University of London Computer Centre

DIMFILM

Preliminary User Guide

John C. Gilbert

C O N T E N T S

PART 1 - GENERAL PLOTTING

Chapter 1 - Preamble

DIMFILM was first released at ULCC in May 1973 and the original User Guide published one year later. In its initial form DIMFILM was described as a library of Descriptive Instructions for MicroFILM and was designed for use with Fortran on the CDC computers then operated at ULCC. In that format it enjoyed considerable popularity for ten years. Additionally, it was also implemented on a number of other computers driving different (non-microfilm) graphics devices.

The advent of the Cray and Amdahl computers at ULCC required a major conversion exercise if DIMFILM was to continue. At the same time it seemed sensible to redesign the internal logic of DIMFILM to enable support of any graphic device through the provision of appropriate driver software and to ensure a high degree of machine independence so implementation on different machine ranges would be considerably simplified and - of overriding importance - the user interface would be constant. Thus the decision was taken that the new release of DIMFILM would provide a user interface in accord with the Fortran 77 standard. The major difference that the user of the first release of the new DIMFILM will encounter is in the change of many routine names. The opportunity was taken to not only reduce lengths of names to six characters but to rationalise certain of the names. Further, to ensure sensible device independence certain parameter ranges were changed - primarily to remove limitations imposed by the then operating CalComp 1670 Microfilm Recorder. These changes have been kept to a minimum and most users will have to do little more than edit references to DIMFILM library routines. A later chapter in this guide is devoted to changes of this nature that will be required during conversion of programs. This chapter will also serve as an addendum to the original User Guide until a full descriptive document is available.

The new DIMFILM has been designed for the future incorporation of many new facilities and enhancements. These will be described in this preliminary guide as they become available.

DIMFILM development has always benefited from the suggestions and requests made by users. We look forward to this continuing and would restate here our commitment to consider all written comments etc. forwarded to us. At the same time, this release is made on the same basis as those previously - DIMFILM is a developing system and as such any corrections/improvements will be incorporated at the earliest possible time. There can be no guarantee that all routines will function according to specification and your cooperation in reporting any failings (however seemingly trivial) will be appreciated. The user interface specification as detailed in this guide is intended to be definitive and we shall endeavour to support it. Although DIMFILM is a developing product it is supported in the same manner as other software at ULCC.

Chapter 2 - General Notes

In this User Guide a number of conventions have been adopted either in the interest of clarity or to circumvent limitations of the typeface. These should be borne in mind when reading the text.

In the following, actual Fortran references are not given (CALL being omitted) and, unless explicitly stated otherwise, parameters are typed implicitly in accordance with standard Fortran default (i.e. real, unless symbolic name commences I,J,K,L,M,N when integer). Thus, for example SUB(X,Y,I) would be referenced with the Fortran statement CALL SUB(X,Y,I) with parameters X and Y being real and I integer. Unless stated otherwise all references are as subroutines.

There is frequent reference to ranges of values; in particular, in reference to acceptable parameter values. A standard notation has been adopted, whereby square brackets [and] denote inclusive lower and upper limits respectively, while round parentheses (and) denote exclusive lower and upper limits. Thus, for example, (0.0,1.0] indicates the range greater than zero and less than or equal to unity. Occasionally it has been necessary to use the notation <= for less than or equal and >= for greater than or equal; where underlining would result in ambiguity equivalent notations <= and >= have been used.

Chapter 3 - DIMFILM - an overall view

DIMFILM is a wide-ranging and comprehensive library of subprograms for the generation of graphical output by computer. Through its use of descriptive names and short argument lists, combined with the sensible use of defaults, DIMFILM offers a straightforward method of plotting suited both to the simplest application and the experienced user, for whom many advanced facilities are provided.

The user is able to define, and plot within, his own coordinate space, while facility to clip and blank at the boundaries of specified sub-areas is possible. Available are several levels of error checking, with which programs may be speedily debugged.

Elaborate plotting is possible with minimal experience. Features such as alternate line styles (e.g. dashed or dotted lines), rotation and translation of axes, specification of colour/intensity etc. are included. For textual output, a variety of fonts is included offering a range of type styles and complexity. The user may readily modify his output script to contain underlined, italicised or emphasised passages, while super- and sub-scripts can be simply incorporated for the output of complex mathematical expressions. A variety of geometric functions is included.

Complementary to the range of vector plotting facilities, DIMFILM provides comprehensive device independent raster functions. These may be used in conjunction with all other features including the wide range of supported colour models.

In addition to these facilities for straightforward plotting, DIMFILM is capable of generating other varied types of graphical output. Many of these are primarily aimed toward the scientific plot suitable for publication, thus various axis scaling (e.g. linear or logarithmic) are supported together with polar plotting. The user has full control over frame and axis annotation. Blanking is operable for the inclusion of legends or other such captions. Where discrete data coordinates are given interpolation may optionally be performed. Histograms, month-labelled axes and pie-charts are among the other features. The advanced user is able to uniquely tailor his output style by redefinition of the various defaults. These defaults enable the novice user to immediately produce acceptable output. A number of plots may be contained in a single graph, while the user has full control over the number and positioning of graphs in a single "frame". Graphical, diagrammatic and textual items may be freely intermixed in any plot.

The production of contour plots is possible through DIMFILM.

Other facilities are under review and will be introduced as they are developed. User suggestions/wishes are valued, and all such offerings will be given serious consideration.

This document constitutes a concise guide to using the new DIMFILM. Descriptions of routines are not intended to be definitive but are accurate for basic access to most features. Definitive documentation will be prepared at a later date. It will be found that when new, advanced facilities are available access by certain basic routines involves predefinition of a range of sophisticated options. However, the actions/functions described here will ensure compatibility with the previous release of DIMFILM when used in conjunction with the other basic routines.

PART 1 - GENERAL PLOTTING

Chapter 1 - Basic Plotting

This section introduces the basic concepts of diagrammatic plotting (or drawing), although restricting discussion to calligraphic (that is, line or vector) graphical output. The topic of shaded (that is, area filled or raster) output is deferred to a following dedicated section. This demarcation enables the whole subject of raster plotting (at both basic and refined levels) to be presented in a unified way, while users with interest only in line drawing may skip that section. However, some aspects of basic plotting will be of essential relevance to users with a principal interest in shaded images (in particular, device control, plotting areas and colour models).

1.1 - Plotter Control

1.1.1 - Device initiation

At the simplest level, DIMFILM may be initialised by directly referencing one of the device nomination routines. One of these must be the first DIMFILM reference. Each device has its own characteristics and capabilities. An Appendix is included to describe each device in detail. This contains details of specific device parameters the user may need to know and also, where relevant, will detail any device specific routines that may access functions/features that are not sufficiently widespread as to form a part of the general DIMFILM package. There is also an Appendix with general notes on differences between different classes of device (for example, in how overlapping vectors are generated and their appearance).

D35 selects the Dicomed in 35mm black and white aperture card mode (on unperforated film), with landscape aspect ratio. The default coordinate ranges will be horizontal: 0.,32236. and vertical: 0.,24708., corresponding to a nominal image size of 36.41x27.94 millimetres on film.

D35P selects the Dicomed in 35mm black and white (on unperforated film) in portrait aspect ratio. The default coordinate ranges will be horizontal: 0.,24708. and vertical: 0.,32236., corresponding to a nominal image size of 27.94x36.41 millimetres on film. This mode is particularly suited to documentation applications for traditional page layouts.

D16 selects the Dicomed in 16mm black and white cine mode (on double perforated film), with landscape aspect ratio. The default coordinate ranges will be horizontal: 0.,32767. and vertical: 0.,23920., corresponding to a nominal image size of 10.26x7.49 millimetres on film.

D35C selects the Dicomed in 35mm colour slide mode, with landscape aspect ratio. The default coordinate ranges will be horizontal: 0.,32148. and vertical: 0.,21698., corresponding to a nominal image size of 36.3x24.5 millimetres on film.

D16C selects the Dicomed in 16mm colour cine mode (on double perforated film), with landscape aspect ratio. The default coordinate ranges will be horizontal: 0.,32767. and vertical: 0.,23920., corresponding to a nominal image size of 10.26x7.49 millimetres on film.

Other device types will be considered where there is sufficient user demand (to this end, please advise us of your requirements).

1.1.2 - Control

Generally, a single job will consist of a number of discrete pictures and it is required that each of these be separate and the display surface cleared and readied for the next image.

FRAME is used to achieve this, the effect being appropriate to the type of device. Thus, film will be advanced on a film recorder, paper advanced or a new page readied on a pen plotter, or the screen cleared on a display terminal.

1.1.3 - Termination

To ensure the plot data is flushed/disposed (as relevant to the device/system) each DIMFILM execution must be terminated by

DIMEND which will correctly conclude the current DIMFILM execution.

It is possible to combine more than one DIMFILM execution in a job-step provided each is correctly initiated and terminated.

1.2 - Plotting Areas

DIMFILM is designed to operate in the user's chosen coordinate range, with checks to avoid plotting beyond this area. Each device when used for DIMFILM initiation will cause its default coordinate ranges to be used for plot bounds. Generally, it is more convenient for the user to specify his own ranges. Such user coordinates may be referred to as world coordinate space . Additionally, the user may specify sub-areas to which plotting will be confined (by clipping at the boundaries) or in which plotting will be prohibited (by blanking). All these areas are defined by their ranges in each of the x- and y- directions. Thus, parameters (XLEFT,XRIGHT,YLOW,YUP) define a rectangle (aligned with the frame edges) running from XLEFT to XRIGHT in the x-direction and from YLOW to YUP in the y-direction.

1.2.1 - Overall Bounds

BOUNDS(XLEFT,XRIGHT,YLOW,YUP)

is used to set the user's overall plotting bounds as the rectangle with x-range [XLEFT,XRIGHT] and y-range [YLOW,YUP], thereby defining world coordinates. This area is mapped to the largest rectangle of the same aspect ratio that can be fitted on the device display surface. Clipping will be performed at the boundaries of this area, which may be redefined at any time. (Redefinition may rescale symbol/marker heights and broken line patterns.) The current plot position will be set to the lower left corner of the area, viz.

(XLEFT,YLOW). Any currently active preclipping/blanking areas will be cancelled.

While the user’s plotting is performed within the defined overall bounds it is possible to select sub-areas to which plotting is restricted or in which plotting is suppressed.

1.2.2 - Clipping

```
      PANE(XLEFT,XRIGHT,YLOW,YUP)
```

will define an area to which plotting is restricted, i.e. all plot data will be pre-clipped at the boundaries of this rectangle (defined in user’s overall coordinates as the rectangle with x-range [XLEFT,XRIGHT] and y-range [YLOW,YUP]).

```
      ENPANE
```

cancels the current pre-clip area, which effectively reverts to the overall bounds.

1.2.3 - Blanking

```
      BLANK(XLEFT,XRIGHT,YLOW,YUP)
```

will define an area in which plotting is suppressed, i.e. pre- blanking will be performed at the boundaries of this rectangle (defined in users’s overall coordinates as the rectangle with x-range {XLEFT,XRIGHT} and y-range {YLOW,YUP}).

```
      ENBLNK
```

terminates pre-blanking of the user’s plot data.

1.3 - Categories of Drawing

The components of any plot may be categorised by function. First, there are the line drawing operations through which the user constructs the picture: this may be the specific drawing of each graphic element or the application of functional routines (including graphing and contouring operations). Second, each plot may include a textual content: this may be specific text defined by the user or text produced automatically by, for example, axis valuing in graph plotting. Third, plots may incorporate non-textual annotation: this might be the outlining of boundaries (for example, the current clipping pane may be drawn) or axes and their ticking during graph generation.

The user may wish to visually discriminate between these categories through using lines of different appearance for each class. The visual appearance of any line is governed by both the colour in which it is drawn and the style (being such properties as solid/broken and thickness). For this reason, DIMFILM categorises the drawing functions into three classes or types - referred to as Colour/Style Types (CSTypes). Each CSType function will be drawn with the same line, both in colour and style - these may be set independently by the user. These groupings of line properties

Similarly, DIMFILM supports three latent groups of Colour/Style which are accessed according to the particular plot function. These CSGroups each define a set of line-drawing properties, including colour/intensity and style, values for which may be assigned to each group independently. The three CSGroups are associated with the three CSTypes.

When a drawing operation is performed DIMFILM first determines the CSType of the function and then ensures plotting is performed with the corresponding CSGroup colour and style.

This may be summarised:

CSType 1	-	general line drawing most functional plots
CSType 2	-	all text
CSType 3	-	non-textual annotation (graphs,etc.) most boundary plotting

When plotting, the corresponding CSGroup is utilised (e.g. for text, a CSType 2 operation, CSGroup 2 is used). The user may set different colours and styles for each of the three groups and the relevant one will be used according to the actual type of the plot operation. (It is possible to set all three groups to the same colour/style.) It is possible to reassign the mapping between plot operation and colour/intensity group - a more rigorous description is given in a later section.

1.4 - Basic Line Drawing

All basic plotting is performed in the user's coordinate space, subject to any transformations in effect (see later), having regard to any active pre-clipping/blanking. Lines are drawn at the current colour/intensity and in the current style - that is, CSType 1 as described at 1.3.2.

1.4.1 - Drawing

```
      OFF2(X,Y)
```

is used to reposition the current plot position to (X,Y) without drawing any line.

```
      ON2(X,Y)
```

causes a line to be drawn from the current plot position to (X,Y) at the current colour/intensity and in the current style (i.e. CSType 1).

```
      POINT(X,Y)
```

will cause a point to be plotted at coordinates (X,Y), subject to any clipping/blanking (CSType 1).

1.4.2 - Line Style

The user may select the style in which he wishes lines to be drawn. Style comprises all properties controlling the appearance of the line other than its colour/intensity - i.e. it is the combination attributes such as the continuity and thickness of the line.

It is possible to specify solid (i.e. unbroken or continuous) lines (being the default) or dashed, dotted or dash-dotted lines. A user defined pattern may also be defined (described in a later section).

The following all set the style for all CSGroups:

DASH specifies the style to be dashed lines.

DOT specifies the style to be dotted lines.

DSHDOT specifies the style to be dash-dotted lines.

DSHOFF cancels any broken line pattern; the style reverts to solid (i.e. continuous) lines (the default).

It should be noted that any broken line style is continuous, in the sense that the pattern remains in phase over adjacent drawn line segments (thereby enabling curve approximations to be drawn in broken patterns) and across clipped/blanked regions. A pattern will be restarted following any non-drawing relocation of the current position (e.g. by references to OFF2, BOUNDS, FRAME etc.).

The thickness of lines may be specified for all or any of the CSGroups. The user specifies a scale factor by which the default line on any device is to be scaled in thickness. It should be noted that thickness is device related: not all devices can support the same range of line widths. In particular, the default on many devices will be for the finest line width so a factor less than unity will be ignored in such cases. Equally, most devices can only support a discrete range of line thicknesses between a set minimum and maximum. Each device will select an appropriate line thickness depending on its default and the specified scale factor (which by default is unity) - the Appendix on supported devices should be consulted for details of each device’s capability.

LINSF(FACTOR)

sets the line thickness scale factor for all CSGroupings (1-3) to FACTOR (>= 0.0).

LINSFn(FACTOR) - n = 1,2,3

sets the line thickness scale factor for CSGroup n (1,2 or 3) to be FACTOR (>= 0.0).

In both the foregoing references, the case of FACTOR = 0.0 is treated as a special case whereby the finest line of which any device is capable is selected.

1.4.3 - Point/Spot Style

By default, DIMFILM generates points, or spots, as a software drawn mark. However, most devices have a facility to plot a point directly; such a point is known as a hardware generated point, and the user may specify whether hardware or software points are required.

HWSPOT will cause all subsequent points to be produced, where supported, on the device.

SWSPOT will cause all subsequent points to be generated by DIMFILM software. [This is the default.]

When software points are produced, the basic spot size will be (approximately) one ten-thousandth of the maximum device dimension.

As with lines, the size of the basic spot may be scaled using scale factors, and this may be specified for all or any of the CSGroups. The user specifies a scale factor by which the default spot on any device is to be scaled. Where hardware spots are selected, it should be noted that the size is device related: not all devices can support the same range of spots. In particular, the default on many devices will be for the smallest spot so a factor less than unity will be ignored in such cases. Equally, most devices can only support a discrete range of spot sizes between a set minimumn and maximum. Each device will select an appropriate spot depending on its default and the specified scale factor (which by default is unity) - the Appendix on supported devices should be consulted for details of each device’s capability.

SPOTSF(FACTOR)

sets the spot size scale factor for all CSGroupings (1-3) to FACTOR (>= 0.0).

SPOTSn(FACTOR) - n = 1,2,3

sets the spot size scale factor for CSGroup n (1,2 or 3) to be FACTOR (>= 0.0).

In both the foregoing references, the case of FACTOR = 0.0 is treated as a special case for hardware points whereby the smallest spot of which any device is capable is selected. For software spots, the basic spot is scaled to create a spot of the required size. For factors above unity the spot will be (partially) infilled. Factors below one will result in an appropriately scaled spot; a zero factor, for software spots will result in no discernible spot being plotted.

1.5 - Basic Text

The DIMFILM user has available to him a very sophisticated text plotting "subsystem". Much of the power and flexibility of this will be described in later sections. At the elementary level the user may control the size, positioning and orientation of his text string. However, within the actual text string the user may incorporate certain escape sequences to change the output characteristics (e.g. underlining, italicisation,super- or sub- scripting, type density etc.) or even to switch to or load an alternate font. DIMFILM offers a range of fonts (initially being a modification and extension of the Hershey occidental data); this range will be expanded as time and manpower permits - again your requirements are sought. The mathematical capabilities of the original DIMFILM are now embodied in the single text output routine. Details of these and the other escape sequences will be described later as they become available. For the purposes of basic text output the user should refrain from the inclusion of characters * (star) and $ (dollar) in text strings.

1.5.1 - Text Production

```
      SYMTXT(STRING)
```

will output the character string passed in the parameter STRING, which must be of type CHARACTER. The string will be plotted from the current symbol string position - which by default is the current position (normally located via reference to ON2 or OFF2. The string will be produced at the current symbol height and orientation, and (by default) will be plotted in the normal left to right mode. The current symbol colour/style (the CSGroup currently associated with CSType 2) will be used.

The passing of character strings to SYMTXT (and, indeed, to other DIMFILM routines) may be accomplished in a number of ways. The user may pass the string in a simple CHARACTER variable, or as a reference to one element of a CHARACTER array, or by directly defining the string argument in the reference to DIMFILM. When using variable references it is possible to pass substrings, rather than the whole string stored in the simple variable or array element.

Consider the following FORTRAN statements:

```
--------------------------------------------------------------------------------

                        CHARACTER*26 STRING
                        DATA STRING/'abcdefghijklmnopqrstuvwxyz'/

                        CALL SYMTXT(STRING)      (i)

                        CALL SYMTXT(STRING(4:6)) (ii)

--------------------------------------------------------------------------------
```

The declaration of STRING is of a simple CHARACTER variable of length 26. The first reference to SYMTXT (i) will cause the whole of STRING to be output, i.e. all twenty-six letters of the alphabet will be plotted. The second reference (ii), however, will pass only the substring of STRING consisting of characters 4 to 6 (inclusive) to SYMTXT, and the plot will be def only. {FORTRAN 77 will assume an omitted first substring pointer to indicate the beginning of the string, and an omitted second pointer to indicate the end of the declared string.}

When a CHARACTER array is used the concept is similar; only a single array element will be processed by SYMTXT. In this case:

```
CHARACTER*10 STRING(3)
DATA STRING(1)/'abcdefghij'/
DATA STRING(2)/'qrstuvwxyz'/
DATA STRING(3)/'1234567890'/

CALL SYMTXT(STRING(3))   (iii)

CALL SYMTXT(STRING(1))   (iv)

CALL SYMTXT(STRING(2)(:6)) (v)

CALL SYMTXT(STRING)      (vi)
```

Here a CHARACTER array of three elements has been declared, with each element having a length of ten characters. Reference (iii) to SYMTXT specifically passes the third element of the array STRING and the plot will comprise the digits 1234567890. Reference (iv) is to the first element and the plot will be abcdefghij. The next reference (v) is to the substring consisting of the first (defaulted) to sixth character in the second array element and the plot will be qrstuv. The last of these references (vi) is an unsubscripted reference to the array and by default the first element will be passed to SYMTXT, and the effect will be identical to that achieved by the specific element reference at (iv).

Finally, the user is free to specify a constant string as the argument:

```
CALL SYMTXT('xyz')
```

In this instance the whole of the CHARACTER constant will be passed and the plot will be of xyz.

In general the string will be plotted in the current default font - this topic is dealt with later.

1.5.2 - Numerics

Whilst it is usual for numeric values to be passed to SYMTXT in the character string, there are cases when the user may wish to output the current value of a variable in his computer program. DIMFILM supports this requirement for both REAL and INTEGER simple variables (or single array elements) and permits the user to supply the Fortran FORMAT whereby the output will be controlled.

RNUM(RNUMB,FMT)

will output a REAL value, according to the supplied format, at the current symbol string position (see descriptions pertaining to SYMTXT) at the current symbol height and orientation. This is a CSType 2 function.

RNUMB holds the REAL value (it may be a reference to an array element) to be plotted

FMT is a string containing the Fortran FORMAT by which the value is to be output, and must be of type CHARACTER. The FORMAT must commence with an opening parenthesis and contain a matching closing parenthesis to terminate output. Text within the FORMAT will be plotted as if it were directly passed to SYMTXT. The FORMAT must be a valid Fortran specification, comprising only one output record of not more than 150 characters; the only format specifier should be compatible with type REAL (i.e. E, F, G, etc.) - scale factors etcetera are supported. However, only one variable may be output (that is, output lists of several entities cannot be handled - the user should use continuation strings {see SYMCON/TXTCON later}).

INUM(INUMB,FMT)

will output an INTEGER value, according to the supplied format, at the current symbol string position (see descriptions pertaining to SYMTXT) at the current symbol height and orientation. This is a CSType 2 function.

INUMB holds the INTEGER value (it may be a reference to an array element) to be plotted

FMT is a string containing the Fortran FORMAT by which the value is to be output, and must be of type CHARACTER. The FORMAT must commence with an opening parenthesis and contain a matching closing parenthesis to terminate output. Text within the FORMAT will be plotted as if it were directly passed to SYMTXT. The FORMAT must be a valid Fortran specification, comprising only one output record of not more than 150 characters; the only format specifier should be compatible with type INTEGER (e.g. I). However, only one variable may be output (that is, output lists of several entities cannot be handled - the user should use continuation strings {see SYMCON/TXTCON later}).

