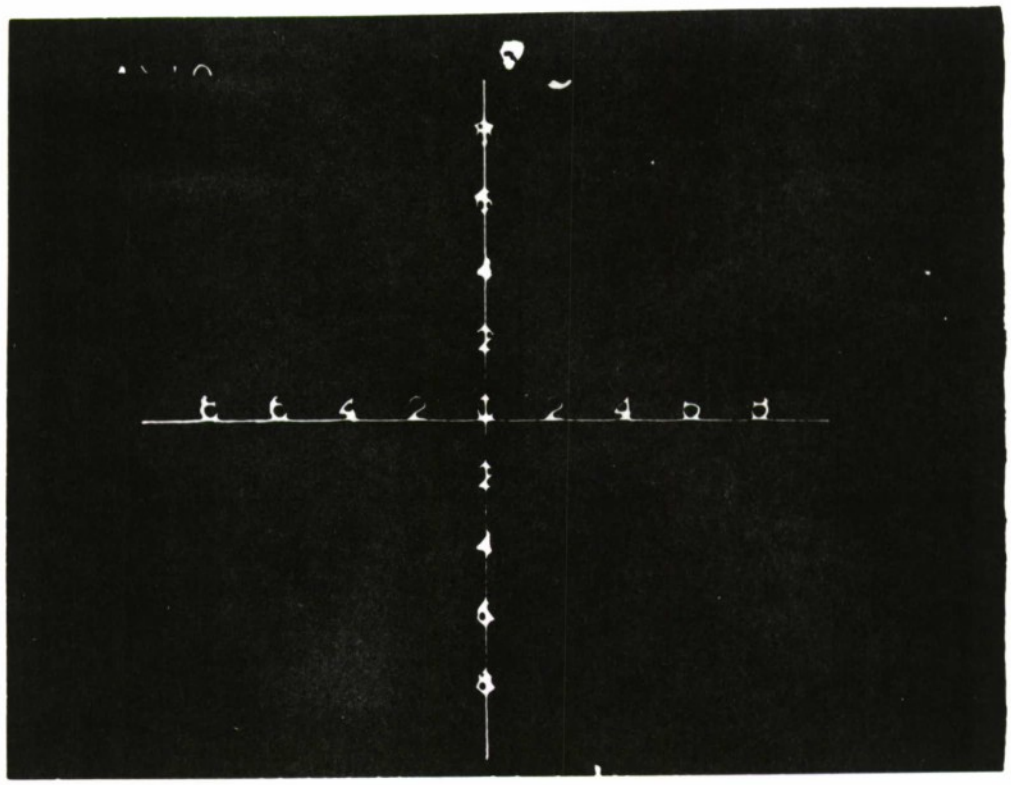
Computers used in this way, far from reducing the scientist to a passive bystander, reinforce the need for the creative human element in experimental science, if only because witless calculation is likely to be so voluminous as to be beyond the power of even the fastest computer. Human judgment and intuition must be injected at every stage to guide the computer in its search for a solution.

I have introduced the paradigm of the computer as instrument and as actor in terms of scientific research, where it has begun to prove its worth. It seems to apply to instruction and learning as well. The examples I shall use to illustrate this are drawn from my own recent experience with THE BRAIN. (Why let the devil have all the good tunes? "THE BRAIN" is an acronym for The Harvard Experimental Basic Reckoning And Instructional Network.)

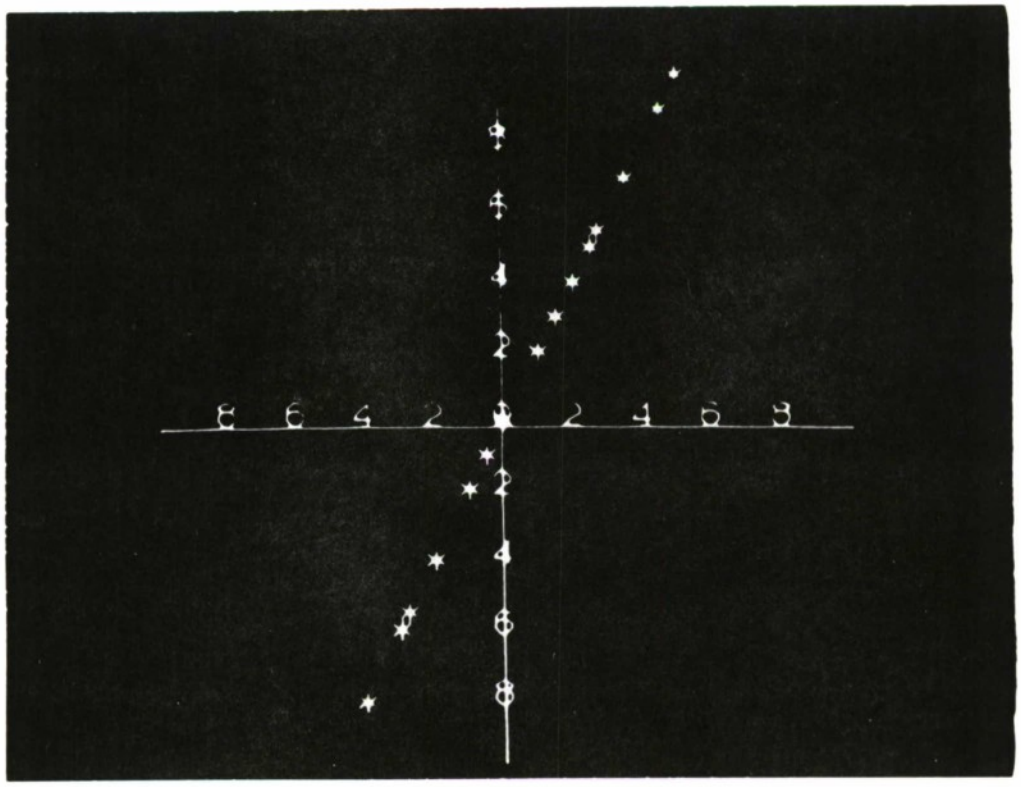
I chose mathematics as the experimental vehicle to permit the quickest possible passage to the desired study of effects. The coherent architecture of modern mathematics minimizes uncertainties about what is being taught or learned, albeit far from eliminating them. Computers being just that, the technical problems of fitting the tool to the task have proved bearable, though thorny and time-consuming. Finally, applied mathematics being a tool of many other disciplines, it opens a window toward wider vistas.

