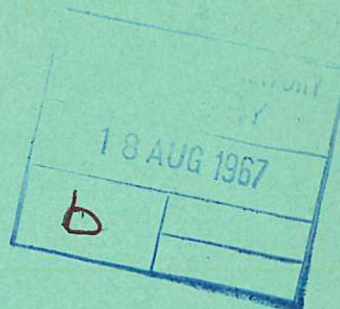


This document is intended for publication in a journal, and is made available on the understanding that extracts or references will not be published prior to publication of the original, without the consent of the author.



United Kingdom Atomic Energy Authority

RESEARCH GROUP

Preprint

## A GRAPHICAL OUTPUT LANGUAGE AND ITS IMPLEMENTATION

F. M. LARKIN

Culham Laboratory  
Abingdon Berkshire

1967



Enquiries about copyright and reproduction should be addressed to the  
Librarian, UKAEA, Culham Laboratory, Abingdon, Berkshire, England

A GRAPHICAL OUTPUT LANGUAGE  
AND ITS IMPLEMENTATION

by

F.M. LARKIN

(Submitted for publication in 'Computer Journal')

ABSTRACT

An outline description is given of a graphical output language (GHOUL), designed for use with automatic digital computers and plotting equipment. An attempt has been made to ensure that the commands constituting the language are comprehensive and as powerful as possible, consistent with the requirement that they correspond closely with natural manual, or mental, operations that a human being would employ while plotting a graph.

An implementation of the language in the form of a graphical output system (GHOST) is also described.

UKAEA Research Group  
Culham Laboratory,  
Abingdon,  
Berks

June 1967 (D/S)



## C O N T E N T S

	<u>Page</u>
1. INTRODUCTION	1
2. DESIGN CRITERIA FOR GHOUL	4
3. OUTLINE OF THE STRUCTURE OF GHOUL:	6
(a) Plotter Configuration	6
(b) Mapping from Mathematical Space to Plotter Space	7
(c) Limiting Rectangles	9
(d) "Selection" of a Space	10
(e) Straight Lines and Curves	11
(f) Characters and Plotting Symbols	14
(g) Numbers	16
(h) Miscellaneous Facilities	17
4. THE GHOST IMPLEMENTATION OF GHOUL	17
5. FUTURE DEVELOPMENTS	21
6. ACKNOWLEDGEMENTS	22
7. REFERENCES	22
APPENDIX A : Summary of GHOST Commands	23
APPENDIX B : The GHOST Standard Character Sets	27
APPENDIX C : Example of the use of GHOST	34

## 1. INTRODUCTION

Whatever we may think of the ability of modern computing machines to process large quantities of information very rapidly, most users (and operators!) will agree that the ability of modern peripheral equipment to disgorge huge quantities of printed paper at a prodigious rate is not an unmixed blessing. At best, most of this printed output goes into the wastepaper basket, away to be repulped, and presumably returns once more to the computer for another cycle in the process. However, even that relatively small fraction of printed output which is retained by the computer user as "potentially valuable" is often difficult for him to assimilate in its pristine numerical form. At worst then, a super-abundance of numerical output can actually retard the activity of problem-solving. Anyone who has solved a partial differential equation on a digital computer will appreciate the need for automatic graphical representation of results. Indeed, if one agrees with Hamming (1962) that "The purpose of computing is insight, not numbers", the over-production of numbers, when the required insight derives most directly from a graph depicting qualitative features of the solution to one's problem, must seem little more than an exercise in futility.

Automatic plotters adequate for many computing purposes have been on the market for several years now, so it is reasonable to ask why they are not used at least as extensively as, for example, line printers. Considerations of

speed, reliability and cost must, of course, raise doubts in the minds of would-be users of graphical output equipment, but the author believes that two other inter-related obstacles outweigh the aforementioned mundane ones, namely:

- (i) the sheer novelty of the concept of calling for graphical representation of results as a matter of course,
- and (ii) the inconvenience of the data format for currently available plotters.

Traditionally, the graphical representation of computed, or observed, results has been a rather menial task, requiring a modicum of draughtsmanship but, nevertheless, much easier than the job of obtaining the results in the first place. However, since the advent of modern computing machines the situation has changed. Often now, the computational work is the easy part, and the graph plotting has become the major portion of the total work involved. Clearly, there is a case for further automation!

Unfortunately, currently available plotters operate from a rather limited set of basic commands, such as "lift the pen", "move the pen to position (21,35)", etc., while the computer user wishes to give instructions like "draw a graph of  $J_0(x)$  for  $0 \leq x \leq 12$ ", "plot contours of  $\left| \frac{\sin(z)}{z} \right|$  over the unit disc", "annotate the axes", etc. The situation is reminiscent of the state of computer programming before the development of high level program-



ming languages. Applying the principle that "similar problems call for similar solutions", the idea of a graphical output language becomes obvious.

It was with the above thoughts in mind that, at the beginning of 1966, the author attempted to identify the basic structure and elements of a servicable graphical output language. An obvious selection of letters from the previous phrase provided the convenient acronym GHOU<sup>L</sup> for such a language, and by a happy coincidence the companion acronym GHOST is an acceptable abbreviation for the implementation of GHOU<sup>L</sup> in the form of a graphical output system. In the following, the term GHOU<sup>L</sup> will be used to refer to a set of hardware-independent instructions which could, in principle, be implemented in any reasonably high level computing language, with a wide variety of hardware; the term GHOST will refer to the particular version of GHOU<sup>L</sup>, together with its associated hardware and software, which has been in daily use at the UKAEA Culham Laboratory since October 1966.