The following examples of FORTRAN usage should help to clarify the use of the format parameter.

```
                    CALL RNUM(A,'(F10.3)')

                      will output the current value of A (REAL) as
                      it would be output by the Fortran format
                      specification F10.3

                    IJK = 2*35 + 3
                        .
                        .
                    CALL INUM(IJK,'(''THE VALUE OF IJK IS '',I3)')

                      will output the current value of IJK (INTEGER)
                      according to the Fortran format, giving -

                        THE VALUE OF IJK IS  73
```

In each case the format could have been supplied in a variable of type CHARACTER. Note also, when single quotes are to be used in a character string it is necessary to insert two such quotes - the whole string being commenced and terminated by single quotes.

1.5.3 - Text Size/Orientation

Text output via SYMTXT will be plotted at the previously specified height and orientation. This latter may be either relative to the frame or to the current axes. Once set values hold until redefined.

SYMHT(HEIGHT)

sets the character height for subsequent strings to HEIGHT, in the dimensions of the users overall coordinates. At first entry to DIMFILM the symbol height is set to one-fiftieth of the maximum bounds range.

SYMANG(ANGLE)

specifies that text strings are to be produced at ANGLE (AGroup1, default degrees - see 1.9 for discussion of angular measure) to the reference X-direction (which may correspond to either axis or frame) in the counter-clockwise direction.

RELAX indicates that subsequent text strings will be oriented with respect to the current positive X-axis.

RELFR indicates that subsequent text strings will be oriented with respect to the X-boundary of the frame. (This is the default action.)

1.6 - Markers

It is possible to plot various marker symbols centred at the current position.

```
      MARKER(NMARK)
```

cause the marker number NMARK from the current marker set to be plotted, centred on the current position and utilising the current colour/intensity (i.e. CSType 1 - basic line plotting rather than text). This will be plotted at the current marker height. Within any marker font there is a maximum of 48 symbols, consequently NMARK should be in the range [1,48].

```
      MARKHT(HEIGHT)
```

sets the current marker height to be HEIGHT in units of the user's current overall bounds. At DIMFILM initialisation this will be set to one-fiftieth of the Y-range.

1.7 - Colour/Intensity

To support a diverse range of graphic output devices DIMFILM provides considerable flexibility in the specification of colour and intensity. By adopting a consistent approach to the interrelationship between these parameters output can be directed to monochrome or achromatic devices and to true or pseudo colour devices.

As described elsewhere, DIMFILM supports three latent groups of colour/style which are accessed according to the particular plot function. These CSGroups each define a set of line-drawing properties, including colour/intensity and style. The user may set up different colours/intensities for each of the three groups and the relevant one will be used dependent on the nature of the actual plot operation. Alternatively, the user may set all three groups to the same colour/intensity value.

1.7.1 - Intensity

For any CSGroup, the relation between colour and intensity is maintained. Where a colour is specifically set the corresponding intensity will be computed according to the predefined relationship (covered later). However, where a specification is made for intensity alone, the colour will be recomputed to maintain the same hue whilst yielding the desired intensity. Thus for monochromatic or achromatic use the intensity only need be specified. (It should be noted that due to the methods of colour mapping not all hues utilise the whole range of intensity. Appropriate diagnostics are issued and hue maintained where possible.)

```
      INTSY(ZINT)
```

sets the intensity for all CSGroupings (1-3) to value ZINT, which should be in the range [0.,1.]. (For each device this unit range is mapped onto the available intensities.)

```
      INTSYn(ZINT) - n = 1,2,3
```

sets the intensity for CSGroup n (1,2 or 3) to value ZINT in range [0.,1.].

At DIMFILM initialisation the intensity is set to a default value of 0.8.

1.8 - Colour Models

DIMFILM provides the user with the means to specify colour either through reference to a look up table or specifically by definition of a triplet of colour defining values in any of the range of supported colour models. The methods may be freely intermixed and the user provided with the maximum flexibility in attaining the desired result. Each method is described in the following section on colour selection. The remainder of this section is concerned with a description of the individual colour models. An appendix will later be provided defining the various look up table definitions in terms of selected models; this should serve as a useful guide to the user wishing to specify his own colour palette.

1.8.1 - General Terms

Light may be seen as achromatic or chromatic: achromatic light has no discernable colour - it is a composite of frequencies in the spectrum yielding "white" light, perceived as a shade in the "grey" scale running from black-grey-white; chromatic light has a dominant colour and is (generally) a mixture of frequencies. Monochromatic light is a specific case of chromatic light: it is a single frequency and may be seen as varying brightness of the one colour.

Achromatic and monochromatic light have a single attribute - intensity - by which the observer can differentiate output from different sources. This is the single parameter that is important when outputting to a black and white device or, indeed, a monochromatic device (such as a green screen storage tube). Chromatic light - or colour - has, however a range of sensations experienced by an observer. In order to uniquely define a colour these attributes must be identified and specified. Colour is to a large extent subjective. Indeed, the physiological response to "colour" is still under investigation. However, it is generally accepted that there are three psycho-physiological characteristics of colour that may be quantified, and which together can define the observed colour; these are typically hue (essentially the "colour"), saturation (the "purity"), and brightness ("luminance" or intensity). These will be considered in turn.

hue - hue is used to distinguish the perceived "colour". This could be a single waveband in the visible spectrum, but would more normally be a group of different wavelengths that are emitted and result in the (subjectively) perceived colour. In practice, a dominant (i.e. single) wavelength may be derived which (possibly mixed with pure white) results in the same physical interpretation by an observer. Thus, hue may be defined as light of a single frequency corresponding to the dominant wavelength. For convenience, the spectrum of visible light may be arranged as a circle onto which the hue may be mapped. This angular measure defaults to degrees. A convention has been adopted whereby zero corresponds to red, with hue increasing counterclockwise through yellow. Note that complementary colours (see CMY discussion below) are diametrically opposed; i.e. they lie 180 degrees apart.

```
--------------------------------------------------------------------------------

               This may be portrayed diagrammatically:

                                       *
                                    -     -
                   green  120.0  -           -  60.0  yellow
                               *               *

                             -                   -

                           -                       -

                          -                         -
             cyan  180.0 *        hue spectrum       * 0.0  red
                          -                         -

                           -                       -

                             -                   -

                               *               *
                    blue  240.0  -           -  300.0  magenta
                                    -     -
                                       *

--------------------------------------------------------------------------------
```

It should be noted that white, gray (any shade), and black are not hues but may be considered as colours.

saturation - saturation is the degree to which a colour is undiluted by white (i.e. the absence of an amount of its complementary colour, which would effectively add "white"). More strictly this is the purity of a colour. It is expressed on a normalised scale [0.0,1.0]; for example, pure red is more saturated than pink. It must be noted that it is independent of brightness; at any brightness a hue may attain any saturation in the range [0.0,1.0] (excepting the cases of black and white - nominally corresponding to minimum, or zero, and maximum brightness). In essence, saturation controls the appearance of a colour running from the pastel shades to strong (and pure) colours. [In some colour models saturation is not truly equivalent to purity, as its ratio relates only to the particular colour subspace modelled.]

brightness - brightness (also variously referred to as lightness, intensity etc.) is equivalent to the intensity attribute of achromatic light. It is notionally independent of hue and saturation. Strictly it would be an expression of the luminance of a colour surface. However, the term brightness (alternatively lightness, intensity) has become accepted in the very subjective area of colour definition. This is normally quantified as a normalised value in the range [0.0,1.0].

While discussing the general definition of colour it is worth considering the terms used by artists. They are more concerned with describing colours by reference to the attributes tints , shades , and tones of pure colours (i.e. the available pigments). From a pure pigment (e.g. one that is fully saturated), the artist may choose to add white pigment (thereby creating a tint - and decreasing the saturation), or adding black pigment (creating a shade - decreasing the brightness). Addition of both white and black pigments results in a tone. All these changes are variations of a single hue.

```
--------------------------------------------------------------------------------

                This may be shown diagrammatically:


                                     Tints
                      White   *---------------------* pure colour
                              |                    /   (i.e. fully
                              |                   /     saturated
                              |                  /            hue)
                              |                 /
                              |                /
                              |               /
                              |              /
                              |             /
                              |   Tones    /
                              |           /
                      Grays   |          /  Shades
                              |         /
                              |        /
                              |       /
                              |      /
                              |     /
                              |    /
                              |   /
                              |  /
                              | /
                              |/
                      Black   *



--------------------------------------------------------------------------------
```

1.8.2 - Look Up Tables

DIMFILM contains a 256 entry colour Look Up Table (LUT) that the user may reference. A simple integer value in the range 0-255 will access a colour (internally defined by an appropriate value triplet) from the LUT. The user may redefine individual entries in the table, or the complete table, either through his own specification of triplet(s) or by reference to one of the standard LUTs supported by DIMFILM (reference may be to the complete table or to an individual member). In defining triplets for insertion into the LUT the user may work in any of the colour models supported in DIMFILM. Although the default model is the RGB system, the user may well prefer to work in one of the other models (possibly due to familiarity or the particular colour properties required). Switching between models is possible at any time, but will not affect any colours already set. The use of look-up-tables is dependent on the output device capabilities; the DIMFILM table of 256 may be excessive for many devices and the user should ensure familiarity with his device. For many purposes the user will find need of relatively few colours, but will want this to be discrete and well defined. With this in mind, all the look up tables offered by DIMFILM will (unless specifically stated otherwise) have a common set of colours for the first 16 entries. The remaining entries will be dependent on the nature of the selected model.

For all look up tables these common entries are:

entry	(R ,G ,B)	colour
0		background (device specific)
1		foreground (device specific)
2	(1.,0.,0.)	red
3	(0.,1.,0.)	green
4	(0.,0.,1.)	blue
5	(1.,1.,0.)	yellow
6	(0.,1.,1.)	cyan (blue-green)
7	(1.,0.,1.)	magenta (red-blue)
8	(1.,.5,0.)	red-yellow (orange)
9	(.5,1.,0.)	yellow-green
10	(0.,1.,.5)	green-cyan
11	(0.,.5,1.)	cyan-blue
12	(.5,0.,1.)	blue-magenta
13	(1.,0.,.5)	magenta-red
14	(1.,1.,1.)	white
15	(0.,0.,0.)	black

Note the foreground/background colours are device defaults (refer to Appendix on devices). They are usually black and white, although which is used as foreground and which background will depend on device characteristics. For example, a pen plotter is likely to have white background (paper) and black foreground (pen’s ink). Conversely, a raster display or film recorder may have a black background (blank screen/unexposed film) on which the default drawing action, the foreground colour, will be white. The inclusion of white and black at 14 and 15 does not imply that all devices are capable of accessing such colours. The result of drawing in the background colour is also device specific. Unless a device is capable of doing so, picture elements cannot be erased dynamically. (E.g. an active film recorder will already have exposed the vector to be erased and no action, barring total redrawing of frame, can eliminate that vector. However, on most raster devices the background colour may be used to erase data and restore the background.) The Appendix on device types should clarify this situation.

1.8.3 - RGB Colour Model

The RGB colour model is used in colour television monitors and in most colour raster displays. The Red, Green and Blue primary colours may be used additively to create any other colour (that is, as primaries they cannot be decomposed into any combination of the other two chosen primaries). Note that red, green and blue are commonly referred to as the additive primaries. For the purpose of RGB colour definition, a unit cube defined by three orthogonal axes is used; with the cartesian axes for the red, green and blue components each normalised to be in the range [0.0,1.0]. [This is a subspace of three-dimensional colour space.] Any colour in this space is uniquely defined by the relative amounts of the three primaries - i.e. by the triplet value (red, green, blue). Of particular interest is the principal diagonal - this is the achromatic, or grey, scale - being the set of values with equal components of each of the three primaries; this runs from black (at the origin) to pure white (when red=green=blue=unity).

```
--------------------------------------------------------------------------------

               The RGB colour cube may be shown diagrammatically:


                         Green                           Yellow
                             *---------------------------*
                            /                           /|
                           / |                         / |
                          /                           /  |
                         /   |                       /   |
                        /                           /    |
                       /     |                     /     |
                  Cyan*---------------------------*White |
                      |      |                    |      |
                      |                         . |      |
                      |      |                    |      |
                      |                           |      |
                      |      |              .     |      |
                      |                           |      |
                      |      |                    |      |
                      |                           |      |
                      |      |           .        |      |
                      |                           |      |
                      |      |        .           |      |
                      |                           |      |
                      |      |   .                |      |
                      |                           |      |
                      | Black* -  -  -  -  -  -  -| -  - *Red
                      |                           |     /
                      |    /                      |    /
                      |                           |   /
                      |  /                        |  /
                      |                           | /
                      |/                          |/
                      *---------------------------*
                   Blue                           Magenta

--------------------------------------------------------------------------------
```

{The orientation of the tri-orthogonal axes R, G and B - as depicted here - is chosen for two reasons: (i) the R-G-B axes correspond to the normal three-dimensional cartesian axis convention for X-Y-Z, and (ii) when viewed as a projection along the achromatic diagonal (from white toward black) there is a good visual relationship with the hue disc for which red lies at zero degrees (see Section 1.7.1).}

1.8.4 - CMY Colour Model

Whereas red, green and blue are known as the additive primaries, their complements - cyan, magenta and yellow, respectively - are referred to as the subtractive primaries. This is because - on transmitted or reflected light -

they remove or subtract their complementary colour. Thus:

```
-red   = blue  + green = cyan
-blue  = green + red   = yellow
-green = red   + blue  = magenta
```

Indeed, the colour subspace of the CMY system is the identical subspace in cartesian space as defined by the RGB colour cube. However, an essential difference is that white light is located at the origin, with black at the opposite vertex. The additive primaries add to black, while the subtractives subtract from white light.

The subtractive primaries are of especial importance in photographic emulsions and in print applications, where the pigments in print dyes operate in a subtractive mode. Thus, for example, green is achieved through the combined effects of yellow and cyan inks. Yellow absorbs blue - leaving green and red - while cyan absorbs red, leaving only green to be reflected from the surface. Similarly, cyan, magenta and yellow inks together will reflect no light with a resultant black. The coloured pigment employed in most hard copy devices (e.g. ink jet, thermal ink transfer and electrographic techniques) functions in this manner - CMY colour space is therefore useful in defining these processes.

The conversion between RGB and CMY triplet values
is given by:

$$\begin{vmatrix} C \\ M \\ Y \end{vmatrix} = \begin{vmatrix} 1 \\ 1 \\ 1 \end{vmatrix} - \begin{vmatrix} R \\ G \\ B \end{vmatrix} \qquad \begin{vmatrix} R \\ G \\ B \end{vmatrix} = \begin{vmatrix} 1 \\ 1 \\ 1 \end{vmatrix} - \begin{vmatrix} C \\ M \\ Y \end{vmatrix}$$

```
--------------------------------------------------------------------------------

                The CMY colour cube may be shown diagrammatically:

                        Magenta                           Blue
                              *---------------------------*
                             /                           /|
                            / |                         / |
                           /                           /  |
                          /   |                       /   |
                         /                           /    |
                        /     |                     /     |
                    Red*---------------------------*Black |
                       |      |                    |      |
                       |                         . |      |
                       |      |                    |      |
                       |                           |      |
                       |      |              .     |      |
                       |                           |      |
                       |      |                    |      |
                       |                           |      |
                       |      |           .        |      |
                       |                           |      |
                       |      |        .           |      |
                       |                           |      |
                       |      |   .                |      |
                       |                           |      |
                       | White* -  -  -  -  -  -  -| -  - *Cyan
                       |                           |     /
                       |    /                      |    /
                       |                           |   /
                       |  /                        |  /
                       |                           | /
                       |/                          |/
                       *---------------------------*
                  Yellow                           Green

--------------------------------------------------------------------------------
```

1.8.5 - Broadcast TV Colour Models

Broadcast television has developed colour models to serve its own particular requirements. Indeed, the need for a colour methodology arose after a number of years of monochrome television transmissions. This dictated the overriding concern for compatibility of a colour signal when received by a monochrome receiver. The outcome was a value triplet one member of which was a measure of brightness . For computer graphics this has the fortunate result that by assigning different levels of brightness to different colours one can be sure that they may be discriminated when displayed on a monochrome device. The brightness axis runs along the achromatic scale (suited to the particular display device characteristics); the two remaining axes retain orthogonality by their definition in the plane perpendicular to brightness. The direction of one of these axes must be chosen, thereby leaving the plane containing the remaining axis defined. Essentially, these two axes are together a measure of the chroma component when all brightness (i.e. the "white" component) is removed. As indicated above in the discussion of general terms hue and saturation describe this colour - and, indeed, the two remaining axes may be related to hue and saturation trigonometrically. However, by maintaining all three axes in three dimensional cartesian space simple transformation exists between this space and RGB (as also to CMY). The importance of this should not be forgotten. Further, a straight line in all these spaces transforms into a straight line in each of the others - thus, interpolations between colours may be accomplished simply. [This is not true in general for transforming interpolations between HSV and HLS spaces and RGB, CMY or "television" colour spaces.]

The definition of "television" colour space depends on the chosen brightness axis and the orientation of the remaining axes. The first is selected physically on the basis of what RGB value yields white (not the ideal model of the diagonal in the colour cube). Here there arise differences between adopted national standards.

PAL (Phase Alternation Line) as used in Great Britain, Germany and most non-American Western nations has defined brightness (Y) as:

$$Y = 0.299R + 0.587G + 0.114B$$

The other two components are known as U and V , and are chosen so U is "near-blue" (to the magenta side of blue. V is then at right angles to U (on the magenta side of red).

The conversions between YUV and RGB are then defined as:

$$\begin{vmatrix} R \\ G \\ B \end{vmatrix} = \begin{vmatrix} 1.000 & 0.000 & 0.140 \\ 1.000 & -0.394 & -0.581 \\ 1.000 & 2.020 & 0.000 \end{vmatrix} \begin{vmatrix} Y \\ U \\ V \end{vmatrix}$$

and

$$\begin{vmatrix} Y \\ U \\ V \end{vmatrix} = \begin{vmatrix} 0.299 & 0.587 & 0.114 \\ -0.148 & -0.291 & 0.439 \\ 5.007 & -4.193 & -0.814 \end{vmatrix} \begin{vmatrix} R \\ G \\ B \end{vmatrix}$$

It is instructive to look at the YUV axes against the RGB axes. [Note: for this representation the RGB axes have been rotated from the previously shown orientation in order to better illustrate the YUV axis selection.]

```
                                 | B
                                 |
                                 |
                                 |
                                 |
                                 |
                                 |
                                 |
                                 |
                                -|                  / Y
                          U  -   |  -             /
                         \-      |      -       /
                        - \      |        -   /
             sat.          \     |          /
                -     -    .\    |        /  -
                   -     .   \   |      /
                   -  - . hue \  |    /        -
                         -     \ |  /
                  -         -   \|/              -
                  -            - *-------------------------
                   -           -/                 -       G
                             - /
                     -     -  /                 -
                         -   /
                       -    /                 -
                     -     /
                 V -      /-               -
                         /   -
                        /       -      -
                       /           -
                      /
                     /
                    /
                   / R
```

[Note: hue angularly from U, saturation radially outwards.]

NTSC (National Television System Committee) is the system used in the United States, South America and Japan. The NTSC definition of brightness (Y) is:

$$Y = 0.30R + 0.59G + 0.11B$$

The other two components are known as I and Q, and were chosen with the I axis being aligned with orange-cyan and the Q axis as green-magenta.

The conversions between YIQ and RGB are then defined as:

$$\begin{vmatrix} R \\ G \\ B \end{vmatrix} = \begin{vmatrix} 1.00 & 0.95 & 0.62 \\ 1.00 & -0.28 & -0.64 \\ 1.00 & -1.11 & 1.73 \end{vmatrix} \begin{vmatrix} Y \\ I \\ Q \end{vmatrix}$$

and

$$\begin{vmatrix} Y \\ I \\ Q \end{vmatrix} = \begin{vmatrix} 0.30 & 0.59 & 0.11 \\ 0.60 & -0.28 & -0.32 \\ 0.21 & -0.52 & 0.31 \end{vmatrix} \begin{vmatrix} R \\ G \\ B \end{vmatrix}$$

The differences between the NTSC and PAL models were occasioned by the criteria used in the location of the I-Q and U-V axes. The NTSC system was based on the degree of differentiability between colours of the same brightness for different hues: the eye being particularly perceptive for orange-cyan. PAL, on the other hand, was designed to minimise errors in hue reproduction (a disadvantage of NTSC - "Never Twice the Same Colour" - is its liability to hue distortion, particularly flesh tones) and to simplify the design of decoders.

1.8.6 - HSV Colour Model

The above described colour models are primarily hardware related. Although, for instance, RGB may be used to define colours rationally it does not lend itself to the artists’ concept of colour. Indeed, a perceptual model requires reference to tints, tones and shades rather than the more physically precise RGB rectangular coordinates. Compromise models exist: HSV is one of these. In actual fact, such models are defined in terms related to the display hardware, but in a way that - at least in practice - is sympathetic to the subjective visual system. The HSV model defines the appearance of a colour by a value triplet measuring its Hue, Saturation and (notional) Value.

If one considers the RGB colour cube it is possible to visualise the solid from a viewpoint along the extended achromatic diagonal. From this vantage point a hexagonal projection is seen. (Three planes are visible, while three others are obscured.)

Viewed along the principal diagonal the RGB colour cube
will appear thus:

```
                         green  *------------* yellow
                               /.            /\
                              /             /  \
                             /    .        /    \
                            /             /      \
                           /        .    /        \
                          /             /          \
                         /            ./white       \
                  cyan  *------------- .  .  .  .  . * red
                         \            .\            /
                          \             \          /
                           \        .    \        /
                            \             \      /
                             \    .        \    /
                              \             \  /
                               \.            \/
                          blue  *------------* magenta
```

[Note: Visible edges are shown solid, invisible
       edges dotted. The black vertex is obscured.]