Let us look through a computer at some mathematical phenomena. To see the promised effects clearly, we'll assume transparency, as through clean, well-fitted glasses, though in truth our present lenses distort, their frames chafe, and their price is high.

Let us assume the student knows that the equation $y = 2x$ describes number pairs (x, y) which, like $(1.5, 3)$, satisfy the equation. He knows that each (x, y) also defines a point in the plane described by the axes of Figure 2.1a. What pattern -- he might then ask himself -- is formed by points which all satisfy the equation? Teacher or student might plot a few, as in Figure 2.1b. "As we began to plot points from the truth set for $y = 2x$," the experimental protocol reports, "the kids had no idea they would all lie on a straight line. Gradually they saw this pattern emerge and it looked very much like [Figure 2.1b]. 'Hey, that's neat!' one of the girls said, and someone else said 'Look, they're all in a straight line.'" And so they crossed Descartes' bridge from algebra to geometry.



(a)



(b)

FIGURE 2.1

Crossing The Bridge Between Algebra and Geometry

Overkill? Perhaps. No one could advocate using so large a tool solely for so small a problem. Still, the glint of promise is there. The same lesson was "taught to the other half of the class the following day using the traditional methods of pencil, ruler and graph paper, and was not nearly as successful.... The teacher spent most of the lesson having to correct mistakes the kids made on their own papers in plotting points from the truth table of $y = 2x$ (Figure 2.2 shows a paper retrieved from one of the students in the class). Many of the kids were very slow to see the straight line pattern of all these points since, at this stage, it was impossible for them to distinguish incorrect reasoning or calculation from errors in graphing."

The bridge crossed, the instrument grows more powerful. Approximating the area under a curve by a sequence of rectangles is an ancient procedure and, in the limit, defines the definite integral. Figure 2.3 shows how 6, 12 and 18 rectangles successively yield 6.1, 5.7, and 5.5 as approximations to $5.1 = 2 + \pi$, the value of

$$\int_0^{\pi} (\sin x + 1)\, dx.$$

The printed form of Figure 2.3, looking like similar textbook figures, cannot portray the excitement of observing convergence step-by-step, developing a feeling for rate of convergence by handily exploring the behavior of inscribed rectangles, rectangles of varying width, with the given curve or with others, ad lib. A formal proof of convergence may then follow, supported by a deep intuitive grasp.

Our instrument's field of view is wide and variable. Through it we can see, as in Figure 2.4, a spiral in space and its projections as on walls meeting the floor at a corner. Or we can turn it toward the complex plane and explore, as a fledgling airplane designer might, the properties of the Joukowski transformations that change circles, like Z in Figure 2.5a into wing shapes like those of Figures 2.5b and 2.5c. The role of P, as it enters the definition of Z in Figure 2.5a is obscure to all but experts. But look how clearly anyone can see in Figures 2.5b and c how the wing shape changes as the point defined by P moves up or across.

What can it mean to anyone but an expert to say that circle Z in Figure 2.5a is changed into wing-shape W by $W = Z + 1/Z$? How immediate, how palpable, how intuitive this remote abstraction becomes when, in an instant, we turn our instrument and see, as in Figure 2.5d, the strings that tie points of Z to their images in W and feel how the plane has been pulled and stretched.

To explain a current research project to his students, my colleague, William Bossert, has used THE BRAIN to act out the behavior of vesicles, microscopic spheres of intercellular fluid held