It has been found in practice that computer users in an open-shop, scientific, environment gain substantial benefits from a convenient graphical output system. Not only are they saved the tedium of poring over large tables of numerical results to find the information they know they are looking for, but often, from a graph, they discover useful information which may be only remotely related to the original purpose of the computation. Automatic graph plotting is a serendipitous technique!

## 2. DESIGN CRITERIA FOR GHOU

The least that may be expected of a language is that it should be capable of expressing, succinctly and unambiguously, all the concepts which it is designed to handle. Ideally, it should also be capable of indefinite extension in order to accomodate any further conceivable concepts which may be introduced at a later date - a forlorn hope! In the case of GHOU the former criterion may be reformulated as follows

- (i) The standard graphical output commands should be readily interpretable in terms of simple manual or mental operations which a human would naturally employ when writing, plotting a graph or drawing a picture.
- (ii) All of these natural, human operations should be catered for by basic commands in the language.

It would be presumptuous to suggest that any language could be perfect, so the best one can do towards satisfying (ii) is to rely on experience of users' requirements in order to define the basic elements of the language, and then to allow for additions to be made. It is virtually impossible to be sure of allowing for indefinite extensibility, but one can go some way towards this desirable goal by adhering to the following rule:

- (iii) The only operands associated with any graphical output operator (command) should be those



immediately required for definition of the operation. Furthermore, the operand lists should be as short as possible.

The implication of (iii) is that compound operations, such as "advance the plotting frame and/or draw axes" should be avoided, as far as the basic language is concerned.

This is desirable because, for example, a particular user might wish to invent a special purpose command "draw axes and annotate them" without incurring a frame advance as well.

We can summarise the above criteria by saying that the basic graphical output commands should be as natural and powerful as possible, consistent with the requirement that they must be "atomic", in the sense that a user should never feel the need for a partial operation. Of course, out of these atomic commands which constitute the basic language a user will be free to construct whatever compound commands best suit his purpose. Indeed, to carry the physical analogy still further, certain frequently used "molecular" commands might profitably be included as standard facilities.

A further, practical requirement is that:

- (iv) The basic operations should be capable of implementation on widely available plotting equipment; in practice, this means "incremental pen plotters" having no built-in, hardware characters.

The particular set of atomic commands initially chosen for GHOU is listed in appendix A and partially described in section 3. It must be emphasised that the names of these commands, limited by the FORTRAN language, are comparatively unimportant. The important features are the conceptual operations and operands. For a detailed description reference should be made to the Users' Guide (Larkin 1967). GHOU includes, in one form or another, the graphical output operations proposed by Knight (1965), but with rather more insistence upon short and "natural" argument lists.

No claim is made that GHOU is a "best" graphical output language in any sense, merely that it is a servicable one. In practice it appears to be easy enough to use, and it is not difficult to modify and extend without conflicting with existing programs which refer to it.

### 3. OUTLINE OF THE STRUCTURE OF GHOU

#### (a) Plotter Configuration

We consider a situation in which graphical information produced by a computer program may be routed simultaneously to any output channels selected from the total number available. Each channel may be identified with an automatic plotter. Accordingly, the user must be provided with commands which enable him to "switch" each individual channel on or off, as desired. Logical switching may be performed at any stage during execution of the originating program. As explained below, this idea of logical switching is extended to cover other facilities, such as broken



lines, italic characters, etc.

Provision is also made for the storage of graphical output information, for example on magnetic tape, and for the frame-by-frame retrieval of this information from the store.

(b) Mapping from Mathematical Space to Plotter Space

When drawing a graph, a human being, consciously or otherwise, employs two distinguishable frames of reference.