If the RGB colour cube is considered as a projection onto the plane orthogonal to the achromatic axis (i.e. perpendicular to the grey diagonal) a hexagonal disc results. For each value of grey there is defined a subcube with its own projection. As the value is increased from zero (black) the hexagonal disc will increase in size, until it is at a maximum for the cube defined by the full (white) diagonal. [Notes: 1 - at black the disc will be a point, 2 - the projection is scaled so the length of a side in the projected disc equals the length of a side in the solid.] Each disc is the projection of the three "brightest" faces of the associated subcube; where the defining diagonal value (= v) runs from 0.0 to 1.0 the subcube will be comprised of sides of that same length; I.e. each disc comprises colours on the surfaces of the cube defined by r=v, g=v, b=v. So by specifying v at least one of r,g,b will equal v; conversely, v = max(r,g,b). Thus the value is defined by the grey value, in the range [0.0,1.0]. The hue is an angular measure of the hue, or colour component, in degrees - nominally in the range [0.0,360.0]. The convention adopted here is for red to equate to 0.0, with counter clockwise rotation (increasing angle) through yellow, green, etc. The remaining member of the defining triplet is the saturation, which is measured radially outward from the value axis of the hexcone. The saturation is essentially a ratio in that for any hexagonal disc section it may take any value in the range [0.0,1.0]; being zero on the axis and unity on the boundary of the disc (i.e. at the triangular surfaces of the hexcone). Saturation = 0.0 refers to the central axis and so covers the range of greys; in particular, with value = 0.0 yields black and = 1.0 gives white - intermediate values are shades of grey. For the case of zero saturation hue is undefined. At all other values of saturation the hue must be specified.

The relationship to the artists perception of colour may be illustrated by considering a fully saturated hue at maximum value. For example, (0.0,1.0,1.0) [hue = 0.0, saturation = 1.0, value = 1.0] is pure red. The addition of white, thereby creating tints, is accomplished by reducing the saturation (holding the value constant). Equivalently, shades may be created by adding black, and is effected by decreasing the value (the saturation being unchanged). Decreasing both saturation and value together results in tones.

The HSV single hexcone colour model may be shown diagrammatically:

```
                              ^ V = value
                              |
                  green       |       yellow
                       -------|-------
                     - .      |      . -
                  -           |           -
               -              |              -
            -           .     |     .           -
         -                    |                    -
   cyan *                 1.0 * white               * red
         \-              .    |    .              -/
          \  -                                 -  /
           \    -             |             -    /
            \      -      .       .      -      /
             \        ----------------         /
              \   blue \      |      /magenta /
               \          .     .          /
                \       \     |     /       /
                 \                         /
                  \      \  . | .  /      /
                   \                     /
                    \     \   |   /     /
                     \       . .       /
                      \    \  |  /    /
                       \             /
                        \   \ | /   /
                         \         / -
                          \  \|/  /         -
                           \     /      -.
                            \ | /   -      .H = hue
                             \ /-           .
                              *-------------------->
                          0.0   black   S = saturation
```

[Note: obscured edges shown dotted.]

1.8.7 - HLS Colour Model

An alternative to the previously described HSV colour model is the HLS model. The basic difference is the substitution of lightness for value; hue remains as for the HSV model, while saturation is essentially equivalent (though not numerically identical). However, single hexcone HSV space is transformed into a double hexcone for the HLS model. This may be considered to be accomplished by deforming the HSV space with the central white point being drawn upwards. The central axis remains the achromatic range, and this is the axis of lightness - again running from 0.0 (black) to 1.0 (white). It is important to appreciate that in this model maximally saturated hues are located at mid-lightness (saturation = 1.0, lightness = 0.5). Considering the surface of the double hexcone, above the mid-section the surface corresponds to the upper (V = 1.0) hexagonal disc of the HSV model so one of R,G,B will be maximum at 1.0 and lightness above 0.5 will be achieved by desaturating this dominant component. Indeed, at L = 1.0 saturation and hue are irrelevant as pure white results {rgb = (1.0,1.0,1.0)}. While the HLS model has saturation measured radially (in range (0.0,1.0), as before) the numeric value for saturation of an identical colour in the HSV and HLS models will differ.

The achromatic scale occurs when saturation = 0.0, when hue may be undefined, and the degree of grey is determined by the value of lightness.

In neither the HSV nor HLS model does lightness/value equate to luminance (as, for example, used in the broadcast TV models), and so different colours may share the same luminance and therefore be represented by the same grey scale on monochromatic displays.

The HLS double hexcone colour model may be illustrated:

```
                                        ^
                                        |  L = lightness
                                        |
                                    1.0 * white
                                       /|\
                                      // \\
                                     /.|||.\
                                    / |   | \
                                   / ./ | \. \
                                  /  /     \  \
                                 /   |  |  |   \
                                /  . /     \ .  \
                               /    /   |   \    \
                              /     |       |     \
                             /    . |   |   | .    \
                            /       /       \       \
                           /       /    |    \       \
                          /        |         |        \
                         /  green. |    |    | .yellow \
                        /        . / . . . . \ .        \
                       /       . ./     |     \. .       \
                      /     .     |           |     .     \
                     /   .        |     |     |        .   \
                    / .           /           \           . \
                   /             |      |      |             \
             cyan *              |  0.5 *      |              * red
                   \-            / .    |    . \            -/
                    \  -        /               \        -  /
                     \    -     |       |       |     -    /
                      \      -  /   .       .   \  -      /
                       \        -----------------        /
                        \   blue \      |      /magenta /
                         \           .     .           /
                          \       \     |     /       /
                           \                         /
                            \      \  . | .  /      /
                             \                     /
                              \     \   |   /     /
                               \       . .       /
                                \    \  |  /    /
                                 \             /
                                  \   \ | /   /
                                   \         / -
                                    \  \|/  /         -
                                     \     /      -.
                                      \ | /   -      .H = hue
                                       \ /-           .
                                        *-------------------->
                                    0.0   black   S = saturation
```

[Note: Obscured edges shown dotted.]

{The HLS convention adopted by Tektronix uses a double cone rather than a double hexcone. Illustrations of the double hexcone often portray it as a double cone by topologically deforming it. Additionally, Tektronix use the convention of blue being located at zero hue. The algorithms incorporated in DIMFILM for the HSV and HLS models relate to the geometry of the hexagonal disc and require no trigonometric evaluations.}

1.9 - Colour Selection

Whereas the preceding section described the various colour models supported by DIMFILM, the present section is concerned with the specification and selection of colours for the purpose of drawing.

As described previously (1.3.2), DIMFILM supports three latent groups of colour/style which are accessed according to the particular plot function in use. Colours may be specifically assigned to any or all of these CSGroups. This colour specification may be either by direct assignment of the value triplet in any of the supported colour models or by reference to a look-up table (members of which may be set by the user). Direct specification of a colour results in the computation of the corresponding intensity (for monochromatic output); however, subsequent change of the intensity (see 1.6, above) will maintain the hue (in terms of the proportion of the R,G,B components).

Specification of colour etc. is considered for each supported colour model separately. For each model, the triplet value will be checked against range limits for each parameter; out-of-range parameters will result in a diagnostic and the colour assignment will be ignored. Internally, DIMFILM will ensure also that the value triplet transforms to within the RGB colour cube space; the R,G,B values will - if necessary - be forced into range, noting that this may well result in a change of hue.

Some DIMFILM subroutines require specification of the triplet of values defining the colour without specific identification of the colour model to be used. (This is the case with certain raster operations and modification of look-up-table entries.) In these cases the current colour model will be used in interpreting the triplet.

1.9.1 - Colour model

DIMFILM is initialised with the RGB colour model as default. Any value triplets without specific colour model reference will be assumed to be in terms of RGB colour space and interpreted accordingly. The user may wish to redefine the current colour model; indeed, it is likely that one of the other models will be preferable for many applications. All subsequent triplet definitions (other than those specifically related to other models) will be interpreted under the new current model. This model may be changed at will, but is not retrospective. Changing the colour model will not affect the appearance of any previously assigned colour.

```
      CMODEL(<model>)
```

sets the default colour model to be that defined by the parameter,which must be of type character. The parameter may be a character variable or a character string. The permitted values are:

RGB

CMY

HLS

HSV

YUV

YIQ

Thus, for example, a reference to CMODEL('HLS') will cause subsequent undesignated colour triplets to be interpreted according to the HLS model. However, this does not preclude references to routines such as CMY2 (see below) being used.

1.9.2 - Look Up Tables

There are very many situations when the user will prefer to make reference to look up tables rather than making colour selections on the basis of triplets of real values. Where only a finite range of colours is required (and elaborate facilities, such as fades etc. are not used) the look up table provides a simple and reliable method of selection. As described earlier, the default DIMFILM look up tables offer as the first 16 entries a range of basic colours. The size of the LUTs (256) makes them suitable for most purposes, and a range of different LUTs is provided for different applications.

```
COLOR(ILUT)
```

sets the colour for all CSGroupings (1-3) to equate to the colour defined by entry ILUT in the Look Up Table.

```
COLORn(ILUT)
```

sets the colour for CSGroup n (1,2 or 3) to equate to the colour defined by entry ILUT in the Look Up Table.

ILUT must be in the range [0,255].

It should be noted that the colour plotted will be that defined by the look up table entry at the time of issuing the plot instruction. Thus it is permissible to change the look up table value after the LUT assignment for any CSGroup has been made.

The user may wish to modify any (or all) entries in the current look up table.

SETLUT(ILUT,V1,V2,V3)

will set entry ILUT (which must be in the range [0,255]) of the LUT to the colour defined by triplet (V1,V2,V3) in the current colour model.

The user may make such changes at any time; they will affect all subsequent plotting that refers to the LUT. The colour will be evaluated under the colour model current at the time of definition; however, there is no restriction that all entries in a LUT shall be defined in the same model. The user may change the colour model and enter different elements of the LUT in those models.

The user may, at any time, reinitialise the current LUT, either by reloading the default LUT or any of the others available in DIMFILM. Any changes made to the LUT will be lost.

LDLUT0 will reinitialise the active LUT with the DIMFILM default look up table.

1.9.3 - RGB colour model

RGB(R,G,B)

sets the colour for all CSGroupings (1-3) to equate to the colour defined by the value triplet (R,G,B) in the RGB colour model.

RGBn(R,G,B) - n = 1,2,3

sets the colour for CSGroup n (1,2 or 3) to be (R,G,B) in terms of the RGB colour model.

R,G,B are each in the range [0.0,1.0]. {Note: Greys may be achieved by setting R=G=B=v, for v in [0.0,1.0]; with black the special case v=0.0, and white v= 1.0.}

1.9.4 - CMY colour model

CMY(C,M,Y)

sets the colour for all CSGroupings (1-3) to equate to the colour defined by the value triplet (C,M,Y) in the CMY colour model.

```
      CMYn(C,M,Y) - n = 1,2,3
```

sets the colour for CSGroup n (1,2 or 3) to be (C,M,Y) in terms of the CMY colour model.

C,M,Y are each in the range [0.0,1.0]. {Note: Greys may be achieved by setting C=M=Y=v, for v in [0.0,1.0]; with black the special case v=1.0, and white v= 0.0.}

1.9.5 - PAL colour model (YUV)

```
      YUV(Y,U,V)
```

sets the colour for all CSGroupings (1-3) to equate to the colour defined by the value triplet (Y,U,V) in the YUV colour model (i.e. that used in PAL broadcast TV).

```
      YUVn(Y,U,V) - n = 1,2,3
```

sets the colour for CSGroup n (1,2 or 3) to be (Y,U,V) in terms of the YUV (PAL) colour model.

Y is in the range [0.0,1.0]. U is checked against the range [-0.439,0.439], and V against [-5.007,5.007]; noting that this space is a superset of RGB space and so certain combinations of YUV may not map into the RGB unit cube. Care should therefore be exercised to ensure mapping onto RGB is possible. Where the mapping onto RGB space results in a colour beyond the unit cube truncation will be applied to the out-of-range R,G,B component - and hue is not retained.

1.9.6 - NTSC colour model (YIQ)

```
      YIQ(Y,I,Q) - I is real
```

sets the colour for all CSGroupings (1-3) to equate to the colour defined by the value triplet (Y,I,Q) in the YIQ colour model (i.e. that used in NTSC broadcsat TV).

```
      YIQn(Y,I,Q) - n = 1,2,3   I is real
```

sets the colour for CSGroup n (1,2 or 3) to be (Y,I,Q) in terms of the YIQ (NTSC) colour model.

Y is in the range [0.0,1.0]. I (which is a REAL variable) is checked against the range [-0.60,0.60], and Q against [-0.52,0.52]; noting that this space is a superset of RGB space and so certain combinations of YIQ may not map into the RGB unit cube. Care should therefore be exercised to ensure mapping onto RGB is possible. Where the mapping onto RGB space results in a colour beyond the unit cube truncation will be applied to the out-of-range R,G,B component - and hue is not retained.

1.9.7 - HSV colour model

HSV(H,S,V)

sets the colour for all CSGroupings (1-3) to equate to the colour defined by the value triplet (H,S,V) in the HSV colour model.

HSVn(H,S,V) - n = 1,2,3

sets the colour for CSGroup n (1,2 or 3) to be (H,S,V) in terms of the HSV colour model.

H - the hue - is in the range [0.0,360.0) [although it will be forced modulus 360.0], while S and V are in the range [0.0,1.0]. {Note: Greys may be achieved with S = 0.0, and V in [0.0,1.0]; with black the special case V=0.0, and white V= 1.0. For zero saturation hue is undefined.}

1.9.8 - HLS colour model

HLS(H,L,S) - L is real

sets the colour for all CSGroupings (1-3) to equate to the colour defined by the value triplet (H,L,S) in the HLS colour model.

HLSn(H,L,S) - n = 1,2,3 L is real

sets the colour for CSGroup n (1,2 or 3) to be (H,L,S) in terms of the HLS colour model.

H - the hue - is in the range [0.0,360.0) [although it will be forced modulus 360.0], while L (which is a REAL variable) and L are in the range [0.0,1.0]. {Note: Greys may be achieved with S = 0.0, and L in [0.0,1.0]; with black the special case L=0.0, and white L= 1.0. For zero saturation hue is undefined. Fully saturated hues are attained at L = 0.5.}

1.10 - Colour/Style Groups and Types

All plotted/drawn lines may have a number of properties associated with them; these may govern the appearance by specifying colour (and/or intensity), thickness or style (e.g. solid or type of broken pattern).

For the purpose of assigning colour/style attributes, DIMFILM recognises a number of classifications of plot operations. These classes are referred to as colour/style types (CSType). The primary type-class comprises line drawing operations (CSType 1). The secondary type-class (CSType 2) is text output, while the tertiary type-class (CSType 3) includes non-textual annotation of graphical plots. Functional plot routines, with the exception of those for boundary plotting, are performed as primary type operations (i.e. CSType 1). The exceptions (primarily including OUTLIN, OWS, OWSVP, OWSWIN, OWCWIN, MARGIN, OPANE and OBLANK) are plotted under CSType 3. Any other exception will be noted in the routine description.

DIMFILM supports three latent groups of colour/style attributes which are accessed according to the class of the particular plot function. These colour/style groups will be subsequently referenced in the form CSGroup n (n = 1,2,3). In number these CSGroups correspond to the CSType classes. Essentially, each CSType has an associated CSGroup (although the mapping need not be 1:1); i.e. at any time there will be a particular colour/style set of attributes associated with each plot-type classification.

For example: assuming the default mapping, the line produced by a call to GRAPH (see later) is of CSType 1 and so would be drawn in the colour and style currently defined for CSGroup 1; text (such as captions and labels produced, for example, by GRDEF) is of CSType 2 and would be in the colour and style set by CSGroup 2; the axes and framing of the graph (which may also be performed by GRDEF) are of CSType 3 and would be in the colour and style of CSGroup 3.

The assignment of plotting operations into CSTypes is fixed. However, the user is free to define the attributes for each CSGroup (via the colour, intensity and line style routines described elsewhere) and also to assign the mapping of CSGroup to CSType. Initially the mapping is linear - with CSType 1 functions being plotted with CSGroup 1 colour/style attributes, CSType 2 with CSGroup 2, and CSType 3 with CSGroup 3.

The user may set up different colours/styles for each of the three CSGroups and the relevant one will be used dependent on the nature of the actual plot operation (i.e. determined by CSType of operation and mapping onto CSGroup). Alternatively, the user may set all three groups to the same colour/style. By reassigning the mapping between plot operation and colour/style group the user may cause different types of operation to be assigned the same colour/style group

It is important to remember that conceptually there are three latent drawing tools - the relevant one being automatically used for any task (and hence becoming active). Each, independently or collectively, may have different properties specified. DIMFILM will also offer the facility of a resident pool of inactive "pens" from which the latent ones may be assigned.

[After each plotting task is performed the currently active task type reverts to 1, and the currently active colour/style will be the CSGroup associated with CSType 1.]

```
      COLPT1(N)   associates CSGroup N with CSType 1

      COLPT2(N)   associates CSGroup N with CSType 2
```

COLPT3(N) associates CSGroup N with CSType 3

{In the foregoing, N must be in the range [1,3]}

COLSET reset the default association; i.e. CSType n is associated with CSGroup n.

Referring to the earlier example with GRAPH, if it is desired that the axes and boundary annotation have the same appearance as the text, it could be accomplished by a reference to COLPT3(2), which would associate CSType 3 functions (non-textual annotation) with the same group as used by text (CSType 2) - CSGroup 2. The user would then only need to set colour and style for these two groups.

1.11 - Angular Measure

There are various units of angular measure that may be used. There are often different requirements as to which set of units is most appropriate for a particular task, and it may be that the user will wish to use different units for different tasks rather than an overall election of one unit of angular measure.

DIMFILM offers the user three alternate types of angular measure, namely degrees, radians, or grads. Additionally, plot parameters with an angular specification are grouped into four Angular Groups (AGroups), each of which may have an independent specification of unit of angular measure.

The groupings and default units are given in tha accompanying table.

Assignment of angular arguments to groups

AGroup	1	2	3	4
Default	Degrees	Radians	Degrees	Degrees
	SYMANG	TRANGE	CIRARC	PARAL2
	ROTATE	POLAR	DTHETA	BOX
	HATCH	POLFN	ELARC1	POLYGN
	CLABHT	RADSEP		
	SHDEGR	POLTOA		
		POLPY3		
		POLPY5		
		POLFN		
		POLON		
		POLOFF		
		TLEVS		
		ELARC2 - PHI1/2	ELARC2 - PSI	

The relevant angular group is noted alongside each primary parameter description.

The group assignment may be changed, or all groups set to one angular unit. A number of routines exist for these purposes.

RADIAN sets radians as the unit if angular measure for all AGroups.

DEGREE sets degrees as the unit if angular measure for all AGroups.

GRAD sets grads as the unit if angular measure for all AGroups.

RADGRP(N) set radians as the unit of angular measure for AGroupN, for N=1 to 4.

DEGGRP(N) set degrees as the unit of angular measure for AGroupN, for N=1 to 4.

GRAGRP(N) set grads as the unit of angular measure for AGroupN, for N=1 to 4.

Chapter 2 - Refined Plotting

2.1 - Textual Plotting

DIMFILM includes very versatile and comprehensive facilities for the output of text strings. The user has full control over the characteristics of a string and may change these from within the string being plotted. This section gives a detailed description of text control and plotting. The user should note that all switches (i.e. escape sequences) may be incorporated in strings passed to the graphical annotation routines; the only limitation there being the inability to continue text strings and the restriction to string length being imposed by the Fortran implementation used.

2.1.1 - Text Founts

(In accord with common computer parlance, the word font will hereafter be used for fount - referring to a set of "printable" characters.)

DIMFILM, as standard, has several fonts permanently resident in memory for immediate access by the user. These fonts comprise three for alphabets and one each for special symbols and markers. Each alphabet font - numbered 1 to 3 - has a maximum capacity of 96 characters and 24 accents (noting that these latter may be combined), while the symbol font has a maximum capacity of 96 and the marker font a maximum of 48 "characters". The DIMFILM system has access to a considerable number of alternate fonts which may be loaded as the resident fonts, either by a subroutine reference or from within a text string. Switching between resident fonts is a more economical process and is to be preferred where possible.

The fonts are arranged to reflect the printable set of 7-bit ASCII, and are mapped onto this set from the host machine. The user need not concern himself with the mapping mechanism while using character strings. [However, the characters available to a user in a character string within a Fortran 77 program are dependent on two factors: i) the particular Fortran implementation, ii) the input character set supported by the host &/or telecommunications system.] To simplify transport of DIMFILM programs it is possible to access the whole character set through use of the standard Fortran 77 character set.

For reasons that will become apparent, the 96 character font is considered to be arranged in 6 columns (each of 16 rows). The layout is as shown:

DIMFILM Font Layout

	Upper Case				Lower Case	
	0	16	32	48	64	80
1		0	@	P	‘	p
2	!	1	A	Q	a	q
3	"	2	B	R	b	r
4	#	3	C	S	c	s
5	$	4	D	T	d	t
6	%	5	E	U	e	u
7	&	6	F	v	f	v
8	’	7	G	W	g	w
9	(	8	H	X	h	x
10	)	9	I	Y	i	y
11	*	:	J	Z	j	z
12	+	;	K	[	k	{
13	,	<	L	\	l	|
14	-	=	M	]	m	}
15	.	>	N	^	n	~
16	/	?	O	_	o	

Description of non-alphanumerics

1		space	28	;	semicolon
2	!	exclamation point	29	<	less than
3	"	quotation marks	30	=	equals
4	#	number sign	31	>	greater than
5	$	dollar	32	?	question mark
6	%	percent	33	@	commercial at
7	&	ampersand	60	[	opening bracket
8	’	apostrophe	61	\	reverse slant
9	(	opening parenthesis	62	]	closing bracket
10	)	closing parenthesis	63	^	circumflex
11	*	asterisk	64	_	underline
12	+	plus	65	‘	grave accent
13	,	comma	92	{	opening brace
14	-	hyphen (minus)	93	|	vertical line
15	.	period	94	}	closing brace
16	/	slant	95	~	overline (tilde)
27	:	colon			

NOTE:- The Fortran 77 characters set comprises

Alphabetic	A to Z
Numeric	0 to 9
Special	= equal
Characters	+ plus
	- minus
	* asterisk
	/ slash
	(opening parenthesis
	) closing parenthesis
	, comma
	. period (decimal point)
	$ currency symbol (dollar)
	’ apostrophe
	: colon
	" quotation mark
	blank (space)

There is a range of fonts available to the DIMFILM user - and more will be added in the future. Traditionally, fonts are designed for reproduction at a specific size and with a standard appearance. To enable output with the wide range of variation permitted by such a system as DIMFILM compromise has to be made. In expanding a character beyond its designated point size smoothness and solidity may become degraded. According to demand, it is intended to enhance the range of fonts (by way of style and alphabet) - consequently user requirements should be made known. One particular difficulty in font design is the output attributes of mono or proportional spacing; each font may be ideally suited to only one of these modes. The present fonts are best output proportionally spaced (this is the default). It is intended to incorporate a selection of fonts that will be more suited to mono spacing.