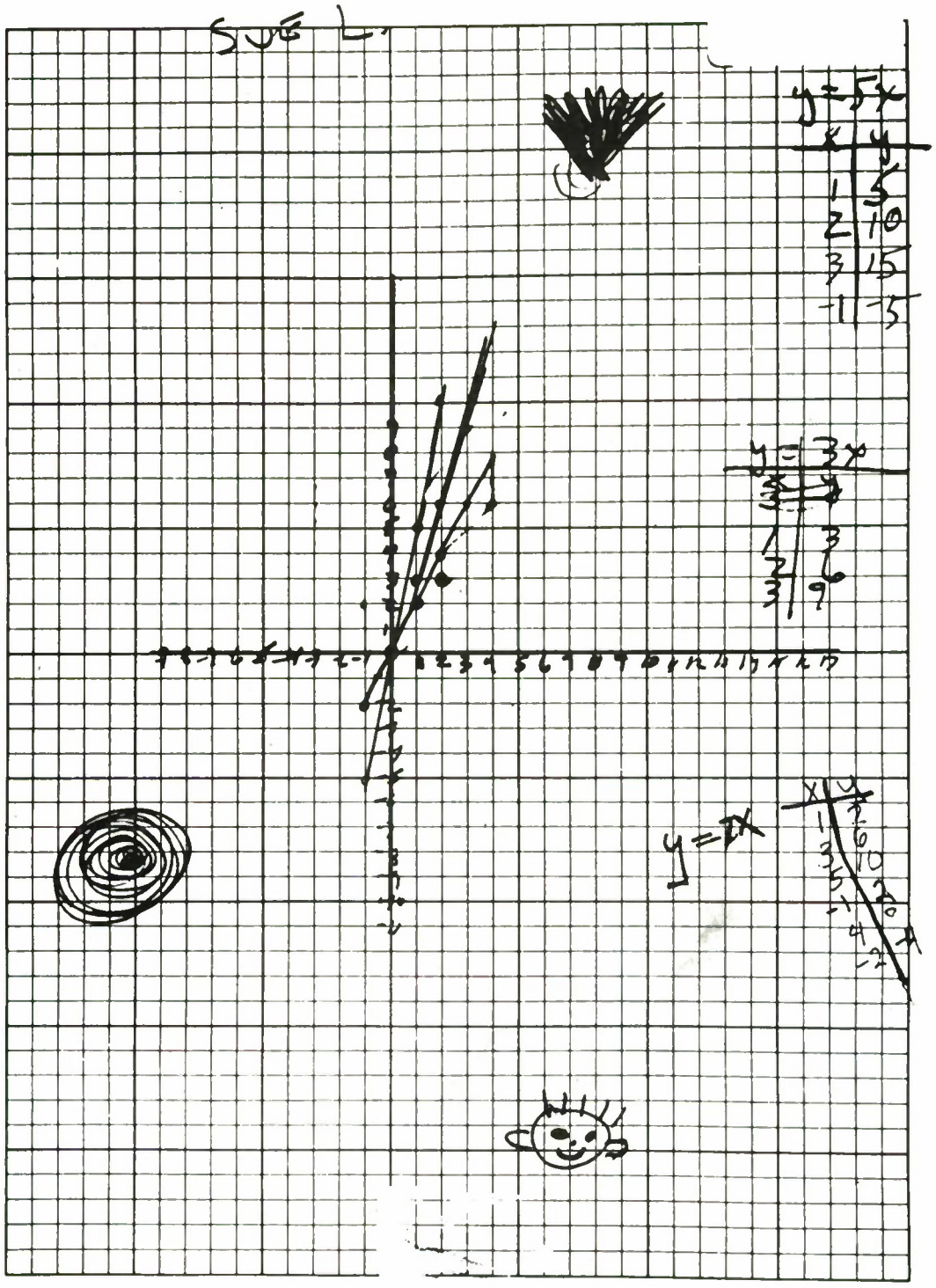


FIGURE 2.2

A View of the Bridge
(without glasses)