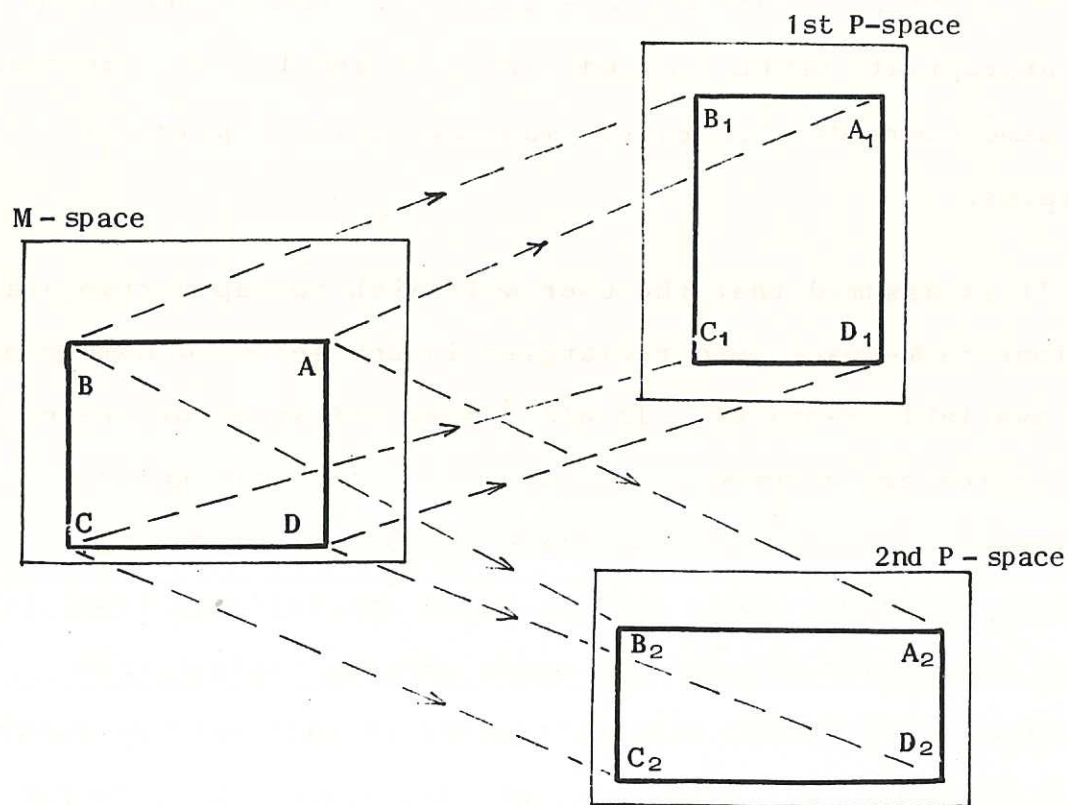


Fig.1.

Illustration of the linear mapping of the M-space region ABCD into  $A_1B_1C_1D_1$  and  $A_2B_2C_2D_2$  in two P-spaces. There is also an implied correspondence between  $A_1B_1C_1D_1$  and  $A_2B_2C_2D_2$ . The outer, limiting rectangles are mutually independent.

One such frame of reference is in plotter space (P-space), i.e. the real, physical plane of the plotting medium, in which titles, annotation, etc., are arranged in convenient positions, with relative distances usually measured in inches or centimetres. The human operator establishes a correspondence between this P-space and the mathematical space (M-space) in which his graph originates. He mentally identifies points in P-space, which he can see, with corresponding points in M-space, which he usually has to imagine. This mapping between M-space and P-space must be defined before plotting of any M-space quantities can be performed, and appropriate GHOU commands are provided for the purpose. The same commands also effect mappings between pairs of P-spaces.

It is assumed that the user will wish to map rectangular regions in M-space into rectangles in any selected number of the available P-spaces. In all cases, the sides of these rectangles are assumed to be parallel with rectangular, cartesian, co-ordinate axes fixed in the corresponding spaces; however, their shapes, sizes and off-sets from their respective origins are under program control. For example, using GHOST, mapping regions in mathematical space, hardcopy space and incremental plotter space could, be put into correspondence by means of the instructions

```
CALL MSPACE
CALL REGION (XMINM,XMAXM,YMINM,YMAXM)
CALL HRDCPY(1)
CALL CMS
CALL REGION (XMINH,XMAXH,YMINH,YMAXH)
```



```
CALL INCPLT(1)
CALL INCHES
CALL REGION (XMINI,XMAXI,YMINI,YMAXI)
```

where the arguments, in an obvious notation, are measured (in the current units) from the appropriate co-ordinate axes.

### (c) Limiting Rectangles

An important distinction is made between the rectangular mapping regions, described above, and limiting rectangles. As well as defining a mapping region in a particular space, the GHOU user may specify a limiting rectangle within which all his plotting will be confined. Thus, for example, one may plot a curve contained within the mapping region in M-space and add a title which lies within the limiting rectangle without overlapping the mapping region. Fig.1 illustrates the correspondence between mapping regions and limiting rectangles in various spaces.

It should be realised that, whereas the mapping regions are necessary in order to establish a correspondence between M-space and the various P-spaces, the limiting rectangles have been introduced in order to simplify the location of textual output. A limiting rectangle is analogous to the the smallest rectangle which just contains all the text on a page in a typical book (excluding page number and footnotes). It also serves, incidentally, to delineate that part of a P-space outside of which no plotting may occur.

(d) "Selection" of a Space

Since a human operator is constantly shifting his mental frame of reference from M-space to P-space and back, while plotting a graph, special GHOU commands are provided in order to simulate the process. This shift of attention is referred to below as selection of the corresponding space, a process which should not be confused with the operation of switching on an output channel. In GHOST the two operations are performed by the same routine, but this is not necessary and, logically, perhaps even rather undesirable.

Once any particular space is selected (regardless of whether or not it is still switched on) all subsequent dimensioned arguments will be interpreted as being measured in the units (inches, centimetres, mathematical units, etc.) which currently apply in the selected space. A P-space cannot be deselected by switching it off; - only by explicitly selecting a different space.

Thus, one might switch on several output channels, select M-space, cause a curve to be drawn (i.e. output on all currently switched-on channels), next select a particular plotter space and finally write a title, all on the same frame. Note that any information plotted when a particular P-space (or M-space) is selected will also appear, suitably scaled, on all currently switched-on output channels, since correspondences between their mapping regions will have been established either explicitly, or by default.