Each font is identified to DIMFILM by an integer value. The present set of fonts (with residency denoted in braces) comprises:

Roman (English) Alphabetic Fonts

sans-serif		seriffed	
11	Simplex {1}	14	Small Complex
12	Duplex	15	Complex {2}
111	Mono/Simplex	16	Triplex
		24	Small Complex Italic
		25	Complex Italic
		26	Triplex Italic
		31	Simplex Script
		35	Complex Script
		46	Gothic English Triplex
		56	Gothic German Triplex
		66	Gothic Italian Triplex

Greek Alphabetic Fonts

sans-serif		seriffed	
1011	Simplex	1014	Small Complex
		1015	Complex {3}

Cyrillic Alphabetic Fonts

sans-serif		seriffed	
		2015	Complex

Symbol Fonts

8001	Mathematical (1) {S}
8002	Cartographic
8003	Astronomical
8004	Astrological
8005	Musical

Marker Fonts

9001	Principal Marker Set {M}

Notes

1: "small" in font description indicates a font that is suitable for output at a smaller size; this size is directly controlled by the user. Character sizes for all fonts are set by SYMHT. (Such fonts are generally characterized by the individual character definition employing fewer coordinates; spacing between multiple "strokes" for small, complex characters is also more suited to output at lesser sizes.)

2: Default assignment of fonts to alphabets is indicated, being:

alphabet 1 - font 11,
alphabet 2 - font 15,
alphabet 3 - font 1015,
symbols - font 8001,
markers - font 9001.

The user may load any available font into any of the memory resident set (appropriate action is taken where selected font is incompatible with load destination; i.e. it is possible to load symbols or markers into alphabet fonts, which will not thereby be completely filled, etc.).

Through restricting the number of memory resident fonts that are available to it, DIMFILM seeks to limit its usage of central processor memory. All other fonts are held on a mass-storage device from which they may be acquired (in place of one of the memory resident set) as needed - this being at the users control. This enables DIMFILM to access a very wide range of fonts, a range that may be periodically updated. Pictorially, the process may be shown:

```
--------------------------------------------------------------------------------


                                        Memory
                                 -----------------------------------
                                 |
                                 |   Alphabets
                                 |   -----------
                                .|  1|         |.
                               . |   ----------- .
             Mass Storage     .  |                .  A
             ------------   L.   |   -----------    .        -----
             |          |   O  . |  2|         |.....B- - - >|   |
             |    F     | . A    |   -----------    .        | T |
             |          |. .D    |                 . E       | E |
             |    O     |.       |   ----------- .           | X |
             |          |. .R. . |  3|         |.    T       | T |
             |    N     |.  O    |   -----------             |   |
             |          | . U    |                           | R |===>
             |    T     |.  T    |    Symbols                | O |
             |          | . I  . |   ----------              | U |
             |    S     |  .N    |   |        | - - - - - - >| T |
             |          |   E    |   ----------              | I |
             ------------   S.   |                           | N |
                              .  |    Markers                | E |
                               . |   ---------               | S |
                                .|   |       |- - - - - - - >|   |
                                 |   ---------               -----
                                 |
                                 ----------------------------------


--------------------------------------------------------------------------------
```

LDABET(IBET,NFONT)

will load font number NFONT into memory resident alphabet IBET; where IBET = 1,2, or 3.

LDSYM(NFONT)

will load font number NFONT into memory resident symbols.

LDMARK(NFONT)

will load font number NFONT into memory resident markers.

2.1.2 - Initial Text Characteristics

There are a number of parameters that characterize text appearance other than size and orientation. For example , mono or proportional spacing, colour, density, italicisation. Such entities are, for the purposes of DIMFILM text output, known as the Text Characteristics. In particular, those which are active at the commencement of text form a set known as the Initial Text Characteristics (subsequently referred to as ITC). It is possible to change most characteristics from within a string, and the set active at any time is known as the Current Text Characteristics (CTC). By default, each output string will commence with the initial characteristics active; i.e. the CTC will be set to the ITC. The user may set certain characteristics of the ITC independently. The ITC comprises:

alphabet - the active alphabet from the memory resident set of alphabetic fonts. {Default = 1}

mapping - the mapping from the input string may be set to one of three modes. Mixed identically maps the input character onto the DIMFILM font; Lower maps upper case input characters onto the lower case set (viz. entries in columns 3 and 4 map directly across to columns 5 and 6); Upper maps lower case input characters onto the upper case set (viz. entries in columns 5 and 6 map directly across to columns 3 and 4). (Columns 1 and 2 are unaffected by any mapping. Note also that use of lower case input characters is outside the scope of Standard Fortran 77.) {Default = Mixed}

pen - two "pens" are available for text production (each having an associated colour/intensity). There is a standard colour/intensity for text and an alternate colour/intensity. {Default = Standard}

spacing - text may be produced with either monotonic or proportional spacing. {Default = Proportional}

italics - text may optionally be italicised (i.e. a slant will superimposed on each output character). Note that this characteristic is additional to any inherent in the font definition; if italics are set they will be superimposed on an italicised font. {Default = italics off}

underline - text may optionally be underlined. {Default = underline off}

heavy - text may be in normal density or heavy density. This affects only the intensity of the text (where supported on a device) and should not be confused with bold type which also employs line thickening. (It is possible for the intensity shift to be specified as a reduction when lightening of the density will result.) {Default = normal}

fonts - font assignment to each resident alphabet, special symbol and marker set. {default = see font descriptions} all other - all other text characteristics cannot be set other than from within a string and always revert to their default state at the end of the input string. Such characteristics include superscripts/subscripts/fractions/accents/backspacing/access to symbols and markers etc. Such states are always cleared at commencement/termination of each string passed via SYMTXT etc. regardless of election for text continuation. Each string will commence at the base level (see later discussion of fractions and super-/sub- scripts). {Default = inactive}

2.1.3 - User Specification of ITC

The user may set the various Initial Text Characteristics and certain values associated with them. Once changed they remain in force until reset, either explicitly or implicitly (e.g. by total reset of all text characteristics). The following relate to the setting of ITC values.

ABET(IBET)

selects the memory resident alphabet that is to be used. IBET must be in range [1,3]; default is 1.

UPPER forces input characters to be mapped onto the upper case set.

LOWER forces input characters to be mapped onto the lower case set.

NOMAP accepts input characters without mapping. This is the default.

ALTTYP specifies that the alternate text pen is to be used.

DEFTYP specifies that the default (standard) text pen is to be used. This is the default (and will normally access colour/intensity group 2).

ALTPEN(IPEN)

sets the alternate text pen to group IPEN, which should be in range [1,3]. The default is 3.

MONO selects mono spacing.

PROP selects proportional spacing. This is the default.

ITALIC selects italicisation of text fonts.

NOITAL turns off any italicisation. This is the default.

SETITL(TANANG)

sets the italicisation to the angle with tangent TANANG; the angle being that at which a vertical stroke would appear measured clockwise from the upward normal to the string direction. The default value being 0.2.

UNLIN selects underlining.

NOUNLN turns off underlining. This is the default.

HVYTYP selects heavy type.

NRMTYP selects normal type density. This is the default.

HVYINT(ZINT)

sets the intensity increment to be applied to the normal type intensity as ZINT. (Note: this may be negative, when HVYTYP would result in a lighter type.) The default value is 0.15.

2.1.4 - Current Text Characteristics

There are a number of parameters, in addition to the Initial Text Characteristics, that determine the appearance of text. DIMFILM provides powerful facilities (by way of escape sequences) whereby the user may modify the ITCs within a string and to effect other appearance/layout changes. At any time the set of active parameters governing text appearance is known as the Current Text Characteristics (CTC) - this includes the ITC parameters as a subset, but additionally incorporates a number of dynamic characteristics.

It is reiterated that the subset of the CTC equivalent to the ITC is normally reset identically to the ITC at the commencement of each string, while the remaining parameters are turned off at such time (e.g. fractions are assumed inactive at string commencement, generally being sensibly terminated at the conclusion of each string).

The CTC comprises the ITC (each of which may be dynamically modified) and additionally:

fractions - two levels of fraction beyond the base text level being supported

super/sub scripts

- up to six levels of super &/or sub scripts is supported (beyond the base level) at each fraction level (including the base text level).

Conveniently considered with the CTC are the escape sequences used to access accents, special symbols, and markers, and those to perform other transient effects (e.g. backspace, etc.). Each of these while active may be deemed a CTC; however, their effects are terminated once executed - i.e. they have a strictly limited scope.

2.1.5 - User Specification of CTC

The user may set the various Current Text Characteristics, and certain values associated with them, from within a text string. Once changed they remain in force until reset, either explicitly or implicitly (e.g. by total reset of CTC to ITC at end of string, or through cessation of their effective scope). Within a text string (passed as type CHARACTER) two characters have special significance. These are * (asterisk) and $ (dollar/currency symbol). Generally, these are taken to indicate commencement of an "escape sequence" [following character(s) determining escape function]. To produce either of these characters in text it is necessary to replicate it. Thus, for each single * required in text it is necessary to include the double **; likewise each occurrence of $$ will yield a single output $. All other occurrences of * or $ will either produce a CTC change or result in a diagnostic (possibly an unrecognised escape sequence or one that is invalid at that point). The * and $ have notionally different functions. As a rule, escape sequences flagged by * initiate an effect, while those flagged by $ terminate an effect (certain actions have no corresponding termination; e.g. spacing is specified to be mono or proportional - the * sequence used logically in each case).

The available escape sequences are given here; all are initiated by * and so the given character syntax for the escape should be preceded by *. If termination is applicable this is specifically stated. Only characters in the standard Fortran 77 character set are used for escape sequences (in particular, upper case letters are mandatory). In the following specifications of escape sequences, certain escape fields are "variable" (and should be entered as appropriate by the user) - these are denoted by lower case strings delimited by < and >. {Some examples follow the escape definitions.}

U - upper case mapping. Input lower case characters (i.e. those from columns 5 or 6) are mapped onto the upper case set (i.e. columns 3 or 4, respectively). {Termination by $U causes mixed (or no) mapping to become effective.}

L - lower case mapping. Input upper case characters (i.e. those from columns 3 or 4) are mapped onto the lower case set (i.e. columns 5 or 6, respectively). {Termination by $L causes mixed (or no) mapping to become effective.}

1 - sets the current alphabet to resident alphabet set 1.

2 - sets the current alphabet to resident alphabet set 2.

3 - sets the current alphabet to resident alphabet set 3.

+ - initiates a superscript, incrementing the super/subscript level for the current fraction. {Termination by $+ cancels the last superscript (and any intermediate subscripts) for the current fraction, decrementing the current fraction super/subscript level for each. See also $0 below.}

- - initiates a subscript, incrementing the super/subscript level for the current fraction. {Termination by $- cancels the last subscript (and any intermediate superscripts) for the current fraction, decrementing the current fraction super/subscript level for each.}

A - selects the alternate type (i.e. colour/intensity). {Termination by $A selects the default colour/intensity combination.}

H - selects heavy type. {Termination by $H selects the default intensity type.}

M - selects mono spaced text.

P - selects proportionally spaced text.

= - turns underlining on in the current super/subscript level of the current fraction. {Termination by $= turns underlining off in the current super/subscript level of the current fraction.}

I - turns italicisation on in the current super/subscript level of the current fraction. {Termination by $I turns italicisation off in the current super/subscript level of the current fraction.}

, - initiates a fraction numerator. The fraction level is incremented and the numerator commenced at that level. {Termination by $, terminates the numerator and commences the denominator [this is equivalent to *.].}

. - initiates a fraction denominator for the current fraction level; a numerator must be active at this level. {Termination by $. terminates the denominator and the fraction; the fraction level being decremented. See also $Q below.}

/<res>/<font>/

- loads the font designated by the unsigned integer value <font> into the resident set <res>, which may be any of 1 2 3 S or M, denoting resident alphabets 1 to 3, special symbol set or marker set respectively. The terminating slash is mandatory. E.g. the sequence /3/2015/ would load the complex Cyrillic font (font 2015) into memory resident alphabet 3. Note: the active alphabet is unchanged, and in the example given complex Cyrillic would not become the current alphabet unless resident alphabet 3 was currently active.

The following characteristics have a limited range of definition; that is, they specify an action that is to take place immediately and then revert to the CTC that was previously active (e.g. the escape to output a special symbol will cause output of a single special symbol and then revert to the previously active alphabet).

:<num> - the special symbol having value <num> will be output. The value, denoted by <num>, should be a two digit unsigned integer in the range [1,96]. (A single digit is permissible if unambiguous in context - i.e. it is not followed by a digit. A single digit value may be unambiguously given as two digits, the first being a zero or space.)

::<num> - the marker having value <num> will be output. The value, denoted by <num>, should be a two digit unsigned integer in the range [1,48]. (A single digit is permissible if unambiguous in context - i.e. it is not followed by a digit. A single digit value may be unambiguously given as two digits, the first being a zero or space.)

V<num> - the character in the currently active alphabet having the value <num> will be output. The value, denoted by <num>, should be a two digit unsigned integer in the range [1,96]. (A single digit is permissible if unambiguous in context - i.e. it is not followed by a digit. A single digit value may be unambiguously given as two digits, the first being a zero or space.)

B - the next output character will be backspaced over the previously output character in this string. The backspacing will be performed so that the medians of the characters are coincident.

N - a "null" character is output. This is essentially used to reset the reference point for subsequent super/sub-scripts, in that a super/sub-script is normally placed relative to the current text level so a string may have both super and sub script suffices aligned with continuation text clearing both script levels. The null is necessary if, for example, a prefixed subscript is to follow a super (or sub) scripted entity.

O - subsequent super- and sub- scripts will be taken to clear the maximum and minimum height extents of the previous character. This is particularly useful for limits of integrals, summations etc. in mathematical expressions (when a backspace before the first limit - i.e. "super" or "sub" script will force the limits to be directly above/below the function symbol). For all other purposes the "limit" script will be taken as a level of super/sub scripting. {Termination by $O will cause reversion to normal super/sub script notation.}

D - diacritics (i.e. accents) will be produced. Each DIMFILM font has an associated set of accents that may be applied to any character in that font (for convenience, a cross out character may be included as an accent). There may be up to twenty-four diacritics associated with any font, and these are referred to by the upper case letters A to X. Single, or multiple, diacritics may be associated with any output character. The character to which the diacritic(s) is to be applied must immediately follow the D escape. If a single diacritic is to be produced then its associated reference (A to X) should itself follow the base character. Where more than one diacritic is to be applied the associated reference letters should be enclosed in parentheses following the base character. For example, to output a with an acute accent (reference B) the sequence *DaB would be used. If the user wished to output a crossed out (reference X) c cedilla (reference A) the sequence *Dc(AX) would be used.

(- this (and the next) escape is specifically to enable access of any character in the DIMFILM font layout from the restricted set of characters of standard Fortran 77. The next output character will be derived from the next non-escape input character by first mapping it left one column in the font layout (i.e. its ASCII value will be decreased by 16). {Note: this mapping is applied before any active case mapping (either upper or lower) is applied.} For example, to output < (less than) the sequence *(L could be used.

) - this (and the previous) escape is specifically to enable access of any character in the DIMFILM font layout from the restricted set of characters of standard Fortran 77. The next output character will be derived from the next non-escape input character by first mapping it right one column in the font layout (i.e. its ASCII value will be increased by 16). {Note: this mapping is applied before any active case mapping (either upper or lower) is applied.} For example, to output \ (reverse slant) the sequence *)L could be used; however, if lower mapping were active the output character which would result is | (vertical line).

The remaining escapes are different in that they terminate effects only. That is, they must be preceded by $.

0 - (zero) is used as $0 to clear all super/sub-scripts to the base level in the current fraction. (Termination of a fraction also has this effect.)

Q - used as $0 to quit all super/sub-scripts and fractions to the base text level. (Termination of a character string input also has this effect.)

The concepts embodied in the above discussion will become clear if the following examples are considered.

```
CALL SYMTXT('This is DIMFILM')
           or
CALL SYMTXT('T*LHIS IS$L DIMFILM')
           or
CALL SYMTXT('*Ut$Uhis is *Udimfilm')

will all plot:  This is DIMFILM

CALL SYMTXT('This is *=underlined *Iitalic$I$= text')

will plot: This is underlined italic text
    (with the word italic italicised)

CALL SYMTXT('z = *,x*+2$+ + 1*.*,1*.2y$.$.')

will plot the equation z equals x-squared plus one
all divided by one upon two y.
```

Certain parameters associated with text layouts may be specified by the user. For example, size and positioning of super and sub scripts may be set to "personalise" the output appearance.

SETCEX(CEXP)

- will set the character expansion factor to CEXP (CEXP > 0.0). Each character will be stretched/squeezed in its width relative to the nominal height. {Default = 1.0}

SETCSP(CSPACE)

- each character will be followed by a space of CSPACE times the current text height. A negative value may be used to close up text (caution should be exercised). {Default = 0.0}

SETFR(FHT,FOFF,FGAP)

- controls the appearance of fractions. Heights of each numerator and denominator will be set to FHT times current text height, while the fraction bar will be set at a position FOFF times current text height above the present text baseline. A gap will be forced between the fraction bar and both numerator base and maximum height extent of denominator; this gap will be FGAP times the fraction numerator/ denominator height. (FHT,FGAP > 0.0) {Default = (0.6,3./7.,0.13)}

SETSUP(SHT,SUPY)

- sets the height of superscripts as SHT (> 0.0) times the current text height at a position SUPY times the current text height above the height of the preceding character (either relative to the actual maximum height of that character or as an absolute position according to the current height - see SUPABS/SUPREL). A negative SUPY will position the superscript below the preceding character relative/absolute height. {Default = (0.4,0.1) relative}

SETSUB(SHT,SUBY)

- sets the height of subscripts as SHT (>0.0) times the current text height at a position SUBY times the current text height above the text baseline at the current level. A negative SUBY will position the subscript below the text baseline. {Default = (0.4,0.0)}

SETOVR(OVHT,OVY,UNY)

- when super/sub-scripts are set as over/under (via escape *O) their height will be OVHT times the current text height, while "over" items will have a gap of OVY times current text height between their base and the reference character, and "under" items will have a gap of UNY times the current text height between their nominal top and base of reference character. (OVHT > 0.0; OVY,UNY not < 0.0) {Default = (0.4,0.1,0.1)}

SETUN(UNGAP)

- underlining will be performed at UNGAP times the current text height below the lowest extent of the underlined section of text (this will clear fractions, subscripts etc., but may not clear individual character descenders). (UNGAP not < 0.0) {Default = 0.1}

SUPABS - superscripts will be positioned relative to the absolute (i.e. currently active) height of the preceding character.

SUPREL - superscripts will be positioned relative to the actual character height maximum of the preceding character (although it will be forced to clear any subscript of a single level applied to the same character). Thus superscripts will, for example, be differently positioned for "x" and "X". {Default = relative} TEMPORARY NOTE - superscripts are currently set as absolute at -0.4 (see SETSUP); the declared default state will be applied shortly.

The default values may cause fractions and superscripts to exceed the basic text extent (i.e. baseline to nominal height). Judicious redefinition of certain values can avoid this (provided over/under items are avoided); it is suggested that FHT=0.4, FGAP=0.1, FOFF=0.5, may help in this regard. However, DIMFILM is designed to accommodate variations in line spacing and the "length" text routines may be used in this respect. Full allowance for such text layout will be made by graph annotation routines, etc.

RESTXT - will reset all text string parameters to their default state (except height and string position).

2.1.6 - String Continuation

It is often the case that the user may require a text string to be output positioned immediately following the last output string. This can be very useful when values (output via INUM/RNUM) are required within a complex string.

SYMCON will result in all following direct text output to be positioned as a continuation of the previously plotted string.Note that all subsequent strings will be contiguous until this option is negated.

ENSYMC will cause all following direct text output to commence at the current plot position.

Each string, including those produced as positionally continuous through the use of SYMCON, will (by default) commence with the set Initial Text Characteristics. For blocks of text, whether or not these be positionally continuous, it is often appropriate for each string to continue with the Current Text Characteristics that applied at the conclusion of the previous string.

TXTCON will result in all subsequent direct text output commencing with the CTC that was active at the end of the previous string. I.e. each string does not revert to the ITC at its commencement. Thus TXTCON should be referenced before the first continuation string (and after the string which is required to commence with the ITC). It should also be noted that any reference to any of the ITC routines will force the next string reference to commence with all ITCs active, after which subsequent strings will have continuous CTCs. (Any reference to other DIMFILM modules, such as graphing or contouring, will force the next direct string reference to commence with the ITC - TEXTCON will, however, remain active for subsequent references.)

ENTXTC will cancel the effect of TXTCON; all strings will be plotted commencing with the ITC.

The foregoing is applicable to direct string references (i.e. SYMTXT, RNUM, INUM etc.) - indirect string output via (for example) graph titling, axis labelling is not affected.

It is reiterated that SYMCON and TXTCON are independent, but may be simultaneously active if required.

2.1.7 - Text String Length

It is often necessary, for layout purposes, for the actual output length of a text string to be known (before it is physically plotted).

STRING(TEXT)

is a REAL FUNCTION, returning as its value the length (in users world coordinate units) that would be occupied were the text string in TEXT (of type CHARACTER) to be passed to SYMTXT and plotted at unit height.

SRNUM(RNUMB,FMT)

is a REAL FUNCTION, returning as its value the length (in users world coordinate units) that would be occupied were the parameters RNUMB (of type REAL) and FMT (of type CHARACTER) to be passed to RNUM and the resultant string plotted at unit height.

SINUM(INUMB,FMT)

is a REAL FUNCTION, returning as its value the length (in users world coordinate units) that would be occupied were the parameters INUMB (of type INTEGER) and FMT (of type CHARACTER) to be passed to INUM and the resultant string plotted at unit height.