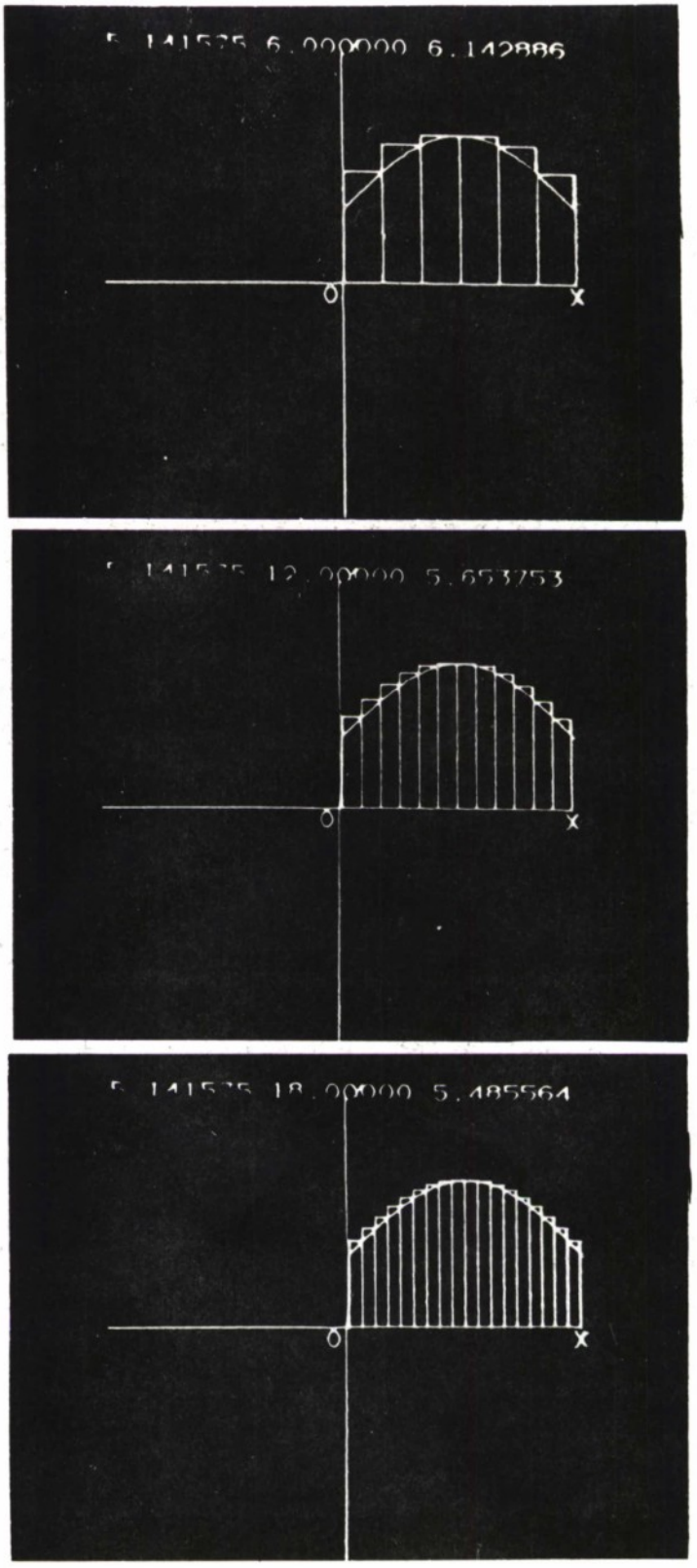


FIGURE 2.3

The Dynamics of Convergence to the Definite Integral

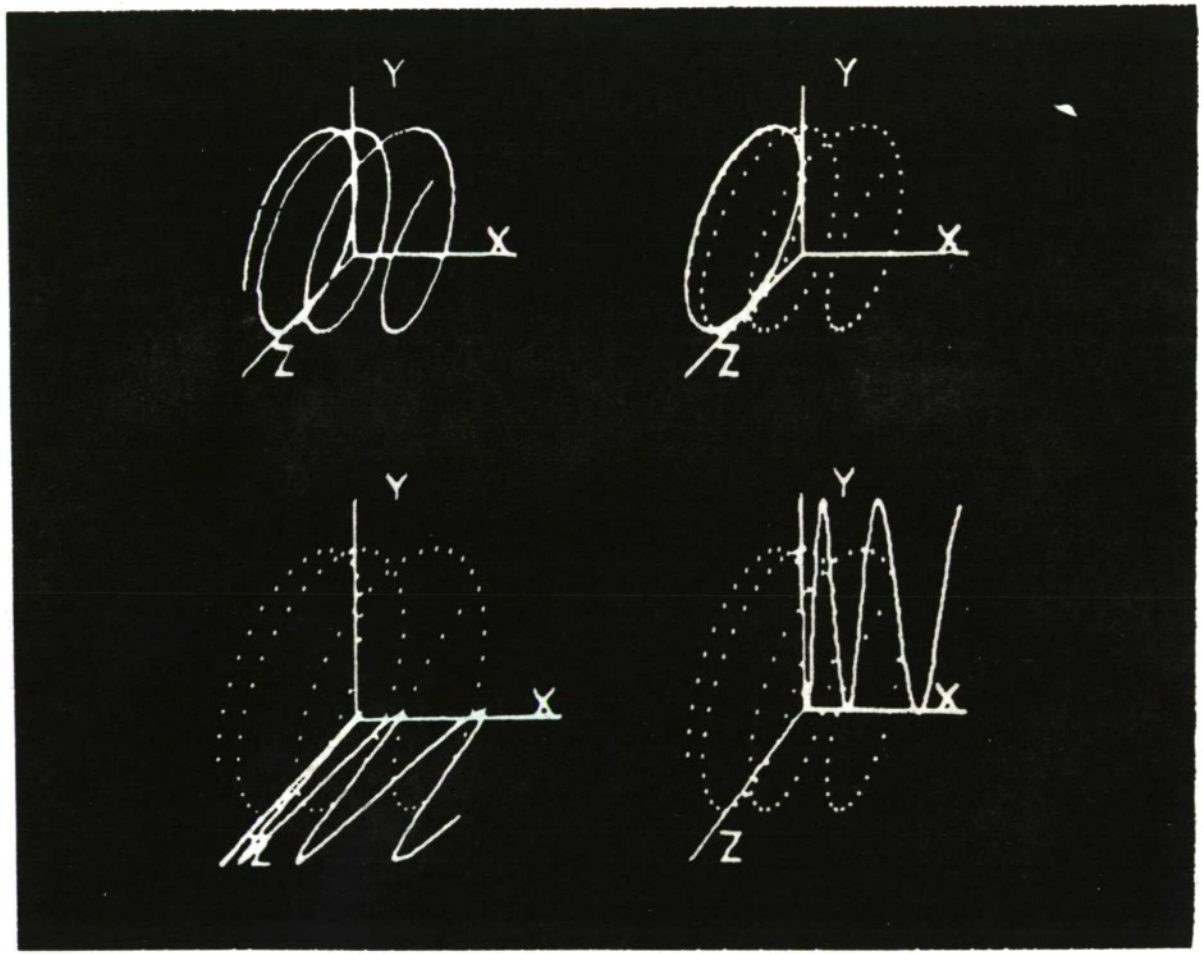
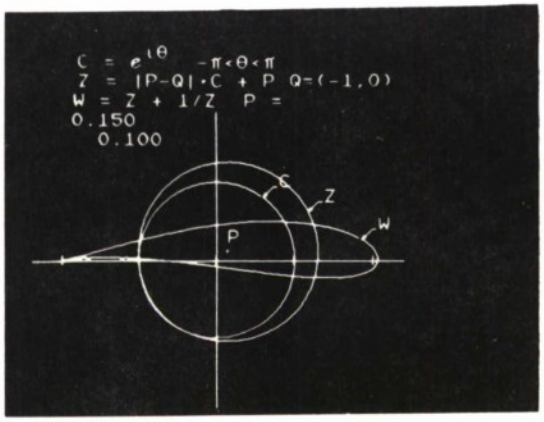
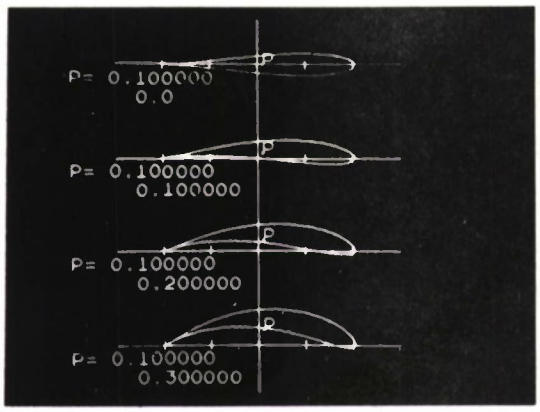


FIGURE 2.4

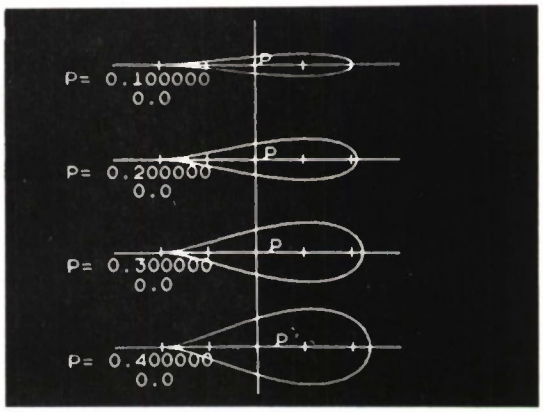
Spiral in Space



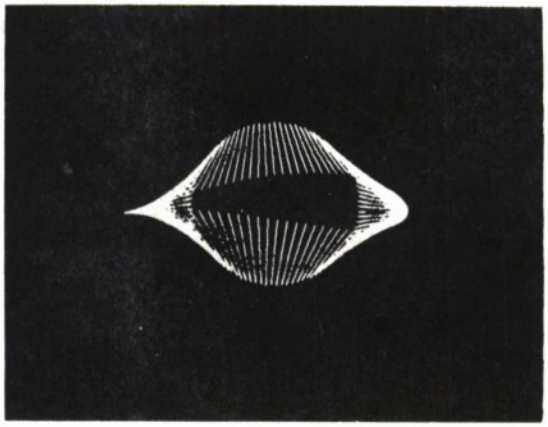
(a)



(b)



(c)



(d)

FIGURE 2.5

From Circle to Wings

within a membrane coat, in moving about the inside of a cell into which they have migrated. The question in how much of the mechanism of fluid transport within the cell can be accounted for by Brownian motion, the random darting common to small particles.