It should be clear that, whereas any number of the available output channels (each with its associated P-space, but not M-space, of course) may be switched-on simultaneously, only one P-space (or M-space) may be selected at any given instant. Furthermore, it is even possible to switch off a selected output channel without deselecting it.

When a user has selected a plotter space he may establish units (e.g. inches or centimetres) in that space by means of appropriate commands. These units are then understood to apply to all dimensions mentioned while the space is selected, until explicitly altered by the user's program.

#### (e) Straight Lines and Curves

Having established his mappings and units the user may proceed to draw straight lines and curves, processes which are best explained by example. Although FORTRAN examples are given below there is no inherent bias in GHOUl towards that programming language. In this section, and those following, any reference to a position or length on a graph should be interpreted as applying to the currently selected space, in the units currently applying to that space.

A straight line joining the points  $(X1,Y1)$  and  $(X2,Y2)$  may be drawn by first specifying the starting point, either by the command

CALL POINT (X1,Y1)

or by the command

CALL POSITN (X1,Y1)

and then following this, either by the command

CALL JOIN (X2,Y2)

or by the command

CALL LINE (DELTAX,DELTAY)

where, of course,

DELTAX = X2 - X1

DELTAY = Y2 - Y1

A polygonal arc may be continued to the points (X3,Y3), (X4,Y4), etc., either by the instructions

CALL JOIN (X3,Y3)

CALL JOIN (X4,Y4)

etc. etc.,

or by the corresponding LINE instructions.

In terms of normal, manual plotting operations the POSITN command lifts the pen from the paper and moves it to the position specified in the argument list; no mark is made on the paper, but the "current character typing position" (see subsection (f), below) has been altered. The POINT command lifts the pen from the paper and sets it down again at the specified point, marking a dot on the paper in the process; it does not alter the current character typing position.

Four basic commands are provided in GHOU for the purpose of drawing smooth curves which interpolate a finite number of given points. These cover the following four most frequently occurring situations:-

- (i) Non-periodic and single-valued graphs;
- (ii) Periodic and single-valued graphs;
- (iii) Open-ended, and possibly multi-valued, curves;
- (iv) Closed and multi-valued curves.



In all cases, the shapes of the interpolated graphs or curves will be invariant under a uniform translation of the co-ordinate axes. A rotation of the co-ordinate axes will result in a change of shape in cases (i) and (ii), but not in cases (iii) and (iv). Thus, the instruction

CALL GRAPHN (XV,YV,M,N)

will result in the drawing of a smooth graph through the points  $((XV(I), YV(I)), I = M,N)$ . The abscissae are assumed to satisfy the conditions

$XV(M) < XV(M+1) < \dots < XV(N-1) < XV(N)$ .

The curve  $Y(x)$  will be single-valued and its shape will be invariant under a translation, but not a rotation, of the co-ordinate axes.

The instruction

CALL GRAPHP (XV,YV,M,N,PERIOD)

performs a function similar to that of GRAPHN, except that the interpolating curve produced will be periodic with period PERIOD.

The statements

CALL CURVEO (XV,YV,M,N)

and

CALL CURVEC (XV,YV,M,N)

will each result in the drawing of a smooth curve through the  $(N-M+1)$  points  $((XV(I), YV(I)), I = M,N)$ . In both cases the shape of the interpolated curve will depend only upon the relative point positions and not upon the position or orientation of the co-ordinate axes; the order of the points along the curve will agree with their order as

implied by their positions in the vectors XV and YV. In the case of CURVEC the interpolated curve will be closed; in the case of CURVEO the curve will be open-ended, starting from the point (XY(M), YV(M)) and ending at the point (XV(N), YV(N)). Of course, the latter pair of commands will be used only when a metrical relationship is known to exist between the two co-ordinate axes, as, for example, when spot positions on a contour are known and it is required to approximate the contour itself.

The one remaining curve-drawing command

CALL GRAPHF (FUNXN)

where FUNXN is the name of any EXTERNAL function sub-program.

e.g. FUNCTION FUNXN(X),

will result in the graph of FUNXN(X) being plotted over that part of the range of its argument X which lies within the mapping region of the currently selected space.

#### (f) Characters and Plotting Symbols

Paradoxically, one of the most important requirements of a graphical output language is the capability of handling non-graphical information. For this reason, liberal facilities are provided in GHOU for the purpose of annotating graphs with alphabetic and numerical characters, as well as symbols indicating plotting positions.

The GHOU user may call upon several character sets, each consisting of 64 characters. The intention is that

any installation implementing GHOUl may include character sets most suitable to its own needs, although agreement has been reached with the National Physical Laboratory (Heap and Nott, 1967) on the 7 standard GHOST sets listed in Appendix C. Within each set an individual character may be distinguished, either by means of its serial number, which can assume any of the values 0,1,2, . . . . , 63, or by referring to it explicitly. Of course, the latter mode will be limited to characters catered for by the available computer input devices.

Nominal character size is fixed by the command

CALL CRSize (HEIGHT)

where HEIGHT is the nominal height, in previously specified units, of the subsequently plotted characters. This "nominal height" is larger than the actual height, being analogous to line depth on a printed page. Nominal character width is automatically set to be 3/4 of the previously specified nominal height.