As an example, suppose the user requires to exactly fit a string into 17.5 of his units. He may pass the string to STRING and obtain the length it would occupy at unit height and select a height accordingly:

```
--------------------------------------------------------------------------------


                      WIDTH = STRING('This occupies 17.5 units')
                      CALL SYMHT(17.5/WIDTH)
                      CALL SYMTXT('This occupies 17.5 units')


--------------------------------------------------------------------------------
```

There is also the case where the user's string may conclude with one or more blanks (possibly when a variable of fixed length is used for several strings input during program execution). In this case:

STRAIL(TEXT,NA,SLONG)

is a subroutine reference, returning in SLONG (REAL) the length (in users world coordinate units) that would be occupied were the text string in TEXT (of type CHARACTER) to have trailing blanks removed and the remainder passed to SYMTXT and plotted at unit height. The number of characters in TEXT, after trailing blanks are stripped, is returned in NA (INTEGER).

All the string length functions/subroutine take full account of embedded escape sequences within the input strings.

DIMFILM offers the user a wide range of raster plotting facilities. These may be freely intermixed with all other plotting operations. However, not all devices can support the (full) range of raster functions. Where possible non-raster devices will perform some simulation of the raster commands, but - depending on the type of device - this may be minimal. The Appendices on device types and specific devices supported should be consulted for details on how the raster operations will be performed. This section details the raster operations as they would function on a full-feature device.

In calligraphic (or vector) plotting, images are created through the drawing of line segments. This is analogous to the draughtsman’s pencil. Raster plotting produces fully shaded (possibly multi-coloured) pictures, and is more akin to the artist’s paintbrush. Digital computers and electronic displays require an ordered approach to picture formation, and so the raster image is composed of a mesh of points. These are generally arranged at regular intervals on a rectangular grid. The smallest point that may be displayed, and to which a unique colour may be assigned, is termed a pixel. These points are so arranged that under normal viewing conditions they cannot be discriminated (within an area of constant colour); that is, they appear to touch and the coloured area is visually continuous. On sophisticated devices the number of pixels is large and they cannot be discriminated even across colour transitions - the image then appears continuous and, given sufficient colour range, will be capable of a high degree of realism. Not all applications of raster graphics require realism. The shading of areas can greatly enhance graphic presentations even when relatively few colours are used.

Different devices may have different numbers of pixels in each orthogonal direction. The usual expression of the raster capability is as ’resolution of H x V pixels’ (in the Horizontal and Vertical directions, sometimes extended by ’x B’ to define number of data, i.e. colour, bits per pixel). An alternative is to express the number of pixels per linear unit of measure (or even per unit of area). The consequence is that to drive a raster device the user (or software) must know the device resolution. DIMFILM is a general purpose graphics library and, as such, is independent (in user terms) of the graphics output device addressed. In consequence, the user defines a raster image in terms of minimum elements (known as cells) and the low level interfaces (the device drivers) convert these into device pixels. There is no need, then, for the general user to be concerned with the device raster specification; beyond the obvious necessity of ensuring the chosen device can adequately represent the generated pictures (in terms of raster resolution, colour representation, etc.). The sophisticated, or more demanding, user will find that careful definition of his coordinate space and his cell mapping will enable precise addressing of a particular device’s raster pixels (and still enabling an approximate depiction on secondary devices - possibly lower resolution preview displays).

In practice, the user defines a cell map, comprising his chosen raster (i.e. cell) resolutions in both horizontal and vertical directions and a mapping of this rectangular area onto his coordinate space. He may then assign a different colour to each cell of this raster grid,thereby forming a cell array . DIMFILM will map this cell array onto each device such that every device pixel contained within any user cell will be assigned the colour of that cell. Clearly, the general case will be that many device pixels are located within a particular cell and so are coloured identically yielding a cell of the designated colour.

```
--------------------------------------------------------------------------------

              This may be illustrated thus:

                    .    .    .    .    .    .    .    .    .    .
                       -------------------------------------
                    .  | *   |*    *    *|   * |  * | *   |.    .
                       |     |     |     |     |     |     |
                    .  | *   |*    *    *|   * |  * | *   |.    .
                       |     |     |     |     |     |     |
                    .  --*----*----*----*----*----*----*----.    .
                       |     |     |     |     |     |     |
                    .  | *   |*    *    *|   * |  * | *   |.    .
                       |     |     |     |     |     |     |
                    .  | *   |*    *    *|   * |  * | *   |.    .
                       -------------------------------------
                    .  | *   |*    *    *|   * |  * | *   |.    .
                       |     |     |     |     |     |     |
                    .  | *   |*    *    *|   * |  * | *   |.    .
                       |     |     |     |     |     |     |
                    .  --*----*----*----*----*----*----*----.    .
                       |     |     |     |     |     |     |
                    .  | *   |*    *    *|   * |  * | *   |.    .
                       |     |     |     |     |     |     |
                    .  | *   |*    *    *|   * |  * | *   |.    .
                       -------------------------------------
                    .    .    .    .    .    .    .    .    .    .

              Notes: i) the cells are shown boxed
                    ii) device pixels are shown as . or *
                   iii) pixels contained in cells = *
                    iv) pixels outside cell map = .


--------------------------------------------------------------------------------
```

The foregoing illustration gives some pictorial idea of how device pixels may be mapped into cells (assignment of cell boundaries to interiors is subject to an implementation convention).

It is (generally) redundant for the user cell resolution to exceed that of the chosen output device for then not every cell can be represented (in this case each pixel will correspond to the one cell containing it, and some cells will be omitted as they will not contain any pixel). Direct device pixel addressing will be ensured when the cell resolution equates to the pixel resolution when the coordinate space is mapped onto the device.

The user of raster plotting through DIMFILM must determine the resolution at which he wishes to work (the cell resolution) and may then communicate with DIMFILM routines in terms of these cells. This may either be defined in terms of the whole cell array or by defining successive single rows of this array - cell scans. Considerations of computer storage may well dictate the chosen mode, as may the algorithm used to generate colour values for each cell.

Whichever mode is utilised, the user may assign colour (or intensity) either by reference to look up tables or specific value triplets. Where scans are utilised there is the option of providing data either for each cell in the scan or for incorporating some compression in the input.

3.1 - Cell Mapping

As depicted above, a cell map consists of a rectangular grid of cells which is mapped onto user coordinate space (i.e. world coordinates). The definition of this cell mapping is common to all subsequent discussion of cell-raster output. The user is required to define the Number of Cells in the X-direction (i.e. horizontally) and the Number of Cells in the Y-direction (i.e. vertically). This gives the cell resolution. Also required is the mapping of this onto the user’s coordinate space. For this it is necessary to define the X-coordinate of the START of the first cell scan and the X-coordinate of the END of that scan, and also the Y-coordinate of the START of the first scan and the Y-coordinate of the END of the last scan. Pictorially this may be shown:

```
--------------------------------------------------------------------------------



                   XSTART                              XEND
                YSTART -----------------..-----------------
                       |    |    |    |    |    |    |    |
                       |  1 |  2 |  3 |    |    |    |NCX |
                       |    |    |    |    |    |    |    |
                       -----------------..-----------------
                       |    |    |    |    |    |    |    |
                       |  2 |    |    |    |    |    |    |
                       |    |    |    |    |    |    |    |
                       -----------------..-----------------
                       |    |    |    |    |    |    |    |
                       |  3 |    |    |    |    |    |    |
                       |    |    |    |    |    |    |    |
                       -----------------..-----------------
                       .    .    .    .    .    .    .    .
                       .    .    .    .    .    .    .    .
                       .    .    .    .    .    .    .    .
                       -----------------..-----------------
                       |    |    |    |    |    |    |    |
                       |    |    |    |    |    |    |    |
                       |    |    |    |    |    |    |    |
                       -----------------..-----------------
                       |    |    |    |    |    |    |    |
                       |    |    |    |    |    |    |    |
                       |    |    |    |    |    |    |    |
                       -----------------..-----------------
                       |    |    |    |    |    |    |    |
                       |    |    |    |    |    |    |    |
                       |NCY |    |    |    |    |    |    |
                  YEND -----------------..-----------------



--------------------------------------------------------------------------------
```

This shows the layout of the cell mapped area. Thus each scan comprises NCX cells, and NCY rows of raster cell scans map onto the area defined by XSTART to XEND in the X-direction (i.e. the ’horizontal’ scan direction) and YSTART to YEND orthogonally in the Y-direction (i.e. the vertical, or direction in which successive scans are produced). It is this area that is mapped from the users world coordinates onto each device display area. This concept is important to all following discussion in this chapter.

3.2 - Cell Arrays

In the simplest form, the user may pass data relating to the whole of the cell mapped area in a two-dimensional data array. In this case the user will have determined the cell values (i.e. colours) for each cell in the NCX x NCY cell grid. In general terms this will be in an array of dimension (NCX,NCY). That is, the value in element (i,j) of this array will be that of the i-th cell in the j-th scan. With cell arrays, the user defines the cell mapped area and passes data for the complete grid of cells.

```
--------------------------------------------------------------------------------

              Pictorially, this may be shown:

                 (1,1)     (2,1)  . . . . . (NCX-1,1)     (NCX,1)

                 (1,2)     (2,2)  . . . . . (NCX-1,2)     (NCX,2)

                 (1,3)     (2,3)  . . . . . (NCX-1,3)     (NCX,3)

                   .         .                   .            .
                   .         .                   .            .
                   .         .                   .            .
                   .         .                   .            .

                 (1,NCY-1) (2,NCY-1)  . . . (NCX-1,NCY-1) (NCX,NCY)

                 (1,NCY)   (2,NCY)  . . . . (NCX-1,NCY)   (NCX,NCY)


--------------------------------------------------------------------------------
```

The user may specify these values as either look-up-table references or actual value triplets (in the current colour model).

3.2.1 - Cell Arrays via LUT

One of the simplest ways to generate a raster is through the definition of a set of CEll Look up table references in an ARray. In this case the cell mapped area must be defined and an array of the LUT values provided.

```
      CELAR(XSTART,XEND,YSTART,YEND,NCX,NCY,LUTARR)
```

will result in the cell array defined in LUTARR being plotted on the cell mapped area using look up table entries.

XSTART is the X-coordinate of the start of the first scan

XEND is the X-coordinate of the end of the first scan

YSTART is the Y-coordinate of the start of the first scan

YEND is the Y-coordinate of the last scan

(all coordinates being in the user’s world coordinate space)

NCX is the number of cells in each scan

NCY is the number of scans

LUTARR is an integer array of dimension (NCX,NCY) with the value of the (i,j)th element being a pointer into the look up table for the i-th cell in the j-th scan. The values in this array should be constrained to [0,255].

There may be occasions when the storage array used for the LUT pointers may not be dimensioned as required (for instance, when only as subset of the storage array is to be plotted). There is, for such purposes, an alternative subroutine CELAR1, which requires additional information concerning the array and subarray dimensions. (The need and use of this will be apparent only to those familiar with Fortran array usage.)

```
CELAR1(XSTART,XEND,YSTART,YEND,NAX,NAY,NCS,NRS,NCX,NCY,LUTARR)
```

will result in the cell array defined in LUTARR being plotted on the cell mapped area using look up table entries. The parameters XSTART, XEND, YSTART, YEND, NCX, and NCY are as for CELAR, above. However, the dimensioning of LUTARR is different, and defined by other parameters:

XSTART is the X-coordinate of the start of the first scan

XEND is the X-coordinate of the end of the first scan

YSTART is the Y-coordinate of the start of the first scan

YEND is the Y-coordinate of the last scan

(all coordinates being in the user’s world coordinate space)

NAX is the first dimension of storage array LUTARR

NAY is the second dimension of storage array LUTARR

NCS is the starting column within LUTARR at which the cell array commences

NRS is the starting row within LUTARR at which the cell array commences

NCX is the number of cells in each scan

NCY is the number of scans

LUTARR is an integer array of dimensions (NAX,NAY) which holds the pointers to the look up table for the NCX x NCY cell array as a subset. Each scan commences at column NCS of LUTARR and the first scan is at row NRS of LUTARR. {For this case, the value of the (i,j)th element of the cell mapped array to be plotted will be found at (NCS-1+i,NRS-1+j) of LUTARR. CELAR is the specific case of CELAR1 with NAX = NCX, NAY = NCY, and NCS = NRS = 1.}

This may be shown schematically as:

```
--------------------------------------------------------------------------------


                               1   NCS                      NAX
                            ------------..-------------..-------
                            |    |    |    |    |    |    |    |
                          1 |    |    |    |    |    |    |    |
                            |    |    |    |    |    |    |    |
                            ------------..-------------..-------
                            |    |    |    |    |    |    |    |
                            |    |    |    |    |    |    |    |
                            |    |  1 |    | NCX|    |    |    |
                            -----=======..=======------..-------
                            |    I    |    |    I    |    |    |
                        NRS |   1I ** |    |    I    |    |    |
                            |    I    |    |    I    |    |    |
                            -----I------..------I------..-------
                            .    .    .    .    .    .    .    .
                            .    .    .    .    .    .    .    .
                            .    .    .    .    .    .    .    .
                            ------------..-------------..-------
                            |    I    |    |    I    |    |    |
                            | NCYI    |    |    I    |    |    |
                            |    I    |    |    I    |    |    |
                            -----=======..=======------..-------
                            |    |    |    |    |    |    |    |
                            |    |    |    |    |    |    |    |
                            |    |    |    |    |    |    |    |
                            ------------..-------------..-------
                            .    .    .    .    .    .    .    .
                            .    .    .    .    .    .    .    .
                            .    .    .    .    .    .    .    .
                            ------------..-------------..-------
                            |    |    |    |    |    |    |    |
                        NAY |    |    |    |    |    |    |    |
                            |    |    |    |    |    |    |    |
                            ------------..-------------..-------


--------------------------------------------------------------------------------
```

The cell mapped array is the subarray of dimension NCXxNCY, with its first element (**) located at (NCS,NRS) within the main storage array of size (NAX,NAY).

3.2.2 - Cell Arrays via Triplet

Not all usage will be satisfied by look up table references. Where more control (and possibly greater colour range) is needed it is necessary to generate CEll Triplet values by ARray. The cell mapped area must be defined, and three arrays must be provided which together define the three colour values for each cell. The meaning of the three arrays depends on which colour model is currently active (e.g. RGB, HLS, HSV, etc.).

```
CETAR(XSTART,XEND,YSTART,YEND,NCX,NCY,V1,V2,V3)
```

will result in the cell array defined by (V1,V2,V3) being plotted on the cell mapped area using direct triplet values interpreted according to the currently active colour model.

XSTART is the X-coordinate of the start of the first scan

XEND is the X-coordinate of the end of the first scan

YSTART is the Y-coordinate of the start of the first scan

YEND is the Y-coordinate of the last scan

(all coordinates being in the user’s world coordinate space)

NCX is the number of cells in each scan

NCY is the number of scans

V1,V2,V3 are real arrays, each of dimension (NCX,NCY), defining the colour triplet value for each cell; the i-th cell in the j-th scan having triplet value {V1(i,j),V2(i,j),V3(i,j)}. The values must be within the ranges required for the currently active colour model which will be used for their interpretation.

3.3 - Cell Scans

The alternative to the cell array method of plotting a raster/cell image is for each scan line of cells to be passed separately to DIMFILM. This may well be the preferred method when large numbers of cells and scans are involved (especially when computer memory is limited).

In this case the user must first define his cell mapped area (there is no default). Each scan line is then passed to DIMFILM, which will interpret successive scan lines as part of a cell mapped image and ensure correct location on the cell mapped area. The first line will be positioned at (XSTART,YSTART) and subsequent lines advanced automatically toward YEND.

A pointer onto the cell mapped area is maintained by DIMFILM to position the next scan. This will be initialised to the cell mapped area X/Y-START at any DIMFILM frame advance, whenever the cell mapped area is redefined, and when a new coordinate transformation becomes active, or when specifically initialised (via RESCMP). The pointer will not be initialised if the actual number of scan lines generated exceeds the predefined cell map limit (in this case, excess lines are ignored). It is permissible for other DIMFILM functions to be performed between consecutive cell scans; the raster image should appear continuous in the cell mapped area (although some devices may show some minor positional discontinuity or colour shift). Equally, it is perfectly acceptable for different scans to be generated by different cell scan techniques (e.g. run length compressed scans may be interspersed with full lines).

3.3.1 - Cell Mapped Area

Prior to any cell scan reference a valid cell mapped area must be defined. Once defined, within a DIMFILM execution, this cell mapped area will remain defined for all subsequent cell scans. It is independent of the cell mapped area as specifically defined via any cell array reference, and so the various types of cell scan operation may be freely intermixed.

```
      CELMAP(XSTART,XEND,YSTART,YEND,NCX,NCY)
```

will define a cell mapped area for use with all subsequent cell scan operations and initialise the cell scan pointer to (XSTART,YSTART).

XSTART will be the X-coordinate of the start of the first scan

XEND will be the X-coordinate of the end of the first scan

YSTART will be the Y-coordinate of the start of the first scan

YEND will be the Y-coordinate of the last scan

(all coordinates being in the user’s world coordinate space)

NCX will be the number of cells in each (full) scan

NCY will be the number of scans for the complete image

It is of note that individual scan lines will be processed up to a maximum of NCX cells, while short scans are permitted.

RESCMP may be used to re-initialise the cell scan line pointer into the mapped area (to X/Y-START). When this is referenced the following cell scans will overlay any previously generated ones on the same mapped area on the current frame. It is included for those cases where such overlays are required. The user is cautioned that the effects on the specific device in use should be considered with care. (An implicit RESCMP is performed on any DIMFILM frame advance.)

3.3.2 - Cell Scans via LUT

Two routines are provided for CEll Look up table reference SCans, with the option of Run length encoding (compression). The routines should be referenced once for each scan line required on the cell mapped area.

```
      CELSC(LSCAN,NC)
```

will cause the NC look up table values in integer array LSCAN (dimension NC) to be applied as the values for NC cells in the next cell scan. These values should be in the range [0,255].

```
      CELSCR(LRSCAN,NC)
```

is for run length encoded scans. The integer array LRSCAN with dimension (2,NC) holds the definition for the next cell scan. For each pair of elements, LRSCAN(1,i) is the number of consecutive cells to have look up table value LRSCAN(2,i); i=1,NC. Thus the length of the scan defined by this reference is the summation of LRSCAN(1,i) for i=1 to NC. (Negative or zero runs are ignored.) Each LUT value should be in the range [0,255].

3.3.3 - Cell Scans via Triplet

As with cell arrays, it is also possible for the user to produce CElls of Triplet values in SCan lines; again these may optionally include Run length compression. Each triplet will be evaluated according to the current colour model. The routines should be referenced once for each scan line required on the cell mapped area.

```
      CETSC(V1,V2,V3,NC)
```

will cause the NC triplet values defined by real arrays V1,V2,V3 (each of dimension NC) to be applied as the values for NC cells in the next cell scan line. The value of the i-th cell will be evaluated as the triplet {V1(i),V2(i),V3(i)} according to the current colour model. (The triplet values should be within the relevant range.)

```
      CETSCR(V1,V2,V3,IRUN,NC)
```

is for run length encoded scans. The integer array IRUN (of dimension NC) holds the run length corresponding to each of the NC triplets defined by the real arrays V1,V2,V3 (each of dimension NC). The i-th triplet {V1(i),V2(i),V3(i)} will be evaluated according to the current colour model and applied to IRUN(i) consecutive cells in the scan. Thus the length of the scan defined by this reference is the summation of IRUN(1,i) for i=1 to NC. (Negative or zero runs ignored.) Each value triplet should be within the range appropriate to the current colour model.

PART 2 - GRAPHING

Chapter 1 - Basic Graphing

1.1 - Introduction

DIMFILM includes a collection of subprograms to facilitate the production of graph plots, while retaining the advantages of the overall system. The graphing routines may be freely intermixed with the general plotting routines. The system is designed to permit the user considerable freedom, and although this must inevitably introduce some pitfalls for the unwary, a large amount of checking is performed, and warning messages are included. These are controlled as are those in the general plotting routines, through the check level. Users interested primarily in graphing output should first familiarise themselves with the earlier sections on basic plotting.

Graphing may be freely intermixed with calls to all other plot routines. DIMFILM attempts to ensure that on exit from any graphing routine, all internal parameters are set to their values on entry. The beam is also positioned so that general plotting may be resumed easily. Such properties as intensity, orientation and height of symbol strings, broken lines etc., will be maintained. It is possible therefore, to interrupt a diagrammatic plot to perform graphical plotting and then to resume. A few limitations are, however, necessary. Primarily, on return, a broken line pattern will commence at the beginning of its sequence (the consequence of a beam off movement). Symbol strings will commence with all their initial features active. If TXTCON has previously been called, it will be operable on all but the first subsequent calls to SYMTXT.

Except where otherwise indicated, all graphical drawing routines are performed as CSType 1 operations. All graph textual annotation will be treated as CSType 2, while non-textual annotation is a CSType 3 operation. All graph titling and labelling (user supplied) will be subject to any escape sequences as described in Section 3. Graphs will also be subject to interruption by markers, when the routine INTRPT has been called. This applies only to plots where a set of data values has been supplied; when interpolation is used the interrupt only applies to the supplied data points ie. a call of INTRPT(N,CHAR) would cause each Nth data point to be marked with character CHAR. The height used would be that currently used outside the graph plot.

The use of intermixed general and graph plotting will be illustrated in a later section. It is emphasised that this mixing may be performed freely within a frame, and at the simplest level it may mean only the changing of intensity for axes etc.

1.2 - Graph Location

The overall size of a graph is taken to be identical to the current pre-clipping rectangle. That is, the user by making a call to PANE (or through the default pre-clipping rectangle being the same as the overall bounds) has set the area to be occupied by the graph. This area includes a margin for captions and values, and the proportions used are preset in DIMFILM.

Of specific interest to the user, is the area occupied by the actual graph plot, excluding margins etc. This is 83% of the pre-clipping rectangle in each direction. The graph labels and values will be in the margins between this rectangle and the actual graph plot. Each margin in the X(or Y)-direction will be 8.5% of the corresponding X(or Y)-dimension of the rectangle.

This enables the user to position the graph plot anywhere within the bounds and graphs in equal area will appear identical, whilst those of similar areas would be reduced or enlarged accordingly. This ensures continuity over a large number of plots and assists the user in making more than one plot in an area, but changing axis scales as required. The actual co-ordinates used in calls to PANE etc. are not referred to by the user during the course of graphical plotting All scaling of graphs is accomplished internally. The X-axis of a graph will always by parallel to the X-direction of the frame and the Y-axis parallel to the Y-direction. Hence, the orientation of a plot is determined by the current mode of the device.