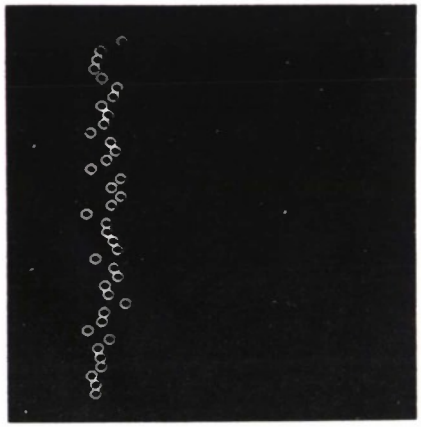
Figure 2.6a shows one style of acting. The boundaries at left and right represent cell walls. The vesicle, played by a circle, starts somewhere between the walls, as shown at the top. At each successive time interval, played by successive positions downward, the vesicle takes a step at random to the left or the right. In the short time interval portrayed, not much has happened. That, in itself, is a lesson: This kind of imitative play-acting, called the Monte-Carlo method, is so tedious and expensive, that it is used in practice only when all else fails, as in complex problems of nuclear reactor design.

In this case, man can endow the computer with the apparatus for solving differential equations, led by a diffusion equation. Figure 2.6b is a snapshot of a solution of this equation at one point in time, showing the concentration, or distribution pattern, of many vesicles all assumed to have started at the same distance away from the walls. Most are still near the starting point, but some have drifted farther away. Whether or not diffusion is quick enough, and whether enough fluid may be carried to the walls this way to account for experimental results soon becomes apparent. Figure 2.6c shows how much flows out of the left side of the cell (top curve) and the right (bottom curve) in the time following the injection of vesicles initially concentrated as shown in Figure 2.6b.

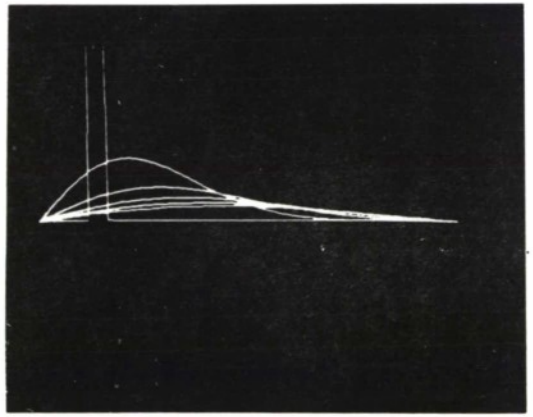
The THE BRAIN has served us, from a first baby-step to a frontier of research. It cares little who uses it: teacher, student, or expert practitioner. It can serve widely-shared goals -- teaching of analytic geometry -- or particular ones -- exploring a unique research problem. The student may use it to solve prescribed exercises, or avail himself freely of as much of its mathematical and graphical power as he is able to use.

These are the possibilities I see. Programs for displays like the preceding may be prepared in advance, and copies mass-produced for use on any computer with THE BRAIN system. Or, they may be generated on the spot, as in answer to "what if" questions. Displays may be viewed by a lone learner or, when amplified by television camera and monitors, by a large group. They may be captured on movie or still film and, as in this book, stray far from their source in multiple inexpensive copies. This wide latitude should permit balancing needs and costs as appropriate in various circumstances.

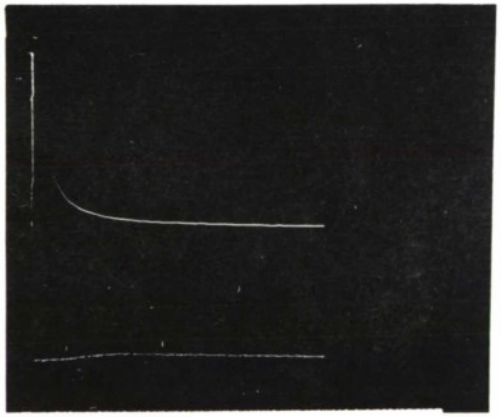
Private tool for the learner, animated blackboard for the lecturer, or anything in between: THE BRAIN plays no favorites with processes. It may even be "customized" to create the illusion of individual tailoring,



(a) Lateral motion of vesicle between cell walls 2000 Angstrom units apart over 2 milliseconds



(b) Concentration versus position between cell walls at 0.01 second intervals following injection at 300 Angstrom units from the left



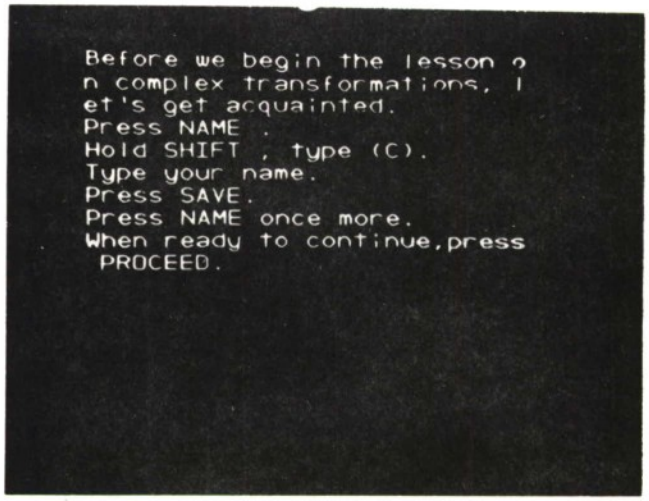
(c) Flux out of left cell wall (top) and right cell wall (bottom) from time of injection to 0.07 seconds later