A single character may be located on a page, either by giving the cartesian co-ordinates of its center, or by specifying the number of lines (measured downwards from the upper edge of the limiting rectangle) and the number of spaces (measured to the right from the left-hand edge of the limiting rectangle) on a (space, line) co-ordinate



grid. Characters may be output individually, or in strings.

For the purpose of character output two modes, plotting mode and typing mode, are distinguished. All plotting mode commands must supply the (x,y) co-ordinates of the position at which a character is to be plotted.

Whenever a character is output a current typing position is established, immediately to the right of the position of the last character. If a typing mode command then follows, the character next referred to is output at the current typing position. Of course, no positional information is supplied with a typing mode command. The GHOU commands available for performing these operations, and other standard typewriter functions, are listed in Appendix A(iv). Facilities for suffixing, superfixing and italicising characters are also included.

#### (g) Numbers

Provision is made in GHOU for the plotting (or typing) of numbers in three different formats, as follows:

Format	Examples
E - format (floating point)	0.12345E 2, -0.987E -6
F - format (fixed point)	12.345, -0.000987
I - format (integer)	12, -98

Six commands provide for number plotting or typing in any of these three formats. In the case of E - format and F - format the number of decimal places required must be specified.

#### (h) Miscellaneous Operations

In addition to the above classified operations, it has been found useful to include in GHOUL the miscellaneous group listed in Appendix A(v). These include facilities for causing lines and curves to be drawn in broken form, rotating and expanding the picture and altering the line plotting density. Also a number of compound operations are included, such as adding scales and graticules to pictures, and drawing borders and contours. A lengthy description of these facilities would be out of place in this paper, but details may be found in the Users' Guide (Larkin 1967).

#### 4. THE GHOST IMPLEMENTATION OF GHOUL

As far as the GHOST user is concerned, the commands constituting basic GHOUL exist in the form of FORTRAN subroutines. His program runs on the E.E.C. KDF9 computer, using the Egdon system (Burns, et al. 1966), when these routines may be invoked simply by naming them in standard CALL statements. Extra commands, concerned with specific hardware operations, and distinguished by asterisks in Appendix A, are also provided. Through these routines (GHOUL commands plus the extra ones) the user has access to the following plotting equipment:-

- (1) A Benson-Lehner model 120 microfilm printer/plotter.

This provides output in the form of

- (a) 35 mm. film, nominal frame size 0.7 x 0.68 inches, (17.75 x 17.25 mm.)

- (b) 16 mm. film, nominal frame size 0.275 x 0.275 inches,  
(7.0 x 7.0 mm.)
- (c) "Hardcopy", nominal frame size 7.0 x 6.8 inches,  
(17.75 x 17.25 cm.)

The hardcopy, each frame of which is effectively a photographic enlargement of the corresponding microfilm frame, can be obtained in two forms. The so-called "quick and dirty" hardcopy is produced, soon after the originating job has run on the computer, on a special type of photographic paper which is developed and dried, but not properly fixed. Such a record fades within a few days, especially if exposed to strong sunlight, but it enables a usable form of the graphical output to be provided quickly. If required, "permanent" hardcopy can be provided. This consists of photographic paper records which have been processed by a standard (time consuming) technique, and is not normally produced until the evening shift of the day on which the originating job is run on the computer. It should be noted that 35 mm. film and 16 mm. film cannot be obtained concurrently, but either may be obtained concurrently with hardcopy.

## (2) An Incremental Pen Plotter

This is not operational at the time of writing, but it is expected to have two (mutually exclusive) plotting pens and a paper width of up to 30"; provision has been made for it in the GHOST software.



(3) An "on-line" graphical input-output display

This comprises a light-pen and cathode ray tube, controlled via a PDP8 computer peripheral to the KDF9. Nominal frame size 9.375 x 9.1 inches.

From the point of view of the user, graphical output channels 1a or 1b, 1c, 2 and 3 may be regarded as physically separate devices which may be controlled, independently or jointly, by his computer program. On entry to the user's program these devices will be in the "switched-off" state; they must be "switched-on" before use and may be "switched-off" again when no longer required. Instructions for effecting this symbolic switching are as follows:

```
CALL FILM35(ONOFF)
CALL FILM16(ONOFF)
CALL HRDCPY(ONOFF)
CALL INCPLT(ONOFF)
CALL DISPLY(ONOFF)
```

In all cases a value of "0" for the input argument ONOFF signifies "off" and a value of "1" signifies "on".

Since GHOST was initially an experimental system, its software was written largely in FORTRAN, rather than the more basic KDF9 Usercode. The disadvantage of a loss in operating efficiency has been more than compensated by the ease with which the system has been modified and extended, even by students visiting the Laboratory for a few weeks during summer vacations! Now that the main features of the system are relatively stable, efficiency is being improved by judicious replacement of FORTRAN routines by Usercode versions. However, a complete FORTRAN version

of the GHOST software is kept up to date for archival purposes, to ease the task of re-implementing GHOUL on any future computer which the Laboratory might use, and also to enable other installations, if they so desire, to take over the system as a going concern in a high-level programming language.