1.3 - Axis determination

In any graphical work the user is obliged to determine the axis types required before commencing a plot. There are few conditions imposed on the user performing graph plotting with DIMFILM, and these are summarised below. The axis type need only be declared if a change is to be made, and clearly this must be done before plotting or assigning values to the axis etc. is performed. By default, it is assumed that both X-and Y- axes are linear. A number of routines are available for changing or resetting axis types, at present the user will only be concerned with:

LINX	to set a linear X-axis.
LINY	to set a linear Y-axis.
LINXY	to set linear X- and Y-axis.
LOGX	to set a logarithmic (base 10) X-axis.
LOGY	to set a logarithmic (base 10) Y-axis.
LOGXY	to set logarithmic (base 10) X- and Y-axis.

Any combination of these calls is permissible, the last declared type for each axis being accepted.

For a basic graph, the user will have a set of (X,Y) co-ordinates that are to be plotted. It may be that the plot is to be made along axes that have ranges automatically determined by the X- and Y- ranges of the data, or alternatively along axes having predefined ranges. The default is for automatic range determination along both axes to be performed as in the former requirement. For the latter requirement, the user may supply a range along either or both axes by the following.

```
XAXIS(XL,XR)

YAXIS(YL,YR)
```

where XL is the left bound and XR the right bound for the X-axis and YL is the low bound and YU the upper bound for the Y-axis. In each case the bounds are given in terms of the user’s graph units.

Should both bounds be equal in a call to one of these routines, then DIMFILM will go into auto mode for that axis.

Axis ranges set in this way remain set until such time as they are either set to some other value or auto mode is restored using the following routines:

AUTOX to set auto range determination on the X-axis.

AUTOY to set auto range determination on the Y-axis.

AUTOXY to set auto range determination on both axes.

1.4 - Drawing the Graph

Having determined axis types and ranges as required, a single call to the routine

```
GRAPH(X,Y,N)
```

will accomplish the plotting. X and Y are REAL arrays (of dimension at least N) holding the N X- and Y- coordinates respectively. This will join consecutive co-ordinates by straight line segments. The co-ordinates should be supplied monotonically in X. N must be at least 2.

To complete the graph with labels, values and to frame it, the following routine may be called:

```
GRDEF(TITLE,XLAB,YLAB,IXY)
```

where TITLE,XLAB and YLAB are all strings of type CHARACTER with an appropriate length:

TITLE is the title to appear at the base of the graph.

XLAB is the label to appear below the X edge of the plot.

YLAB is the label to appear to the left of the Y edge of the plot.

IXY is used to control the presence or absence of axes, according to its value - =0 :- neither axis will appear, =1 :- the X-axis only will appear and be ticked, =2 :- the Y-axis only will appear and be ticked, =any other value :- both axes will appear and be ticked.

In addition to these titles, labels and axes, the plot will be framed (i.e. the actual graph will be boxed), all edges will be ticked and values will appear adjacent to the ticks.

(It should be noted that any axis can only be plotted if it lies within the range of the orthogonal co-ordinate. For example the X-axis at y=0 can only be plotted if zero is included in the Y-range.)

Essentially, this routine is present to offer a default set for the annotation of the plot. It is not expected that all users will wish for this set of annotations and consequently a large number of routines are included to perform the individual parts. It is suggested that each user constructs his own standard set of calls in a single subroutine and references this at the end of each plot. Of course, he may wish to annotate each plot differently and in this case, the full range of routines may be accessed. These routines will now be described.

A simple convention is used to refer to the sides of any line. For lines in the X-direction there is a lower (L) and upper (U) side, referring to the normal Y-values which are orthogonal to this line.

Similarly, for a line in the Y-direction there is a left (L) and right (R) side. The graph itself may, or may not have the axes included in it and is bounded by four edges, surrounding which are the margins; the whole being contained exactly within the pane.

For all the following routines that perform titling or labelling, the symbol strings are passed through the SYMTXT routine and all the facilities described in section 3.1 are available (although it should be noted that text continuation will not operate, each string commencing with the selected features). Text strings will be appropriately centred.

Titles may be placed below the lower edge or above the upper edge by the following.

```
      LTITLE(TITLE)
```

for titling below the lower edge

```
      UTITLE(TITLE)
```

for titling above the upper edge

TITLE is a string of type CHARACTER of the appropriate length, which contains the text to be passed to SYMTXT.

Labels may be placed against any edge or either axis, although for the latter option it is necessary for the axis to be within the orthogonal axis range.

```
      LXLAB(LABEL)
```

labels the lower edge

```
      UXLAB(LABEL)
```

labels the upper edge

```
      LYLAB(LABEL)
```

labels the left edge

```
      RYLAB(LABEL)
```

labels the right edge

LXALAB(LABEL)

labels below the X-axis

UXALAB(LABEL)

labels above the X-axis

LYALAB(LABEL)

labels to the left of the Y-axis

RYALAB(LABEL)

labels to the right of the Y-axis

In each of these calls LABEL is a string of type CHARACTER of the appropriate length, which contains the text to be used as the label.

The edges and axes may be ticked as required, the ticks corresponding to steps in the range in the appropriate direction and taking account of the axis type. The spacing of ticks etc., will be considered below.

LXTIK will cause ticking of the lower edge

UXTIK will cause ticking of the upper edge

LUXTIK will cause ticking of both the lower and upper edges

LYTIK will cause ticking of the left edge

RYTIK will cause ticking of the right edge

LRYTIK will cause ticking of both the left and right edges

In each of these cases the ticking would be directed inwards.

LXATIK will cause ticking on the lower side of the X-axis

UXATIK will cause ticking on the upper side of the X-axis

LYATIK will cause ticking on the left of the Y-axis

RYATIK will cause ticking on the right of the Y-axis

Values corresponding to the ticking of edges and axes may be plotted.

LXVAL plots values below the lower edge

UXVAL plots values above the upper edge

LUXVAL plots values below the lower edge and above the upper edge

LYVAL plots values to the left of the left edge

RYVAL plots values to the right of the right edge

LRYVAL plots values to the left of the left edge and to the right of the right edge

Similarly for the axes,

LXAVAL plots values below the X-axis

UXAVAL plots values above the X-axis

LYAVAL plots values to the left of the Y-axis

RYAVAL plots values to the right of the Y-axis

The axes themselves may be drawn by the following

DRAWXA to plot the X-axis

DRAWYA to plot the Y-axis

As an alternative to ticking the edges etc., the user may grid the plot entirely using the following

XGRID will plot grid lines normal to the X-axis (i.e. parallel to the Y-axis)

YGRID will plot grid lines normal to the Y-axis

XYGRID will completely grid the plot in both directions

It is clear that when ticking (or gridding) and valuing are applied, or labelling an axis, it is essential that the ranges for each axis are defined first. They may either be explicitly defined by calls to XAXIS and YAXIS, or if auto mode is in operation on either axis,they will be known internally after the plotting of the graph (i.e. as described in this section through a call to GRAPH). Clearly then a sequence for these calls has been formulated. This may be summarised in priority, after the definition of the pre-clipping area:

1) axis type declarations

2) axis range specifications

3) ticking, valuing, gridding or labelling axes

Titling and edge labelling may occur at any point. If automatic mode is in operation on either axis then the graph call which fulfills the function of range determination must occur at step 2, otherwise it may appear anywhere after stage 1. Steps 1 and 2 will be retained from any previous graph plot and so do not need redefinition for subsequent plots. Of course, if auto mode is active then the previously determined range(s) will hold until reset by a call to a graph plot routine.

Although the user has considerable freedom, it is suggested that the safest course is :

1) axis type declarations (if changes to be made)

2) axis range specifications (if non-auto, or to be reset)

3) graph plot

4) all annotations (in any order)

The edges of the graph area may be plotted by

```
      GRFRAM
```

This will be required in nearly all cases. The frame is drawn as CSType 3.

To outline the current pane, call

```
      OPANE   (as described in section 2 - CSType 3.)
```

The determination of value intervals (also used for ticking and gridding) is performed automatically (although see section 4.2 later) depending on the axis ranges.

For linear axes with a range of length d x 10**i with d greater or equal to 1.0 and less than 10.0 (i.e. d in [1.0,10.0)) an interval between values will be chosen according to the value of d as shown below:

```
        d in [1.0,3.0)     interval = 2.0 x 10**(i-1)

        d in [3.0,7.0)     interval = 5.0 x 10**(i-1)

        d in [7.0,10.0)    interval = 1.0 x 10**i
```

The values marked will be integral multiples of the interval, thus ensuring that 0.0 would be a value point. The choice of these intervals maintains a reasonable number of divisions without overcrowding (always more than 4 and less than 16). The values themselves will be plotted in a format to maintain the maximum accuracy without excessive digits appearing. Where possible a common power of ten factor will be removed (and indicated adjacent to the edge or axis being 'valued').

The same value points are used for the tick spacing, the tick lengths being one hundredth of the size of the shorter pane side.

For logarithmic axes, to ensure clarity, if there are less than two cycles along the axis values of the logarithm (base 10) the values will be plotted and determined as for the linear axes. If at least two cycles are contained along the axes then the values as integer powers of 10 will be appropriately placed at cycle boundaries.

Ticking will occur at these boundary points, and if fewer than five cycles are contained on the axis, then intermediate half-size, faint ticks will be plotted (these correspond to the normal intermediate divisions of a log cycle).

When gridding is specified the spacing is the same as the corresponding ticking as described above.

Chapter 2 - Refined Graphing

The features described in the previous section enable the user to plot straight line graphs from a set of co-ordinate pairs, against linear or logarithmic axes. The immediate extension of this is for some form of interpolation to be performed between consecutive data points to give a smooth curve for the finished plot. DIMFILM provides the choice of two polynomial interpolations - third or fifth degree. To effect either of these the call to GRAPH should be replaced by

```
      POLY3(X,Y,N)
```

for third degree polynomial interpolation

```
      POLY5(X,Y,N)
```

for fifth degree polynomial interpolation

X and Y are REAL arrays (of dimension at least N) holding the N pairs of co-ordinates to be graphed. There is a proviso that N must exceed the degree of the polynomial interpolation being performed. Plotting of an interpolated graph will terminate should a singularity be found during the interpolation process.

By default the number of steps between consecutive abscissae at which interpolated values will be found is 5. The larger this value, the smoother the resultant curve, although execution time for the program will increase correspondingly. The number of steps for this interpolation may be altered by

```
      INTERP(NSTEPS)
```

where NSTEPS is the number of intermediate steps at which the interpolation polynomial will be evaluated.

An alternative output that might be required from a set of co-ordinate pairs might be a simple point plot, when no join is made between the points. To accomplish this,

```
      PTPLOT(X,Y,N,NCHAR)
```

where X,Y and N are as before, and NCHAR is the number of the required marker symbol at each point (from the current Marker Font), NCHAR is an integer in the inclusive range 1 to 48 . As for the normal call to GRAPH, N must be at least 2.

When discrete co-ordinate pairs are provided, it may be desired to produce a histogram bar chart. The bars will in each case be in the Y- direction, and correspond to each abscissa input. The appropriate instruction is

```
      HISTGR(X,Y,N,BAR)
```

X,Y and N being as before.

The fourth parameter, BAR, controls the type of histogram that will be produced.

BAR = -1. - auto-width histograms will be produced, so that adjacent bars touch.

= 0. - histogram lines are produced (i.e. a special case of fixed width bars).

> 0. - histogram bars width BAR (in user axis units) will be plotted.

Several variations of histogram bar charts are possible.

A step histogram may be plotted by

```
      STEPGR(X,Y,N)
```

the three arguments being the same as the first three arguments for HISTGR.

The effect is to produce an auto-width histogram with common verticals removed. This is the common staircase type of histogram.

It is possible to shade the histogram bars (when bars of a finite width are to be produced). This is performed by

```
      SHDEGR(X,Y,N,BAR,THETA,GAP)
```

The first four arguments are those for HISTGR, with the limitation that for BAR = 0.0 histogram lines (i.e. bars of zero width) only will be produced. The last two arguments are used to define the type of shading, and correspond to the arguments for HATCH (section *.x)

THETA is the angle (AGroup1, default degrees - see I.1.9) the shading lines make with the positive X-direction, and should be in the range (-90.0,90.0)[degrees].

GAP is the perpendicular distance, in user overall units, between consecutive shading lines.

The shading of each bar is performed separately, and is defined under HATCH. The user may force two histogram bar charts, each with different shading types, to be plotted with bars of the two data sets adjacent for comparison by the appropriate positive specification for BAR, and a shift in X range (by BAR) between the calls to SHDEGR for the two data sets.

With all the histograms, it is the case that a vertical line may coincide with an edge of the graphical area. DIMFILM has no knowledge of whether the user will draw the edges, and consequently the default is to plot such verticals. However, when the Y edges are plotted, such verticals will be superimposed, which may not be desired.

HREP causes the suppression of histogram verticals that coincide with the Y-edges, while

HPER restores the default, whereby the plotting of such verticals is permitted.

It will often be the case that a graphical plot of a known function is to be made. In this case the appropriate call would be

```
      GRAPHF(FN)
```

where FN is a REAL FUNCTION, supplied by the user, and which should be declared EXTERNAL in the calling program. The function should have two REAL arguments and be referenced as Y = FN(X,DX)

where the value returned in the function is the function value (i.e. Y co-ordinate) at X. The argument DX on return should be the increment in the X direction to be made for the next plotted point. These values are all in terms of the axis co-ordinate scales.

GRAPHF starts at the minimum X-axis value, and proceeds until the maximum is reached or , in the case of a curve which becomes double valued in X , the minimum is passed. That is, plotting commences until the graph leaves the X-axis range. Alternatively, the graphing will cease when two consecutive zero values for DX are returned. The general comments under Part I, section 4 relating to FNPLOT apply, with references to X and Y ranges being, in the case of GRAPHF, references to the axis ranges.

With GRAPHF it is obvious that automatic axis scaling cannot be applied; hence both axis ranges must have previously been determined. If either axis is in auto mode at the initiation of a call to GRAPHF, then the current axis range for that axis will hold, ie. it will be the range used for the immediately preceding graph plot (possibly determined under auto-mode). Should no previously defined range exist then an error condition will occur and no plot will result.

All the above graph plotting routines take full account of the various axis types.

A further axis type is available, and is of particular use for commercial plots. This is the month axis.

```
      MONTHX(SMONTH)
```

will set the X-axis to months,

```
      MONTHY(SMONTH)
```

will set the Y-axis to months.

In either case, the REAL argument SMONTH is used to indicate the starting month along the appropriate axis. The months January to December running from 1. to 12. respectively, a fractional value is permitted to denote the plot is to commence at a fraction of a month. The left limit will be the 1st of this month.

E.g. February 7th would be 2.25 for SMONTH, and the axis would commence at February 1st with the first plot occurring at February 7th.

The value of SMONTH must be in the range 1.0 to 13.0 . If an illegal value is requested, 1.0 will be substituted.

A subsequent call to GRAPH, POLY3, POLY5, PTPLOT, or HISTGR with arguments (X,Y,N) (and BAR in the case of HISTGR, and NCHAR in the case of PTPLOT) will cause plotting with the appropriate month axis from the starting point SMONTH at monthly intervals for the N values along the orthogonal axis. This is the auto-mode month axis, the argument N being the number of months to be as plotted. The orthogonal axis values should be held in the REAL array X or Y as appropriate, the other array being a scratch array that will be used during the call, and its contents lost.

This situation will be clarified by example:

```
LOGY   Y-axis set as a log axis

MONTHX(3.0)   X-axis a month axis, commencing at March 1st

GRAPH(X,Y,N)
```

This will plot the N values from REAL array Y at monthly intervals starting at March 1st, i.e. Y(1) will be the Y value plotted against March 1st, Y(2) against April 1st, etc. REAL array X must be available for use by GRAPH. The X axis will be divided into months.

Similarly,

```
LINX   X-axis set as linear axis

MONTHY(3.0)   Y-axis a month axis, starting at March 1st

GRAPH(X,Y,N)
```

will plot N values from REAL array X at monthly intervals commencing with March 1st, i.e. X(1) will be the X-value plotted against March 1st, X(2) against April 1st, etc... REAL array Y must be available for use by GRAPH. The Y axis will be divided into months.

After a call of MONTHX or MONTHY the relevant axis will remain a month axis until reset (for example, by a reference to LINx or LOGx). It is not permissible for both axes to be of type month for any plot.

Ticking and valuing will apply to month axes with each month being ticked when 12 or fewer months are plotted along the axis. For greater numbers of months not every month will be marked, as indicated:

Number of months	Month interval for ticking, etc.
<=12	1
(12,24]	2
(24,36]	4
(36,60]	6
>60	12

The above methods have shown the use of automatic month plotting, when each successive data point is exactly a month from the previous point.

It may be necessary to plot at intervals other than exact months, in which case it is necessary to define the number of months that are to be plotted along the month axis.

MONSX(NMNTHS)

sets the number of months along the X-axis

MONSY(NMNTHS)

sets the number of months along the Y-axis

NMNTHS is the number (INTEGER) of months to be plotted along the appropriate axis. These calls do not replace the calls to MONTHX or MONTHY, which should also be made. Any subsequent call of AUTOX, AUTOY or AUTOXY will reset the relevant month axis to auto mode.

When this mode of fixing the month range is in operation the previously scratch array in the X,Y arguments to GRAPH etc., is used differently. It should now hold the offset, in months, from the beginning of the starting month (as set by MONTHX etc). An example will illustrate this:

LOGY sets logarithmic Y-axis

MONTHX(2.5) sets X-axis to months, beginning February 14th for auto mode month plots.

MONSX(10) indicates fixed number (10) months along X-axis

GRAPH(X,Y,N)

N points will be plotted, Y(I) will hold the Y-value of the Ith point (I=1,N) and X(I) holds the offset, in months, from the 1st February (this being the start month), although the axis will start at the 14th (as set).

Suppose X(1)=1.25 , X(2)=1.50 , X(3)=2.00 , ...

Y(1) would be plotted as March 7th (1.25 months after Feb 1st)

Y(2) would be plotted as March 14th (1.50 months after Feb 1st)

Y(3) would be plotted as April 1st (2.00 months after Feb 1st)

etc.

When discrete data is provided to a graphical routine (excepting, therefore, GRAPHF) the user may wish symmetric or asymmetric error bars to appear for each data point in the X and/or Y directions.

Separate calls exist for this purpose, and are listed below:

```
      GRAPHE(X,Y,N,MX,MY)
```

which corresponds to GRAPH, producing straight line joins between data points.

```
      PTPLTE(X,Y,N,NCHAR,MX,MY)
```

corresponding to PTPLOT, each data point being plotted with the marker designated by NCHAR

```
      POLY3E(X,Y,N,MX,MY)

      POLY5E(X,Y,N,MX,MY)
```

corresponding to POLY3,POLY5 respectively with polynomial interpolation between data points

The arguments are of somewhat different form than the corresponding routines.

X and Y are real arrays of dimension X(N,MX), Y(N,MY), where N is the number of data points.

MX and MY may each have the value 1, 2 or 3.

Considering the array Z(N,MZ); Z=X,Y, then in each case Z(I,1) holds the Ith Z data value which will be plotted.

For MZ=1 no error bars will be produced in the Z-direction.

MZ=2 implies symmetric error bars should appear in the Z-direction, and for the Ith data point, the Z error bar will be +/- Z(I,2) in magnitude.

MZ=3 implies asymmetric error bars in the Z direction, in which case the Ith data point will have Z error bar given by +ABS(Z(I,2)), -ABS(Z(I,3)).

The error bars are plotted with the same line type as the graph, although the tick heads to each bar will be solid. It should be noted that MX need not equal MY, and for error bars in one direction only the other MZ should be set to 1. Tick heads will have the same size as other graphical ticks.

To force the error bars to be of different line type it is necessary to make a standard call to GRAPH etc. followed by the required line type change and then a call to PTPLTE with NCHAR set to 0 (unless some character is required at the data points additional to the error bars).

It is possible for axes to be drawn at any particular X- or Y-value, i.e. at other than the normal zero lines. To set this

```
      AXCUT(XO,YO)
```

where (XO,YO) is the co-ordinate pair at axis intersection. By default this will be at (0.,0.). The axis intersection point will then define axis positions for all subsequent axis drawing, ticking labelling etc., until reset.

The user may wish, for linear axes, to control the interval between ticks, and hence values.

```
      DIVUNI(DIVX,DIVY)
```

(with DIVX,DIVY 0.0) will cause all subsequent linear axis ticking etc., to be at intervals of DIV- in the axis units for the relevant axis. If this value is found not to be sensible automatic ticking etc., will be performed. The condition for DIV- to be used is that it is less than one-half and not less than one-hundredth of the relevant axis range. It may be specified for only one axis by setting the other DIV- non-positive.

To return to automatic division determination on both axes

```
      AUTD
```

There is also provision for controlling the intensity step for the intermediate logarithmic axis ticks.

```
      LOGTIK(RSTP)
```

will result in a step down in intensity of RSTP for the intermediate ticks. Note that RSTP is a real value in the range 0.0 to 1.0 . The default value is 0.2

The user may also control the criteria used for determining whether log cycles should be ticked. Section 1 (of Chapter 4) indicated that if less than two cycles appeared along an axis, then values of the logarithm would be plotted, while if five or more cycles were contained on an axis, no intermediate ticks would appear. To change these values:

```
      LOGLIM(ZMIN,ZMAX)
```

where ZMIN and ZMAX are the real values of these criteria. ZMIN being the minimum number of cycles for cycle values etc., to appear, ZMAX the number of cycles at and above which intermediate ticking will not appear.

When automatic mode is selected, each graph has individually determined axis ranges (on either or both as specified AUTOX, AUTOY etc.). It is often the case that while automatic range determination may be used for the first graph, it may be wished to plot subsequent graphs against the same axis ranges. This may be forced by:

SAMEX which fixes the X-range as that currently held

e.g. after a call to a graphical range that X range is retained (as though the relevant call to XAXIS had been made) for all following graphs, and automatic X-range determination is turned off.

SAMEY acting on the Y range

SAMEXY acting on both X and Y ranges

By default axis values are plotted in a real format (i.e. including the decimal point). It is possible, when the range permits, for integer values to be plotted by linear axes or edges.