FIGURE 2.6

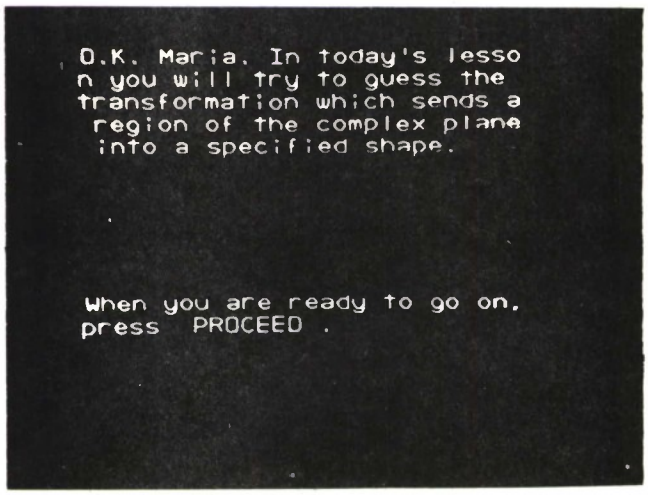
THE BRAIN Acting out Fluid Transport in a Cell

then greet you by filling the blank in "OK _____" with a name it asked you to give it. The straightjacket of multiple choice may be strapped on the learner as in Figure 2.7 or he may be allowed an occasional escape into free play, or left entirely on his own. The choice remains, as it should, with the people, not the instrument.

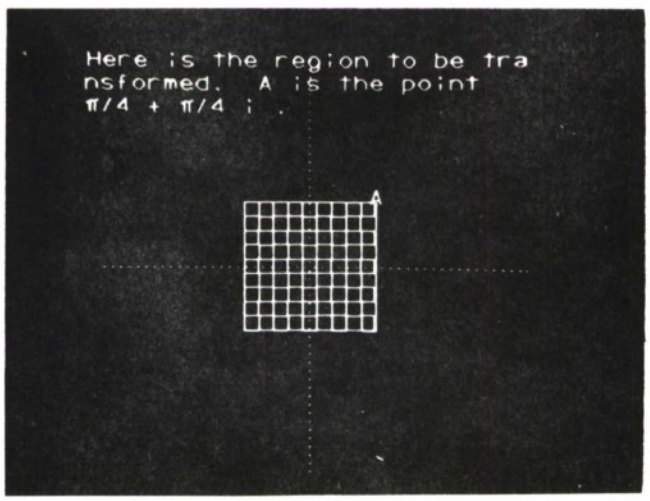
Doing justice to the possibilities of THE BRAIN or accounting fully for its high cost or for its many technical limitations is beyond the scope of this essay. My aim here was not to claim an addition to current practice, but only to show explicitly one of the exciting promises that I see.



(a)



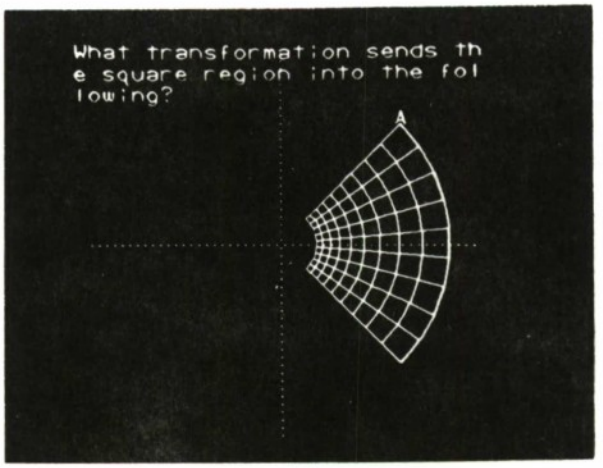
(b)



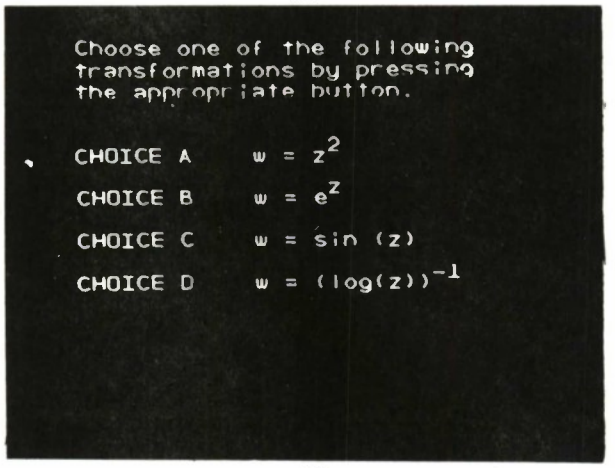
(c)

FIGURE 2.7

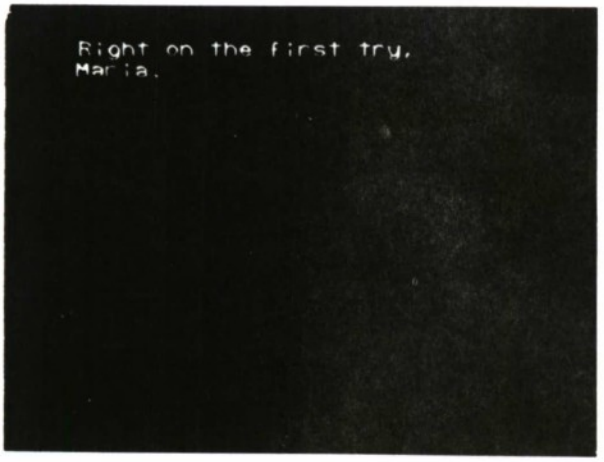
Programmed Instruction on THE BRAIN



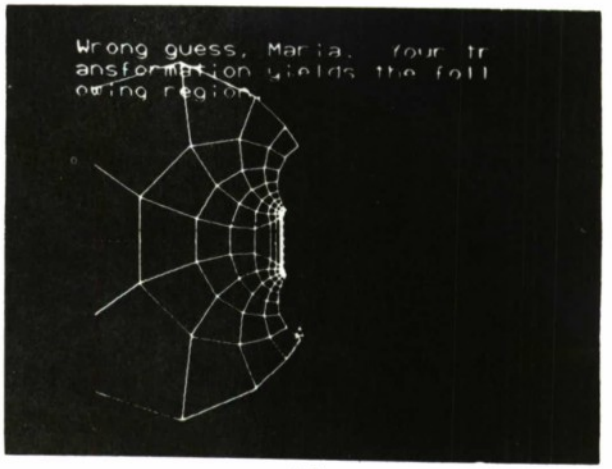
(d)



(e)



(f)



(g)

FIGURE 2.7 (cont.)