The GHOST operating arrangement which fits most conveniently with other usage of the Culham KDF9/PDP8 computing system is the following. In order to minimise demands on core storage during execution of the user's program, the GHOST operations invoked by that program merely write their operands sequentially onto a magnetic tape. At some convenient later time, usually every 1 or 2 hours, the machine operators rewind the storage tape, which in general contains graphical information from several different users' jobs. The main GHOST software is then used to process this information, i.e. to generate basic input data for the individual plotters. During the same processing run on the KDF9, information for the B-L 120 is written onto an IBM tape; information for the on-line CRT display is transmitted to magnetic disc files from which it can be accessed by on-line consoles and subsequently handled by special PDP8 software. The IBM tape is then dismounted from the KDF9 and run at leisure on the off-line Benson-Lehner 120 printer/plotter. Information for the incremental pen plotter, when it is installed, could be output directly from the KDF9 or via any convenient buffer store.

Since speed of response is an important feature of on-line working, a streamlined version of GHOST, catering only for the CRT display and by-passing the intermediate storage tape, is also available. This provides the on-line user with a rapid-response graphical output facility, at the expense of an effective loss in available core storage.

It should be realised that the normal GHOST user need know nothing about the mechanics of the processing of his graphical output. As far as he is concerned, the plotters may all be regarded as on-line, peripheral devices.

## 5. FUTURE DEVELOPMENTS

The GHOU commands whose GHOST implementations are listed in Appendix A should not be regarded as a complete set; they merely constitute the set which was felt to fulfill the basic minimum requirements of an open-shop, scientific computing establishment. It is anticipated that, as with any working language, new words and concepts will be coined as and when the need arises. For example, additional facilities such as stereoscopic plotting, logarithmic graticules and non-linear mappings (conformal, or otherwise) are quite easy to provide in terms of the existing facilities. Users may also wish to compose extra character sets consisting, for example, of stylised electronic components.



## 6. ACKNOWLEDGEMENTS

The author gratefully acknowledges the help of all those who have assisted in developing the GHOST software, both permanent staff and visitors to the Laboratory. In particular, B. Brunning, Mrs. G.M. Leslie, J. McDowell and A. Sykes have all made lasting and valuable contributions.

## 7. REFERENCES

- HAMMING, R.W., "Numerical Methods for Scientists and Engineers", McGraw-Hill, 1962.
- HEAP, B.R. and NOTT, C., private communication.
- LARKIN, F.M., "A user's Guide to the Culham Graphical Output System", (in preparation).
- BURNS, D., HAWKINS, E.N., JUDD, D.R., and VENN, J.L., "The Egdon System for the KDF9", Comp.J., Vol.8, No.4, Jan. 1966.
- KNIGHT, G.B., "Standard Graphic Output Subroutines as Proposed by a Committee of SHARE". Proc. 4th. Meeting of UAIDE New York, N.Y. October, 1965.

# APPENDIX A

## Operations Available in the UKAEA Culham GHOST implementation of GHOU

\* Indicates that the operation refers to specific hardware. Other implementations might alter or delete these operations, since they are not strictly part of the GHOU graphical output language.

### (i). Control of Graphical Output Hardware

Operation	FORTTRAN Subroutine
Switch 35 mm. camera on or off; select if ONOFF = 1	FILM35(ONOFF) *
Switch 16 mm. camera on or off; select if ONOFF = 1	FILM16(ONOFF) *
Switch Hardcopy camera on or off; select if ONOFF = 1	HRDCPY(ONOFF) *
Switch Incremental Plotter on or off; select if ONOFF = 1	INCPLT(ONOFF) *
Switch On-line Display on or off; select if ONOFF = 1	DISPLY(ONOFF) *
Advance frame on 35 mm. camera	FRAM35 *
Advance frame on 16 mm. camera	FRAM16 *
Advance frame on Hardcopy camera	FRAMHC *
Advance frame on Incremental Plotter	FRAMIP *
Advance frame on On-line Display	FRAMDP *
Advance frame on all "on" devices	FRAME
Select red pen on Incremental Plotter	REDPEN } mutually *
Select black pen on Incremental Plotter	BLKPEN } exclusive *
Store subsequent graphical output commands on tape NTAPE	STORE(NTAPE)
Suspend storage on tape NTAPE	SUSPND(NTAPE)
Cancel storage on tape NTAPE	CANCEL(NTAPE)
Retrieve frames numbered M, M+1, ... N, from storage tape NTAPE	RETRVE(NTAPE,M,N)
Repeat preceding BL-120 frame NTIMES times	REPEAT(NTIMES) *

(ii) Mapping From Mathematical Space to Plotter Space

Operation	FORTRAN Subroutine
Select mathematical space	MSPACE
Define mapping region	REGION(XMIN, XMAX, YMIN, YMAX)
Define limiting rectangle	LIMITS(XMIN, XMAX, YMIN, YMAX)
Define plotter space units to be inches	INCHES
Define plotter space units to be cms.	CMS
Define plotter space units to be mms.	MMS