IXAX specifies that future valuing of linear X-axes should, when possible, be in integer format

IYAX acts similarly for subsequent linear Y-axes

Real values may be restored for all subsequent valuing by:

RXAX for real values on X-axes

RYAX for real values on Y-axes

Common scale factors may well be included for integer values axes where necessary, and it should be appreciated that ranges such as (10.0,10.9) are incapable of being valued integrally. Also, the user should note that the X and Y values passed to GRAPH etc., should still be real numbers.

Chapter 3 - Alternative Co-ordinate Systems

In addition to the rectangular co-ordinate systems described above, DIMFILM provides the means to plot against polar axis systems, i.e. radial plots.

The area occupied by a polar plot is the same portion of the window as used for cartesian plots. In the case of polar plots, however, the total area is not used, the scaling and positioning of the axis system being performed to maximise the area used.

The calls are closely analogous to those previously described for rectangular co-ordinate systems, and the priorities for ordering of calling sequences as given in Section 1 still hold.

The radial axis type may be specified by:

```
      LINR   to set linear radial axis (this is default)

      LOGR   to set logarithmic radial axis
```

The units of angular measure for angles to the radial plotting routines belong to AGroup2 (therefore being radians by default). The section on Angular Measure within the part on Basic Plotting gives further information and describes ways of changing the default.

Axis ranges may be supplied by the user:

```
      RRANGE(RMIN,RMAX)
```

will set the radial range to be RMIN to RMAX, where RMAX is greater than RMIN which is greater than or equal to 0.0

```
      TRANGE(THETA1,THETA2)
```

will set the angular range from THETA1 to THETA2, both angles being AGroup2 (default radians) and should each be in the range [0.0,2pi][radians].

If either of these calls is made the appropriate axis range is fixed until reset. By default, automatic range determination is provided. This may be reselected at any time:

```
      AUTOR   sets automatic radial range determination

      AUTOT   sets automatic angular range determination

      AUTORT   sets automatic range determination for both axes
```

Having determined axis types, ranges as required:

```
      POLAR(R,THETA,N)
```

will plot the N co-ordinate pairs from R, THETA (both REAL arrays of minimum dimension N), where R holds the radial values, and THETA holds the angular values (AGroup2, default radians).

The graph may be completed by

```
      POLDEF(TITLE,RLAB)
```

where TITLE and RLAB are CHARACTER arrays (of appropriate length) holding captions to be passed to SYMTXT

TITLE is the title to appear at the base of the plot

RLAB is the label to appear below the plot

The polar boundary will be included with radii at 90 degrees, each radius will be ticked and valued, and angular values will be plotted around the circumference.

Titling etc., may be performed through LTITLE, UTITLE, LXLAB, UXLAB, LYLAB, RYLAB as for rectangular plots. Due to the varied boundary shapes and positions that may apply to radial plots it is not sensible to provide any rigid labelling format. Generally, the -XLABEL should be satisfactory for describing the radial axis, and -YLABEL for the angular values.

Ticking and valuing may be independently controlled. In the following description it should be borne in mind that in the general case the extremities of a plot will not be coincident (only in the exceptional case of the angular range 2pi [=360 degrees] will this be so), and between the extremities (i.e. going from THETA1 to THETA2, the angular range limiters) one or more principal radii may be included (i.e. those at integral multiples of pi/2 radians, viz. 0, pi/2, pi, 3pi/2, 2pi [equivalent to 0., 90., 180., 270. degrees]).

POLOUT will produce the bounding outline of the polar plot area - i.e. the arcs of minimum and maximum radius, and the radii at the bounding angles will be plotted.

POLFR will produce the bounding outline (as POLOUT) and also plot the included principal radii.

RXVAL will plot the radial values at the extreme radii.

RVAL will plot the radial values at the extreme radii and at included principal radii.

RXTIK will tick the radial values on the extreme radii.

RTIK will tick the radial values on the extreme radii and on included principal radii.

It should be noted that, generally, the tick length is taken into account when valuing radii to avoid any overlap of tick and value. The user may exercise some control over this by:

```
      RADSEP(ANGLE)
```

ANGLE is specified as AGroup2 (default radians). It controls the appearance of ticks for principal radii. If a principal radius is closer than ABS(ANGLE) to an extreme it will not be ticked. For non-coincident radii of angular range extremities to be separately valued there is a minimum necessary angular separation. If ANGLE >0.0, this separation is ANGLE. For zero or negative ANGLE automatic computation taking account of value positioning and character size is made. At less than the (fixed or computed) separation angle only the radius of least angle will be valued. While for less than twice the separation angle, values will not be spaced by the tick size from the radius, i.e. values will be adjacent to radii. The default value of ANGLE is -0.1 (radians), i.e. automatic computation of separation angle is made, but included principal radii closer than 0.1 radians to an extrema are not ticked.

```
      POLGRD   will grid the polar plot.
```

The gridding of a polar plot comprises the plotting of radial lines and circular arcs at certain radii. Due to the concentration of radii at the centre of a polar plot it is generally necessary to incorporate some thinning of radial lines as the centre is approached.

This is done by dividing the maximum radius, and each division having a separate angular increment between radial grid lines. The circular arcs that appear are dictated by the ticking (and valuing) of the radii, consequently the division points must be adjusted to coincide with these arcs. The number of divisions fixes the size of each division, and the tick point along a radius following nearest to each division is selected to bound a division. However, a spacing less than .4 of the fixed division is ignored to prevent nonsensical thinning of radii.

By default 3 divisions are used with angular separation of radii being pi/2, pi/4, pi/12 (radians) [equivalent to 90, 45, 15 degrees]. These may be changed by the user by

```
      TLEVS(XLEV,N)
```

where N, INTEGER, is the number of levels to be used for the thinning of gridding radii - a maximum of 4 is permitted 1<=N<=4 and XLEV is a REAL array of dimension 4, with the first N elements holding the angular intervals between radii at each division (from 1 at the centre to N at the maximum radius). The angles belong to AGroup2 (default radians).

To cause angular values to be plotted along the arc of maximum radius,

```
      TVAL
```

The angles so plotted will always be in degrees, and will occur at the interval specified for the highest level set (i.e. by default at pi/12 radians).

Ticking of the outer arc may be done at the angular intervals by

```
      TTIK
```

for linear intervals on the radii, overriding of the automatic division control is possible by

```
      RDIV(D)
```

when D, REAL, will be the interval between ticks etc., on the radii. This is subject to the same conditions as the similar feature for cartesian plots.

Automatic division may be restored by:

```
      AUTORD
```

Logarithmic axis ticking is controlled as for cartesian plots, and the call to LOGLIM (described in section 2) controls logarithmic limits for both types of plot.

Analagous with the calls for interpolation, point and function plotting against rectangular axes (as described in section 2) are calls for the similar purpose with polar plots.

```
      POLPY3(R,THETA,N)
```

for third degree polynomial interpolation.

```
      POLPY5(R,THETA,N)
```

for fifth degree polynomial interpolation.

R and THETA are REAL arrays (of dimension at least N) holding the N pairs of co-ordinates to be plotted. N must exceed the degree of the interpolation polynomial. The number of interpolating steps between consecutive THETA values is controlled by INTERP (see section 2), and by default is 5. THETA is of AGroup2 (default radians).

Plotting a symbol at each co-ordinate pair with no connecting line is accomplished by

```
      POLPTS(R,THETA,N,NCHAR)
```

where R, THETA and N are as before, and NCHAR (1<=NCHAR<=48) is the desired marker symbol from the Marker Font. (N>2).

To plot a polar function,

```
      POLFN(FN)
```

where FN is a REAL FUNCTION, supplied by the user, and which should be declared EXTERNAL in the calling program. The function should have two REAL arguments and be referenced as

```
       R=FN(THETA,DTHETA)
```

where the value returned in the function is the radial value (i.e. R) at angle THETA, and DTHETA, on return is the increment to be made in the angle for the next plotted point. THETA and DTHETA are both in AGroup2 (default radians).

The general details of this call and its operation are directly analagous to GRAPHF, the description of which is in section 2, and to which the user is referred.

Analagous to SAMEX, SAMEY, SAMEXY are the references

SAMER which fixes the R range at the current range

SAMET fixing the angular range

SAMERT fixing both the radial and angular ranges

A different form of plot concludes this section. This is the "pie chart" plot. In this, data is apportioned a sector of a circle in direct proportion to its fraction of the total. As with all graphical plots, the piechart occupies a portion of the current window. Within this space the chart is centred.

```
      PIECHT(R,CAP,PERCNT,N,IPERS)
```

R is the radius of the chart, in current users units, and this must not exceed half of 83% (the graph area) of the minimum pane dimension.

N is the number of data items to be plotted.

CAP is a CHARACTER array of dimension N, containing the N captions of the data items. When percentages are to be plotted for each data item, this will be appended to each caption.

IPERS indicates the form of data input, and whether percentages are to be plotted.

If IPERS is negative, the actual data value of each item is input through PERCNT, in which case there will be exactly N sectors. When IPERS is positive, the actual percentages associated with each are input through PERCNT. In this case, a check is made by totalling the percentages 1 to N, and if a total in excess of 100.1 is found an error is recorded and the call ignored. Conversely, a total less than 99.9 will result in a blank sector appearing to force the total to 100.

When the absolute value of IPERS is 1 no percentage will accompany the data names on the plot, while an absolute value of 2 causes percentages to be plotted.

PERCNT is a REAL array of dimension N, holds the percentages or values (as determined by IPERS) associated with each item.

This section contains details of a handful of routines that will probably be accessed only infrequently by the user. They do, however, offer the means for the more ambitious to obtain superior graphical plots. Unless specifically stated, the following facilities refer only to plots against rectangular axes, i.e. those discussed in sections 1 and 2.

Although DIMFILM is able to provide ticking of the various bounds and axes, as described in II 1.4, it has no prior knowledge of whether axes will in fact be drawn. Where axes are included, a tick at the axis point will result in a heavier density or a colour "change" where axis and tick coincide. Conversely, if the axis tick were omitted and the axis not drawn then a seemingly nonsensical gap in the ticking would appear. By default ticking is suppressed at the axis as it is felt that generally an axis will be incorporated if it falls within range. To reverse this, so that ticking occurs at the axis point the following routine may be called :

```
      AXTIK
```

The default condition may be restored by :

```
      NOAXT
```

It is obviously the user who must consider whether or not his axis will be contained in the plot.

The ticking of bounding edges is directed inwards. A set of routines exist to place outward pointing ticks on the bounding edges:

```
      LXOPT   for outward ticking of the lower edge

      UXOPT   for outward ticking of the upper edge

      LUXOPT  for outward ticking of both upper and lower edges

      LYOPT   for outward ticking of the left edge

      RYOPT   for outward ticking of the right edge

      LRYOPT  for outward ticking of both left and right edges
```

It is also possible to plot any combination of edges (rather than the complete frame of four edges, through GRFRAM - see II.1.4). The ticking of the open ends of the plotted edges may also be controlled.

```
      EDGES(LKJI,ITICK)
```

LKJI is a four digit INTEGER, each digit being 0 or 1. There is a 1:1 correspondence between the digits and the four edges, and a value 1 causes the relevant edge to be plotted, while for 0 it is not plotted.

The correspondence is:

I - lowest edge

J - left hand edge

K - uppermost edge

L - right hand edge

ITICK controls the ticking of the open ends of the plotted edges (i.e. at ends of the plotted edges not meeting another plotted edge). The values ITICK may have are :

+1 - inward ticks

-1 - outward ticks

+2 - in/out double ticks

0 - no ticks

For example CALL EDGES(0011,2) would produce the lower and left edges with double ticks.

It may sometimes be more convenient to know the actual plotting co-ordinates (in the users overall frame system) of a point known in terms of currently used graphing axes. This is necessary when one wishes to overplot text on a graph, or, possibly, to blank out a section of the graph (where it is known that no relevant data will occur) for future use as a legend box.

To obtain the actual plot co-ordinates (XA,YA) of a point with current graph co-ordinates (XG,YG) , call the following:

```
GRTOAC(XG,YG,XA,YA)
```

Note: the desired coordinates (XA,YA) are actual coordinates in terms of the users (world) coordinates (e.g. as determined through a reference to BOUNDS). They take no account of any rotation or temporary origin. As such they are suitable for use as blanking/clipping parameters, and should only be used for plotting when rotation and temporary origin are both zero.

There is provision for movement from the current position to a point referred to by its graph coordinate (XG,YG); this may be either a positional or drawing move.

```
      GRON(XG,YG)
```

is equivalent to ON2; drawing will occur from the current plot position to the actual point with graph coordinates (XG,YG) at the current colour/intensity and in the current style (i.e. CSType 1).

```
      GROFF(XG,YG)
```

is equivalent to OFF2; the current plot position will be repositioned to the actual point with graph coordinates (XG,YG) without drawing any line.

Note: in these cases the movement is to the actual point with graph coordinates (XG,YG). This point will become the current plot position. Effectively rotations and temporary origin are accounted for (although there is no rotation within the graphing area).

```
                    The sequence
                        CALL GRTOAC(XG,YG,XA,YA)
                        CALL ON2(XA,YA)
                   will not generally have the same result as
                        CALL GRON(XG,YG)
```

When there is no rotation and no temporary origin the results will be identical, otherwise the coordinates (XA,YA), which are actual world coordinates, will be transformed by ON2 according to the existing rotation and temporary origin before the move is made. It is emphasised that (XA,YA) are the actual coordinates relative to the user’s bounds; they are not relative to the current plotting axes.

Directly analogous routines exist for polar plots. These are

```
      POLTOA(R,THETA,XA,YA)
```

which will convert the given coordinates (R,THETA) on the polar plot to actual coordinates (XA,YA).

```
      POLON(R,THETA)
```

to draw a line (CSType 1) from the current plot position to polar point (R,THETA).

```
      POLOFF(R,THETA)
```

to reposition the current plot position to polar point (R,THETA).

The notes pertaining to the equivalent cartesian routines apply equally to those for polar coordinates.

PART 3 - CONTOURING

Chapter 1 - Basic Contouring

1.1 - Introduction

DIMFILM includes a collection of subprograms for the production of basic contour plots. These may be used in conjunction with all other facilities of the overall system. As with the graphing routines, the user has considerable freedom. Error checking is performed and warning messages included, these being controlled through the check level. Users interested primarily in contour plots should first familiarise themselves with the earlier sections on basic plotting.

Contour plots may be freely intermixed with calls to the general plotting routines. DIMFILM attempts to ensure that on exit from any contour routine, all internal parameters are set to their values on entry. The beam is also positioned so that diagrammatic plotting may be resumed easily. Such properties as intensity, orientation and height of symbol strings, broken lines etc., will be maintained. It is possible therefore, to interrupt a diagrammatic plot to perform graphical plotting and then to resume. A few limitations are, however, necessary. Primarily, on return, a broken line pattern will commence at the beginning of its sequence (the consequence of a beam off movement). Symbol strings will commence with all their initial features active. If TXTCON has previously been called, it will be operable on all but the first subsequent calls to SYMTXT.

Except where otherwise indicated, all contour plotting routines are performed as CSType 1 operations and all textual annotation will be treated as CSType 2. Additional annotations and framing may be incorporated through use of the graphical annotation routines described in Part II.

At present, all DIMFILM contouring routines require regularly spaced input data. However, it is intended that fuller contouring facilities will be made available in due course.

1.2 - Contour Location

The contour plot is positioned in precisely the same way as graphing output, thereby permitting the use of the graphical annotation routines. Thus, the contoured data will be fitted into the current pre-clipping rectangle with appropriate margins allowed for titles. By default the sub-area will be centred as 83% of the pre-clipping rectangle in either direction. The contour grid will be scaled to fit this area. This enables the user to position the contour anywhere within the current bounds, and to generate multiple contour plots per frame (through respecification of PANE). Orientation of the contour plot is determined by the current mode of the device.

1.3 - Contouring Regularly Spaced Data

Contours are constructed based on a number of datum points. This section is concerned with contouring regularly spaced data. In this case a rectangular base mesh is provided with datum points at each intersection. The numeric data is passed via a two-dimensional real array GRID defined as:

```
      REAL GRID(NX,NY)
```

This defines a mesh of NX by NY elements, whereby the value of the (i,j)th element of the array GRID is the value of the mapping variable (e.g. for a physical relief map this would be the height) at the (i,j)th intersection of the base defining mesh. Pictorially, this is shown as:

```
--------------------------------------------------------------------------------


            (1,NY)                                           (NX,NY)
              *-------*------  . . . . . . -----*-------*-------*
              |       |                         |       |       |
              |       |                         |       |       |
            (1,NY-1)--*-------*- . . . . . . ---*-------*-------*
              |       |       |                 |       |       |
              |       |       .                 .       |       |
              .       .       .                 .       .       .
              .       .       .                 .       .       .
              .       .       .                 .       .       |
              |       |       .                 |       |       |
            (1,3)-----*-------*--- . . . . . . -*-------*-------*
              |       |       |                 |       |       |
              |       |       |                 |       |       |
            (1,2)---(2,2)-----*----  . . . .  --*-------*----(NX,2)
              |       |       |                 |       |       |
              |       |       |                 |       |       |
              *-------*-------*------  . . . ---*-------*-------*
            (1,1)   (2,1)   (3,1)                  (NX-1,1)  (NX,1)


--------------------------------------------------------------------------------
```

The rectangular mesh defined by the regularly spaced data will be mapped onto the contouring (i.e. graphing) sub-area of the current pre-clipping rectangle. That is, the (NX-1) intervals in the X-direction are exactly scaled to the contouring area X-dimension, while the (NY-1) intervals in the Y-direction are scaled to exactly fit the contouring area Y-dimension. Consequently, unless the proportion of the contouring area X:Y dimensions equals (NX-1):(NY-1) grid dimensions there will not be identical X- and Y-scales. If, for example, measurements have been conducted at different intervals in the X and Y directions it will be possible to reflect this in the definition of the pre-clipping rectangle (via PANE) to force equal scaling in the output plot.

For example, where the data is defined via GRID(6,3), and sampling was at equal intervals in both the X and Y directions, the pre-clipping rectangle should be redefined with its X-dimension a factor of 5/2 [=(6-1)/(3-1)] of its Y-dimension. {This assumes equal annotation margins - the default.}

Alternatively, GRID(7,4) with X sample interval half the Y sample interval reflects data from a square base mesh (6/3, but each X interval is half each Y interval), requiring a square contouring area; i.e. a pre-clipping rectangle with equal X and Y dimensions. {Same assumption.}

Note: DIMFILM routines are written to ensure the integrity of all arguments (except, obviously, the case of arguments provided for the return of information to the calling program; for example, inquiry routines). However, in the case of contouring from a grid of data the values passed in array GRID may be modified in their least significant bit only. This obviates the provision of scratch space to the routines, but should be borne in mind if subsequent calculations are to be performed on the data (the effect is likely to be minimal). This effect is not cumulative between successive calls to contouring routines.

1.3.1 - Basic Contouring

At the basic level, the user will have his regularly spaced data stored in an array and will know the values at which contour lines are required. The most basic routine will compute contour intersections with the mesh lines (via linear interpolation between mesh points) and generate specified contours. This routine is intended as a quick preview facility and produces contours that are straight lines between these points. The production of smoothed contours is described in the next section (Refined Contouring). By default, contour values (where possible) will be automatically included across the lines.

```
      CONTR(GRID,NX,NY,NC,CVALS)
```

GRID is a two dimensional REAL array of dimensions (NX,NY). This holds the datum values that define the rectangular mesh with NX columns and NY rows.

NX is the number of columns of data (NX>1).

NY is the number of rows of data (NY>1).

NC is the number of contour levels that it is desired to plot (NC>0).

CVALS is a singly dimensioned REAL array of dimension (NC) which holds the NC values at which contours are to be plotted.

The contoured area may be boxed in the same manner as for graphical routines; that is, by a reference to GRFRAM. Other titling or labelling may be applied as for the graphical routines.

1.3.2 - Refined Contouring

The user is able to control many defaults such as contour annotation and interpolation. This latter is used to generate smooth contours. Once a default is changed it will remain in effect until reset by the user.

The cubic interpolation of contours may be simply controlled by the user via:

CINTER(ISTEPS)

which will turn on the cubic interpolation of contours, where ISTEPS is the number of subintervals of the basic mesh that are to be used in evaluating the interpolating curve. (ISTEPS>0)

NOCINT to turn off this interpolation.

The contours plotted will be generated using a cubic interpolation. It should be noted that on some data forms unacceptable oscillations and inaccuracies may be introduced. This should be borne in mind when interpreting results. The number of steps at which the interpolating function is evaluated will affect the smoothness of the plots - but will also be reflected in the degree of computation involved. Clearly the size of the data grid will be a relevant criterion when setting this value, as will the physical size of the contour area in the frame.

The user has considerable freedom in specifying the way in which contour values may be plotted across the generated contours. The default is for such labelling to be produced wherever possible (proximity to an edge, or next contour, may prohibit production of these labels).

NOCLAB will turn off the plotting of contour values.

CLABEL will turn on the plotting of contour values.

CLABHT(HT,ANGLE)

The user may specify the character height at which contour values will be plotted and the orientation. The default character size is computed at 1/50 of the least dimension of the current pre-clipping rectangle (and may be further scaled if CQUAD or CxHALF is active), and values to be generated horizontally with respect to the frame.

CLABHT(HT,ANGLE)

will set the character height and orientation for contour values used in evaluating the interpolating curve.

HT is the character height in overall user units. (HT>0.0 sets height, other values will cause the default height to be effective)

ANGLE is the angle at which values will be produced, and is measured counter-clockwise from the positive X-direction of the frame (AGroup 1, default degrees).

The contour values that are output are, by default, produced according to the Fortran format specification G9.3. This format may be changed by the user. Additionally, in deriving the plotted contour value a scale factor may be applied to the actual contour values.

      CFMT(JFMT)

   will set the format for outputting contour values to that passed in JFMT (type CHARACTER, of length <11). This parameter should hold the required real format as a character string commencing with an open parenthesis and terminating with a close parenthesis. E.g. a valid parameter would be '(F10.3)'.

      CSCALE(SCALE)

   non-zero values of SCALE will be used as a factor by which all contour values will be pre-scaled before plotted as annotation.