Programmed Instruction on THE BRAIN

Appendix I

Contributing Staff

Anthony G. Oettinger, Director

Adrian Ruyle, Assistant Director

Robert Anderson

Arra Avakian

Alfred Bork

William Bossert

Peter Christy

Robert DesMaisons

Richard Fateman

Robert Fenichel

Michael Fischer

Yehoshafat Give'on

Martha Greenberg

Inez Hazel

John Lipson

Albert Meyer

Charles Prenner

Randall Spitzer

Ben Wegbreit

Kenneth Winiecki

Andrew Zucker

Appendix II

Publications

a) Related to THE BRAIN

Bossert, W., and Schwartz, W. B., Relation of Pressure and Flow to Control of Sodium Reabsorption in the Proximal Tubule. Am. J. Physiol. 213 No. 3 793-802 (1967).

A study of a function of the kidney by means of mathematical modelling on an interactive computing system.

Oettinger, A. G., The Uses of Computers in Science. Scientific American, Vol. 215, No. 3, Sept., 1966, pp. 160-172.

Examples of the computer's role today as a research instrument, with speculation about future applications.

Oettinger, A. G., and Marks, S., Run, Computer, Run: The Mythology of Educational Innovation, Harvard University Press (to be published in May, 1969.)

A critical analysis of educational technology

Ruyle, A., Brackett, J., and Kaplow, R., The Status of Systems for On-Line Mathematical Assistance. Proc. Natl. Conf. ACM, Thompson Books (1967).

General recommendations for design of interactive mathematics-oriented computing systems, based on an examination of four operational examples: AMTRAN, The Lincoln Reckoner, MAP, and the Culler-Fried system.

Ruyle, A., The Use of Computer-Driven Displays in Undergraduate Mathematics Instruction. IEEE NEREM Record (1967).

Brief illustration of the application of low-cost graphic terminals in teaching functional analysis and differential equations.

Ruyle, A., The Development of Systems for On-Line Mathematics at Harvard, in Interactive Systems for Experimental Applied Mathematics, Klerer, M. and Reinfelds, J., eds. Academic Press (1968).

Ideas behind the development of three interactive computing systems by Project TACT. Includes an examination of design criteria, description of appearance of the systems to the user, and an overview of system architecture for the latest system, THE BRAIN.

b) Related to Computer Graphics and Networking

"A head-mounted three dimensional display," by I. E. Sutherland, AFIPS conference Proceedings 1968, Vol. 33, pp. 757-764.

"A clipping divider," by R. F. Sproull and I. E. Sutherland, AFIPS conference Proceedings 1968, Vol. 33, pp. 765-775.

"A futures market in computer time," by Ivan E. Sutherland, Communications of the ACM, Vol. 11, No. 6, June 1968.

"Real time color stereo computer displays, " by R. Land and I. E. Sutherland, Applied Optics, March 1969.

"Fast drawing of curves for computer display," by D. Cohen and T. M. P. Lee, AFIPS conference Proceedings 1969, Vol. 34, pp. 297-307.

"A class of surfaces for computer display," by T. M. P. Lee, AFIPS conference Proceedings 1969, Vol. 34, pp. 309-319.

"The blind leading the blind," by R. I. Land, Computers and Humanities.

"Computer Art," by R. I. Land, Leonardo (International Art Journal, Paris).

"Graphic input/output of nonstandard characters," by H. Hayashi, S. Duncan and S. Kuno, Comm. ACM 11, 9 (Sept. 1968) pp. 613-618.

"Three dimensional curves and surfaces for rapid computer display," by T. M. P. Lee, Harvard TR-69-1, Contract F19628-68-C-0379.

"Incremental methods for computer graphics," by D. Cohen, Harvard TR-69-2, Contract F19628-68-C-0379.

"Computer-assisted design of complex organic molecular syntheses," by E. J. Corey and W. Todd Wipke, to be published in "Science".

"The Voynich Manuscript," by Jeffrey Krischer. A term paper for course Ling. 205, Harvard University, Spring 1969.

Appendix III

Project Reports

1. Complete documentation is available for THE BRAIN in four manuals:

Primer	(General introduction and examples of system use)
User's Reference Manual	(Catalogue of facilities available to users)
System Architecture	(Description of system design and algorithms)
System Operation and Maintenance	(Guide for importers of the system)

2. Ninety formal memoranda document all phases of project activity.

3. Over 40 student seminar papers examine system design and use in fields ranging from engineering design to high school teaching.

4. Three scientific reports describe contributions to computer science:

   Anderson, R. H., Syntax-Directed Recognition of Hand-Printed Two-Dimensional Mathematics. TACT Report No. 1. Aiken Computation Laboratory, January 1968.

   Lewis, H. R., Two Applications of Hand-Printed Two-Dimensional Computer Input. TACT Report No. 2. Aiken Computation Laboratory, May 1968.

   Kalin, R., List Processing and Project TACT. TACT Report No. 3. Aiken Computation Laboratory, May 1968.

Appendix IV

Project History

The broad goals of Project TACT have been to determine what teaching and learning processes can best take advantage of new technology in computer hardware and software.

The history of the project research has run roughly as follows:

Several college courses which might benefit from computer assistance were analyzed for their actual and potential content. This was done in part by organizing a graduate seminar, Applied Mathematics 271, in the spring term of 1965, in which ten graduate students studied a variety of on-line computing systems and wrote term papers on the application of computer technology to learning in ten segments of a course on calculus.

To gain system design experience, Project TACT began in 1965 as implementation of the Culler-Fried OLC system on Project MAC's CTSS. This system, called TOCS, soon incorporated a number of new ideas and, by early 1967, when its evolution was discontinued due to shortage of time and space on CTSS, TOCS had become quite distinct from its OLC antecedent. A graphic terminal, installed at Harvard for remote use of TOCS, allowed use of the system for demonstrations and research. TOCS was also used by other CTSS customers.