(iii) Straight Line and Curve Drawing

Operation	FORTRAN Subroutine
Lower plotting pen onto point (X,Y)	POINT (X,Y)
Join current plotting point to point (X,Y) by a straight line	JOIN (X,Y)
Draw straight line with vector components (DELTA X, DELTA Y) from current point	LINE(DELTA X, DELTA Y)
Draw smooth, open-ended, rotationally invariant curve through given points	CURVEO(XV, YV, M, N)
Draw smooth, closed, rotationally invariant curve through given points	CURVEC(XV, YV, M, N,)
Draw non-periodic graph through given points	GRAPHN(XV, YV, M, N,)
Draw periodic graph through given points	GRAPHP(XV, YV, M, N, PERIOD)
Draw graph of given, EXTERNAL function	GRAPHF(FUNXN)



(iv) Output of Characters and Point Plotting Symbols

Operation	FORTRAN Subroutine
Select character set NOSET	CTRSET(NOSET), CRSET(NOSET)
Define character size	CRSIZE(HEIGHT)
Define character typing position in (x, y) co-ordinate system	POSITN(X, Y)
Define character typing position in (space, line) co-ordinate system	PLACE(NSPACE, NLINE)
Space NSPACE algebraic character positions to the right	SPACE(NSPACE)
Effect NLINES algebraic line-feeds	LINEFD(NLINES)
Perform the combined operation "carriage-return, line feed"	CRLNFD
Plot numbered character at point (X, Y)	PLOTNC(X, Y, NOCHAR)
Type numbered character at current position	TYPENC(NOCHAR)
Read first NC characters of string into vector PHRASE	READCS(PHRASE, NC)
Plot first NC characters of string Phrase	PLOTCS(X, Y, Phrase, NC) †
Type first NC characters of string Phrase starting at current typing position	TYPECS(Phrase, NC) †
Plot number Z in E-format, NDP decimal places, decimal point at point (X,Y)	PLOTNE(X, Y, Z, NDP)
Plot number Z in F-format, otherwise as for PLOTNE	PLOTNF(X, Y, Z, NDP)
Plot number I in I-format, otherwise as for PLOTNE	PLOTNI(X, Y, I)
Type number Z in E-format, with sign position located at current typing position	TYPENE(Z, NDP)
Type number Z in F-format, otherwise as for TYPENE	TYPENF(Z, NDP)
Type number I in I-format, otherwise as for TYPENE	TYPENI(I)
Select or deselect italic characters	ITALIC(ONOFF)
Select suffix character plotting mode	SUFFIX
Select superfix character plotting mode	SUPFIX
Revert one level towards normal character mode	NORMAL
Select large hardware characters on BL-120	LARGEC *
Select small hardware characters on BL-120	SMALLC *
Select light plotting density on BL-120	LIGHT *
Select heavy plotting density on BL-120	HEAVY *
Select standard hardware character orientation on BL-120	STNDRD *
Select sideways hardware character orientation on BL-120	SIDWYS *
† Note that "Phrase" may be either a vector or a literal character string	

(v) Miscellaneous Commands

Operation	FORTRAN Subroutines
Draw rectangular box of prescribed dimensions	BOX(XMIN,XMAX, YMIN, YMAX)
Rotate subsequent graphical output through ANGLE, about the point (XC, YC)	ROTATE (XC, YC, ANGLE)
Select line density of subsequent output	DENSITY (LTORDK)
Ncminate subsequent lines to be drawn in a broken style	BROKEN(N1, N2, N3, N4)
Revert to full lines	FULL
Select or deselect diagnostic printing	DGNSTC(ONOFF)
Select or deselect standard frame annotation	ANNOTE(ONOFF)
Draw graticule spanning mapping region	GRATIC
Annotate scales on mapping region	SCALES
Draw and annotate co-ordinate axes	AXES
As for GRATIC, SCALES and AXES, except that the specified increments, for purposes of annotation, are given as arguments	GRATSI(SPINCX, SPINCY) SCALSI(SPINCX, SPINCY) AXESSI(SPINCX, SPINCY)
Connect an ordered sequence of points by straight lines, or plot characters	PTPLOT(X, Y, M, N, NC)
Select, or deselect, informative printing of arguments of certain routines	GARGS(ONOFF)
Draw specified contours in surface defined by spot heights	CONTRA(A, JXS, JXF, NX, JYS, JYF, NY, H, M, N)
Draw rectangular border around mapping region	BORDER
Expand subsequent dimensions, about (XC, YC) by given RATIO	EXPAND(XC, YC, RATIO)
Combined operation of EXPAND and ROTATE	EXPROT(XC, YC, RATIO, ANGLE)
Stop hardcopy or incremental plotter output after NFRAME frames	GPSTOP(NFRAME) *
Terminate graphical output (empty storage buffers)	GREND

# APPENDIX B:

## The Standard GHOST Character Sets.

### Character Set 0 : B-L 120 hardware characters

In this table, and the succeeding ones, the index number of a character whose co-ordinates are  $(\alpha, \beta)$  is given by  $10\alpha + \beta$ .

$\alpha \backslash \beta$	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	(space)	A	B	C	D	E	F	G	H	I
2	J	K	L	M	N	O	P	Q	R	S
3	T	U	V	W	X	Y	Z	.	,	$\alpha$
4	$\beta$	?	$\gamma$	+	-	*	/	=	(	)
5	.	o	$\Sigma$	$\pm$	$\pi$	'	"	~	\$	d
6	$\partial$	$\delta$	$\square$	$\int$						



Character Set 1 : Contains Hollerith and Teletype characters

$\alpha \backslash \beta$	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	(space)	A	B	C	D	E	F	G	H	I
2	J	K	L	M	N	O	P	Q	R	S
3	T	U	V	W	X	Y	Z	.	,	;
4	:	?	!	+	-	*	/	=	(	)
5	[	]	<	>	↑	'	"	#	\$	&
6	@	%	←	↘						

Character Set 2 : Lower case Roman characters and miscellaneous mathematical symbols.















$\alpha \backslash \beta$	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	(space)	a	b	c	d	e	f	g	h	i
2	j	k	l	m	n	o	p	q	r	s
3	t	u	v	w	x	y	z	.	,	;
4	:	?	!	+	-	*	/	=	(	)
5	.	o	$\leq$	$\geq$	$\pm$	$\equiv$	$\approx$	$\sim$	$\sqrt{\phantom{x}}$	$\wedge$
6	$\partial$	$\nabla$	$\square$	$\int$						

Character Set 3 : Upper case Greek characters and miscellaneous mathematical symbols

$\alpha \backslash \beta$	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	(space)	A	B	$\Gamma$	$\Delta$	E	$\Phi$	$\Gamma$	H	I
2	$\Theta$	K	$\Lambda$	M	N	O	$\Pi$	$\Xi$	P	$\Sigma$
3	T	Y	$\Psi$	$\Omega$	X	Y	Z	.	,	;
4	:	?	!	+	-	*	/	=	(	)
5	$\Rightarrow$	$\Leftarrow$	U	$\cap$	$\supset$	$\subset$		$\perp$	E	$\Delta$
6	<	>		$\top$						



Character Set 4 : Lower case Greek characters and point plotting symbols

$\beta \backslash \alpha$	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	(space)	$\alpha$	$\beta$	$\gamma$	$\delta$	$\epsilon$	$\phi$	$\gamma$	$\eta$	$\iota$
2	$\theta$	$\kappa$	$\lambda$	$\mu$	$\nu$	$\omicron$	$\pi$	$\xi$	$\rho$	$\sigma$
3	$\tau$	$\upsilon$	$\psi$	$\omega$	$\chi$	$\upsilon$	$\zeta$	.	,	;
4	:	?	!	+	-	*	/	=	(	)
5										
6										

Character Set 5 : Contains Flexowriter case "normal" and KDF9 line printer characters

$\alpha \backslash \beta$	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	(Space)	A	B	C	D	E	F	G	H	I
2	J	K	L	M	N	O	P	Q	R	S
3	T	U	V	W	X	Y	Z	.	,	;
4	:	?	!	+	-	*	/	=	(	)
5	[	]	<	>	↑	'	"	10	£	-
6	→	%								

Character Set 6 : Contains Flexowriter Case "shifted" characters

$\alpha \backslash \beta$	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	(space)	a	b	c	d	e	f	g	h	i
2	j	k	l	m	n	o	p	q	r	s
3	t	u	v	w	x	y	z	.	,	;
4	:	?	!	+	-	*	/	=	(	)
5	[	]	<	>	↑	x	÷	≠	£	—
6	→									



## APPENDIX C

```
C      EXAMPLE OF THE USE OF GHOST. GRAPHS OF BESSEL FUNCTIONS.
C
C      EXTERNAL BESS J0, BESS J1
C
C      SWITCH ON AND SELECT HARDCOPY. ADVANCE TO A NEW FRAME.
C      CALL HRDCPY(1)
C      CALL FRAME
C
C      DEFINE UNITS, MAPPING REGION, LIMITING RECTANGLE, CHARACTER
C      SET AND SIZE.
C      CALL CMS
C      CALL REGION(2.0,14.0,2.0,12.0)
C      CALL LIMITS(0.0,15.0,0.0,15.0)
C      CALL CTRSET(1)
C      CALL CRSIZE(0.5)
C
C      WRITE TITLE AT TOP OF GRAPH
C      CALL PLACE(10,3)
C      CALL TYPECS(26HGRAPHS OF BESSEL FUNCTIONS,26)
C
C      SELECT MATHEMATICAL SPACE AND DEFINE MAPPING REGION
C      CALL MSPACE
C      CALL REGION(0.0,12.0,-0.5,1.0)
C
C      DRAW THE CURVES OF J0(X) AND J1(X)
C      CALL GRAPHF(BESS J0)
C      CALL GRAPHF(BESS J1)
C
C      ANNOTATE THE CURVES, AFTER DEFINING CHARACTER SIZE IN M-SPACE
C      CALL CRSIZE(0.075)
C      CALL PLOTCS(1.0,0.9,1HJ,1)
C      CALL SUFFIX
C      CALL TYPECS(1H0,1)
C      CALL NORMAL
C      CALL PLOTCS(3.3,0.4,1HJ,1)
C      CALL SUFFIX
C      CALL TYPECS(1H1,1)
C      CALL NORMAL
C
C      DRAW BORDER AROUND MAPPING REGION. DRAW AND ANNOTATE THE AXES.
C      CALL BORDER
C      CALL AXES
C
C      TERMINATE GRAPHICAL OUTPUT.
C
C      CALL GREND
C
C      CALL EXIT
C      =====
C      END
```

## GRAPHS OF BESSEL FUNCTIONS