Two terminals connecting to Culler's OLC in Santa Barbara, California, were acquired in late 1965. The OLC was used as an object of study by the graduate seminar, AM 271, in the spring of 1966, and saw considerable action in classroom teaching, using a TV camera and monitors supplied by Harvard University.

TACT's in-house system, THE BRAIN (The Harvard Experimental Basic Reckoning And Instruction Network), was built on experience in the implementation of TOCS and the use of both TOCS and the OLC at Harvard. Initial specifications, machine selection, and broad system layout were accomplished by the summer of 1966. The seminar, AM 271, contributed further to designing THE BRAIN in the spring of 1967. By the summer of 1967, the system was fleshed out and beginning to show signs of operation; and by the spring of 1968 was sufficiently advanced to be used in experimental classroom teaching by members of the AM 271 seminar.

Throughout this period, difficulties with IBM -- including insufficient documentation, hardware gaffes, and carrying their crude operating system OS through 5 successive releases -- slowed development of THE BRAIN to the extent that Project TACT, in conjunction

with the Harvard Computing Center, has demanded from IBM a rebate of $37,942 plus 10% reduction in continuing monthly computer rentals.

At the time of expiration of the SD-265 Contract, THE BRAIN is a working interactive system which provides graphic and computational assistance in a variety of mathematical and statistical applications. This system runs on an IBM 360 model 50 computer under the standard IBM operating system OS, and requires only minor OS appendages and the terminal hardware to be exported from Harvard.

The main stream of project effort was supplemented by a related but fairly distinct subproject begun in April 1967 to investigate the possibilities of computer graphics and computer networking. Under the direction of Ivan Sutherland, this project used a PDP-1 computer with attached CRT display scopes as its main facility. The major developments of this project were:

1. Linking the project's PDP-1 computer with the Harvard Computing Center's time sharing SDS-940 computer by means of a high speed parallel data channel. While this necessitated some hardware design, it primarily involved the construction of an integrated operating system to make the facilities of the 940 available to graphics programmers on the PDP-1.

2. Development of algorithms to display objects with hidden lines removed.

3. Development of programs to recognize characters drawn on a RAND tablet. These programs display on a CRT scope a standard form of the character recognized, and allow corrective interaction with the user in case of error.

4. Development, in conjunction with the Chemistry Department, of programs to enable a chemist to input a chemical molecule, observe it on a scope, and change parts of it to see the implications of these changes on the overall molecule.

5. Development of programs for the input, editing, storage, retrieval and composition of Chinese characters using a scope and light pen.

In addition, roughly a dozen student projects were undertaken by members of Professor Sutherland's seminar AM 252 on computer graphics. These include curve fitting with splines, cabinet projections of cubic parabolas, display of parsed structures, an airplane simulator, etc.

During development of THE BRAIN, Project TACT developed new computer technology for eventual incorporation into the system.

1. Robert Fenichel, of the project, developed a flexible system for algebraic manipulation. This system, called FAMOUS (Fenichel's Algebraic Manipulator for On-line USe) was implemented on Project MAC's CTSS in LISP 1.5.

2. Robert Anderson, of the project, devised and implemented a parser for two-dimensional hand-written mathematics, as described in TACT Report No. 1. (See Appendix III.) Anderson's parsing program provides not only the ability to parse formulae where limits of integration are tacked to the ends of an integral sign and numerators appear above denominators, but also the ability to understand three diagrams and two-dimensional diagrams of chemical bonds. The complete implementation of this parser, however, needs mating to an on-line computing system, such as THE BRAIN, to fully explore its value in working situations.

3. Harry Lewis, supported by Project TACT, transcribed a subset of Anderson's work, combined it with a character recognizer devised by the Networking and Graphics Research Project, and added routines for floating point arithmetic and graphic display, to produce a nicely engineered package for hand-written specification of complex transformations on the Networking and Graphics PDP-1. Lewis' package, called SHAPESHIFTER, is described in TACT Report No. 2. (See Appendix III.) SHAPESHIFTER uses three scopes: One for the complex plane, a second for the transformed plane, a third for scratch use in conjunction with the tablet. While this system demonstrates the utility of one aspect of Anderson's parser, and has been employed fairly extensively in demonstrations, its potential for teaching complex variables and its value for developing subjective insights in this field have not yet been examined.

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* Harvard University Cambridge, Mass.	2a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED 2b. GROUP N/A

3. REPORT TITLE

FINAL REPORT SD-265 (PROJECT TACT)

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Final Report, July 1964 through March 1968.

5. AUTHOR(S) *(First name, middle initial, last name)*

None

6. REPORT DATE 1 October 1968	7a. TOTAL NO. OF PAGES 39	7b. NO. OF REFS 26
8a. CONTRACT OR GRANT NO. F19628-68-C-0300 b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) ESD-TR-69-121 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*	

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Directorate of Planning and Technology, Electronic Systems Division, AFSC, USAF, L G Hanscom Field, Bedford, Mass. 01730

13. ABSTRACT

The project objective has been to determine what creative thought processes can best take advantage of new technology in computer hardware and software. The plan has been to acquire or develop on-line computer systems of significant mathematical power, and to explore their use in vivo in teaching and research situations. The main product of project research is THE BRAIN (The Harvard Experimental Basic Reckoning And Instructional Network), an interactive computing system which operates on a standard IBM 360 Model 50 under the standard IBM operating system OS. This working system is easy and flexible to use in mathematical and engineering investigations as well as in teaching, and has been engineered for straightforward exportation from Harvard.

DD FORM 1 NOV 65 1473

Unclassified

Security Classification

14. KEY WORDS	LINK A ROLE	LINK A WT	LINK B ROLE	LINK B WT	LINK C ROLE	LINK C WT
On-line						
Interactive						
Graphical						
Mathematical						
Statistical						
Remote-Access						
Function-Oriented						
Human Engineering						
Exportable						
Education						
Engineering						