Frequently, contour data may be symmetrical (for example, plots of some electromagnetic fields). In these cases, the user need supply only a quarter or half of the total pattern and the various contouring routines will fill out the whole area by a series of mirrorings.

```
--------------------------------------------------------------------------------


                        A-----------------S-----------------B
                        |                 |                 |
                        |                 |                 |
                        |                 |                 |
                        |                 |                 |
                        |                 |                 |
                        |                 |                 |
                        P-----------------Q-----------------R
                        |                 |                 |
                        |                 |                 |
                        |                 |                 |
                        |                 |                 |
                        |                 |                 |
                        |                 |                 |
                        D-----------------T-----------------C


--------------------------------------------------------------------------------
```

The total area that is to be contoured is given, in every case, by ABCD. The user may provide a data grid for either the upper-right quadrant, the upper half or the right half and have the complete area symmetrically contoured by appropriate reflection(s).

CQUAD will assume subsequent GRID data corresponds to the upper-right quadrant SBCQ, and reflect about both axes in turn.

CVHALF will assume subsequent GRID data corresponds to the vertical half SBCT, and reflect about ST.

CHHALF will assume subsequent GRID data corresponds to the horizontal half ABRP, and reflect about PR.

CWHOLE will restore subsequent GRIDs to accommodate the whole area ABCD (this is the default).

All the foregoing default modifiers should be referenced before any contour for which they are to take effect.

There are alternate forms of contouring routines, applicable to regularly spaced data, that may be used. These permit contours at equal intervals to be plotted or for closer control of the line colour/intensity for different contours.

CONTR2(GRID,NX,NY,NC,CSTART,CSTEP)

will generate contours at regular intervals.

GRID is the two dimensional REAL array of dimensions (NX,NY). This holds the datum values that define the rectangular mesh with NX columns and NY rows.

NX is the number of columns of data (NX>1).

NY is the number of rows of data (NY>1).

NC is the number of contour levels that it is desired to plot (NC>0).

CSTART is the value of the first contour to be plotted.

CSTEP is the value by which the contour value will be incremented for successive contours.

CONTR1(GRID,NX,NY,NC,CVALS,ICSG)

will generate contours at specified intervals, each being of independently set CSGroup colour/style.

GRID is a two dimensional REAL array of dimensions (NX,NY). This holds the datum values that define the rectangular mesh with NX columns and NY rows.

NX is the number of columns of data (NX>1).

NY is the number of rows of data (NY>1).

NC is the number of contour levels that it is desired to plot (NC>0).

CVALS is a singly dimensioned REAL array of dimension (NC) which holds the NC values at which contours are to be plotted.

ICSG is a singly dimensioned INTEGER array of dimension (NC). The absolute value of all elements in ICSG must be in the range [1,3]. The absolute value of ICSG(i) will be used as the CSGroup identifier determining the colour/style in which contour of value CVALS(i) will be plotted. When the actual value of ICSG is positive the corresponding contour will be labelled in the same colour/style.

```
CONTR3(GRID,NX,NY,NC,CVALS,ICLUT)
```

will generate contours at specified intervals, each being plotted according to a different colour look-up-table reference. All arguments, bar the last, are identical with those for CONTR1.

ICLUT is a singly dimensioned INTEGER array of dimension (NC). The absolute value of all elements in ICLUT is a reference to the colour look-up-table entry that will be used in plotting the corresponding contour and, when the actual value is positive, the value label(s) for that contour. Each element must have an absolute value not exceeding 255.

If the user requires even greater control of contour line/annotation style all the above CONTRx routines may be repeatedly called for the plotting of single contour lines, with change of colour/style between each. However, no account of adjacent contours can be taken when determining where annotations are produced.

PART 4 - DIMFILM Guidelines

The new DIMFILM has undergone a major rewrite. While rationalising certain routine names and converting to Fortran 77 the opportunity was taken to improve the internal operation of the product. Thus, all internal (i.e. inaccessible to user) routines now have the prefix DFX. A similar convention was applied to common blocks and internally accessed files. The user is therefore cautioned not to use the prefix DFX for any purpose associated with DIMFILM. Failure to respect this condition may well result in unpredictable results and the probable failure of DIMFILM.

Appendix 1 - Using DIMFILM at ULCC

DIMFILM is currently available on both the Amdahl 5890 and the Cray X-MP at ULCC. It is intended that the user interface as described in this document will not be subject to change. Please advise Graphics Development Group at ULCC of any problems with this product. We shall endeavour to fix reported bugs at the earliest possible moment. Similarly new features will be incorporated as they become available and will be documented by revisions to this guide.

It should be noted that some microfilm services (notably, colour film) may be subject to a charge. In these cases users must ensure that their intended usage is covered by an official college order. This should be directed to the Microform Section of the Operations Division (not Allocation and Control or Administration). The order may cover a single account number, a department or whole institution, and may indicate either a monetary limit or expiry date. Any enquiries regarding this should be directed to Claire Steward (extension 365).

1.1 - Amdahl 5890 - VM/CMS Service

The DIMFILM library is produced under VS FORTRAN Version 2, a Fortran 77 compiler available under VM/CMS. DIMFILM is built with the latest version of this compiler.

Under CMS the implementation of DIMFILM and its internal interface to the Dicomed film recorder makes use of virtual device addresses - these may be in the range 180-183 (inclusive), and so should be avoided by the user.

1.1.1 - VS Fortran Version 2

The DIMFILM library for use with VS Fortran is available on the GRAPHICS mini-disk. Access to this must be achieved by inclusion of an appropriate command:

```
GIME GRAPHICS
```

To access the DIMFILM library it is further necessary to incorporate a couple of GLOBAL commands. These, for regular users of DIMFILM, might best be included in their PROFILE EXEC file:

```
--------------------------------------------------------------------------------


                        GLOBAL TXTLIB DIMFILM VSF2FORT CMSLIB

                        GLOBAL LOADLIB VSF2LOAD


--------------------------------------------------------------------------------
```

A typical Fortran program would appear as:

```
                      PROGRAM TEST
                      ...
                      CALL D35
                      .. <Program including calls to DIMFILM routines>
                      CALL DIMEND
                      STOP
                      END
```

This should be compiled, loaded and executed in the normal manner. The GLOBAL commands will ensure the correct satisfaction of references to DIMFILM routines at the appropriate point.

The DIMFILM library is large, the memory required will depend on the modules referenced by the user.

When multiple libraries are used with DIMFILM it is possible that the default Loader table sizes may be exceeded - to avoid this, the default table size may be increased by including a command similar to the following:

```
SET LDRTBLS 14
```

Note: Users who are new to the Amdahl will find general documentation on running jobs in the Amdahl Handbook, published by ULCC. In particular see Chapter 2.2 for details of running jobs and Chapters 3.0 and 3.2.1 for details of using the FORTRAN compiler.

1.2 - Amdahl 5890 - MVS Service

The DIMFILM library is produced under Fortran 77 and made available for use with both VS Fortran and Siemens Fortran 77. Separate libraries are maintained for use with these compilers. DIMFILM is built with the latest version of each compiler, and so all jobs should follow the JOB statement with

```
/*JOBPARM P=PROC02
```

1.2.1 - VS Fortran

The DIMFILM library for use with VS Fortran is available on SYS2.DFILMVS.LIB, and should be used in conjunction with release 5.0 of VS Fortran, which is obtained by immediately following the JOB statement with

```
/*JOBPARM P=PROC02
```

To access the DIMFILM library LLIB1='SYS2.DFILMVS.LIB' may be used in the FVS procedure. A simple job to produce output on 35mm black and white film might be :

JCL

```
    //<jobname>   JOB  '<name and delivery>',REGION=2000K
    //*PASSWORD <MVS password>
    /*JOBPARM P=PROC02
    //         EXEC    FVSCLG,LLIB1='SYS2.DFILMVS.LIB'
    //C.SYSIN  DD  *
          PROGRAM TEST
          ...
          CALL D35
          .. <Program including calls to DIMFILM routines>
          CALL DIMEND
          STOP
          END
    //
```

Phoenix 3

```
    //<jobname>   JOB  '<name and delivery>',REGION=2000K
    //*PASSWORD <MVS password>
    //           EXEC   PHOENIX3
    FVSCLG PROGRAM=%H+ LIBRARY=SYS2.DFILMVS.LIB
          PROGRAM TEST
          ...
          CALL D35
          .. <Program including calls to DIMFILM routines>
          CALL DIMEND
          STOP
          END
    +
    //
```

The DIMFILM library is large and it will almost certainly be necessary to request more than the default memory allocation using the REGION parameter on the JOB statement. The default is 600K and the maximum is over 7000K, although no more than 6500K should normally be requested. (Numbers are decimal and specify units of 1K (1000) bytes).

When multiple libraries are used with DIMFILM it is possible that the default Linkage Editor/Loader table sizes may be exceeded giving an MVS IEW0484 TABLE OVERFLOW error. To avoid this, the default table sizes may be increased by including the following parameters in the appropriate JCL EXEC statement:

```
Using the Linkage Editor:
      LPARM='SIZE=(2000K,64K)',LREGN=2000K
or using the Loader:
      LPARM='SIZE=2000K',LREGN=2000K
```

1.2.2 - Siemens Fortran

The DIMFILM library for use with Siemens Fortran is available on SYS2.DFILM77.LIB. This may be accessed by utilising the LLIB1 parameter in the F77 procedure. It is recommended that Version 4 of Siemens Fortran be used - considerable savings in CP time and memory can be achieved. A simple job to produce output on 35mm black and white film might be:

```
//<jobname>   JOB  '<name and delivery>',REGION=2000K
//*PASSWORD <MVS password>
/*JOBPARM P=PROC02
//     EXEC    F77CLG,LLIB1='SYS2.DFILM77.LIB'
//C.SYSIN  DD  *
      PROGRAM TEST
      ...
      CALL D35
      .. <Program including calls to DIMFILM routines>
      CALL DIMEND
      STOP
      END
//
```

```
Phoenix 3

    //<jobname>   JOB  '<name and delivery>',REGION=2000K
    //*PASSWORD <MVS password>
    //          EXEC   PHOENIX3
    F77CLG PROGRAM=%H+ LIBRARY=SYS2.DFILM77.LIB
          PROGRAM TEST
          ...
          CALL D35
          .. <Program including calls to DIMFILM routines>
          CALL DIMEND
          STOP
          END
    +
               //

--------------------------------------------------------------------------------
```

The DIMFILM library is large and it will almost certainly be necessary to request more than the default memory allocation using the REGION parameter on the JOB statement. The default is 600K and the maximum is over 7000K, although no more than 6500K should normally be requested. (Numbers are decimal and specify units of 1K (1000) bytes).

When multiple libraries are used with DIMFILM it is possible that the default Linkage Editor/Loader table sizes may be exceeded giving an MVS IEW0484 TABLE OVERFLOW error. To avoid this, the default table sizes may be increased by including the following parameters in the appropriate JCL EXEC statement:

```
--------------------------------------------------------------------------------

                     Using the Linkage Editor:
                           LPARM='SIZE=(2000K,64K)',LREGN=2000K
                     or using the Loader:
                           LPARM='SIZE=2000K',LREGN=2000K

--------------------------------------------------------------------------------
```

Note: Users who are new to the Amdahl will find general documentation on running jobs in the Amdahl Handbook, published by ULCC. In particular see Chapter 2.2 for details of running jobs and Chapters 3.0 and 3.2.1 for details of using the

```
--------------------------------------------------------------------------------
```

FORTRAN compilers.

1.3 - Cray X-MP - UNICOS Service

The DIMFILM routines are available as a library, compiled with the CFT77 compiler and accessible on /lib as libdimfilm.a. The normal usage will involve only the declaration of dimfilm as a library; either to segldr or cf77, via the -l option. There is no need to use any other graphics library in order to produce output on film. A simple job to produce output on 35mm black and white film might be:

```
                cat test.f <</EOF
                      PROGRAM TEST
                      ...
                      CALL D35
                      .. <Program including calls to DIMFILM routines>
                      CALL DIMEND
                      STOP
                      END
                /EOF
                cf77 -l dimfilm test.f
```

alternatively, the last command may be replaced by:

```
                cft77 test.f
                segldr -l dimfilm test.o
```

The DIMFILM library is large; the required memory will depend on the modules acccessed by the user program.

Where plot data is staged via the front end, an indication of file size and frames generated will be given, but confirmation of successful staging is a function of the front end.

Note: Users who are new to the Cray will find general documentation on running jobs in the Cray Handbook published by ULCC. In particular see Chapter 2.2 on

----

running jobs and Chapter 3.1 on the use of the CFT77 FORTRAN compiler.

Appendix 2 - Graphics Devices

There are many types of graphic output devices for the display of computer generated images. These may differ significantly in the way they generate graphical output, and this may have considerable effect upon the visual appearance of pictures (which may look very different when directed to alternative devices). This can be very confusing. An understanding of the basic principles of the operation of a device is essential in order to be able to see why it produces the results it does. There may also be significant differences in the functioning of devices that affect their suitability to certain applications, or that dictate alternate modes of operation for efficient utilisation.

2.1 - Image Appearance

Of paramount concern to the user of any graphics output device is the actual appearance of the picture - and how it may differ from that expected. This section deals with the makeup of the image in its final perceived form and how different devices may present the same data. A generalised description of picture production is followed by consideration of different device classes.

2.1.1 - Colour Generation

[Note: Terminology in this section should not be interpreted rigorously or pedantically; the intention is to introduce the inexperienced user to the complexities of colour mixing, etc. Consequently, terms such as colour and hue are used here in a general sense, and are interchangeable.]

There are, essentially, two ways in which a colour image may exist; it may be permanent (i.e. hardcopy) or transient (e.g. electronic image on a cathode ray tube - CRT). Additionally, a permanent image can be created directly or derived from a transient image (or from a combination of several). All these methods of image formation result in subtle (and not so subtle) differences in the final product.

Transient images are typically produced through emission of (coloured) light from a luminous source. Indeed, the CRT relies on the eye/brain physiological interpretation of colour whereby any hue may be created through the admixture of appropriate proportions of the three primary colours; these being, red, green, and blue. Equal combinations of all three (saturated) yields white, while varying the proportions of each independently will create any other colour. Red, green, and blue, then, are known as the additive primaries (or simply as the primary colours/ primaries). Mixing the saturated primaries in equal proportions gives the following results:

---

none	==> Black
Red + Green	==> Yellow
Green + Blue	==> Cyan
Blue + Red	==> Magenta
Red + Green + Blue	==> White

It is the application of luminous sources of the additive primaries that is fundamental to the colour CRT (and television). On the colour CRT, red, green, and blue potential light sources are located in close proximity to each other, where they may be independently excited in varying proportions. Each such cluster (or triad) of red/green/blue will then be interpreted by the viewer (at sufficient distance) as a single source of the hue associated with that combination. Such spatial integration will be referred to again. {This is why colour graphic displays are often referred to as RGB. They accept separate signals for each of the Red, Green, and Blue components. The standard broadcast television transmission technique relies on a different convention, which is briefly treated in the discussions on colour models in the main part of the User Guide.}

As the three primaries are independent (i.e. none of them can be formed of any combination of the other two), it will be seen that the hues resulting from the combination of any two primaries have the property that they contain no component of the third primary. Thus, for example, cyan (which is the combination of green and blue) has no trace of red. That is, a cyan pigment or dye will filter white light (the combination of red + green + blue) of all red. For this reason cyan, magenta and yellow are known as the subtractive primaries (or, alternatively, as the secondary colours/secondaries). The value of this will become evident in the later discussion.

The relationship between the additive and subtractive primaries can be clearly shown by a diagram of the hue spectrum:

```
                                       *
                                    -     -
                   green  120.0  -           -  60.0  yellow
                               *               *

                             -                   -

                           -                       -

                          -                         -
             cyan  180.0 *        hue spectrum       * 0.0  red
                          -                         -

                           -                       -

                             -                   -

                               *               *
                    blue  240.0  -           -  300.0  magenta
                                    -     -
                                       *
```

Permanent images may either be reflective or transmissive. The former encompasses all forms of hardcopy on an opaque base (e.g. paper) while the latter is the province of transparencies requiring projection. However, if one considers an overhead transparency this may be viewed either with back-lighting (i.e. analgous to projection) or it may be placed on a reflective (e.g. white) background and viewed under incident lighting. This is a good analogy to the effect of transmissive ink on the printed page; the coloured transparency functions by filtering the incident light and transmitting only a particular colour through the clear base medium, this is reflected (in this case) by the white base and retransmitted through the transparency with (ideally) no further filtration. It is reasonable, then, to direct attention first to hardcopy output and to look at the ways in which coloured inks are used by the printer (these techniques are relevant to all present forms of colour hardcopy device).

The printed colour on a page is the result of either reflective or transmissive dyes/pigments and the way they are combined on the page. The first category relies on an essentially opaque pigmented ink (or paint) which relies on reflection of the required colour and absorption of all others. Mixtures of such inks will, when applied to the paper, yield a colour that corresponds to the proportional mix of those yielded by each ink individually (perfect mixing is assumed throughout this discussion). If two such inks are applied sequentially to the paper (the first allowed to dry thereby precluding mixing) the resultant colour will be that of the last used ink (in reality the opaqueness of a paint or ink can result in some modification of the perceived hue). In essence, for such inks the colour is dependent on the particular properties of a single ink. This is the method of colour creation used in oil painting - the pigments are mixed additively and then applied to the canvas. However, it is well worth noting that the artist has a different colour requirement than the physicist or photographer. These two are intent on capturing the real world (theoretically or photographiclly, respectively). For them the hue spectrum based on the red, green, blue model (and, thus, cyan, magenta, yellow) is an adequate representation of light (i.e. the combination of various monochromatic frequencies). The artist is dependent on displaying colour through the removal of various combinations of monochromatic frequencies from the incident light through selective absorption by the pigments in his paints. In fact, the standard artist’s palette comprises red, yellow, and blue as his primaries with corresponding secondaries orange, green, and violet. White is now the absence of any colour, and a separate pigment is specifically needed for black. The artist also relies on a white pigment to lighten any of the hues. It would not be possible to use the additive primaries as the mixture of any two would (theoretically) remove all colour and yield black. In practice, red and blue paints result in violet due to the pigments used.

---

none	==>	White (canvas)
Red + Yellow	==>	Orange
Yellow + Blue	==>	Green
Blue + Red	==>	Violet
Red + Yellow + Blue	==>	Olive-Brown
Black	==>	Black

Turning to transmissive dyes/inks, a single such ink will transmit its own colour (absorbing others) and this may then be reflected from the base material and re-transmitted. Such inks are translucent. If two (or more) are overlaid, the first will transmit its own colour which will then pass into the second (lower) ink, which will in turn transmit only its own colour. Thus only that subset of colours that is common to both inks can be transmitted through to the base, where it will be reflected and re-transmitted. Each ink has subtracted (i.e. removed, or filtered) particular colour components from the incident light. From the earlier consideration of the relationship between the primary colours and their secondaries, it should be fairly obvious that these latter colours are ideal candidates for the printer’s palette. Clearly, the additive primaries would be no use - any combination would bar all transmission (each will pass only its own component). However, the subtractive primaries will be seen to be ideal as each will remove only its complementary primary and transmit the other two.

---

none	==> White
Yellow + Cyan	==> Green
Cyan + Magenta	==> Blue
Magenta + Yellow	==> Red
Yellow + Cyan + Magenta	==> Black

Note: it must be remembered that these are transmissive, not reflective.

In actuality, most printing processes based on the subtractive primaries employ a fourth ink for perfect representation of black. {Theoretically, yellow + cyan + magenta yield black, whereas the mixing of printer’s inks tends to produce a rather dark brown.}

The foregoing description of hardcopy colour has been on the basis that each point in the picture is coloured by application of one or more inks to that point. As the standard printing process is not suited to delivery of varying quantities of each ink to a point (rather any point either has none or a fixed amount of each ink) the subtractive process would be capable of printing only eight discrete colours (being, yellow, cyan, magenta, green, blue, red, black and white). Clearly, much printed material requires rather more than eight hues. This may be achieved through different techniques.

If a standard newsprint picture is studied closely it will be seen that this is composed of a vast number of discrete points. For a black and white picture the points are dispersed with varying size. Thus in a given area there will be a greater density of ink where the dots are largest, while those with decreasing size will have less ink density over an area. When viewed from a normal reading distance this variation in dot size, and hence ink density, will be perceived as varying shades of grey - between black and white (assuming white paper). This techniques is known halftoning. The print screen may be around 60 to 80 dots per inch for newspapers, up to 150 dots per inch in quality magazines and books.

For colour printing a similar technique may be applied. The master image (e.g. film) is passed through a process known as colour separation to yield four colour masters (i.e. yellow, cyan, magenta, black). These are screened and colour plates are produced for each colour; the plates now consisting of single tone images with areas of different dot size (and so different colour density). The final step is for the final image to be printed from each plate with its corresponding ink (preserving registration between each pass). This is basic four- colour lithography. {Note, that as overlapping colours are used this requires the use of translucent inks.}

These techniques are all relevant to computer generated graphics (and may be applied to transient images, also). Here the problem is that the dot size is likely to be fixed, as is the display resolution. Considering monochromatic images first, halftoning may be simulated if four pixels (i.e. an array of 2x2 dots) is used to represent each point in the image. With a bi-level device (i.e. monochromatic with no intensity modulation - black or white, only, at any pixel/dot) the 2x2 array gives the option to set any number of the four points - from none to all four - as black, thereby giving 5 levels of dot density. If this is viewed from sufficient distance (i.e. the pixels are sufficiently small) the perceived effect is that the bi-level device is now showing 5 levels. If the device actually offered 4 levels (zero to three) then a 2x2 arrangement could have a summed value of zero to twelve and the perceived range would be extended to thirteen. This process - equivalent to screened half-tones - is known as dithering, dots of different intensity/colour being grouped to create the illusion of a greater range (due to spatial integration). When the device is a colour display/printer with only three primaries available then a 2x2 array yields 125 different colours/shades, because for each primary, 5 intensities may be achieved. It is worth noting that dithering can be applied to opaque (i.e. purely reflective) paints, although the range will then less as each pixel can only be one of the paints (assuming no mixing). Dithering 3 such colours over a 2x2 array yields a total of 34 different colour combinations (compare with 125 for subtractive inks). {The present paragraph has, up to now, assumed that the image data is of lower resolution than the display so multiple pixels (dots) may be assigned to each data point in the image. Where the image and display are of the same resolution the same techniques may be applied to increase the intensity/colour range of the device through ordered dithering. A decision, based on the actual address of each point in the image and its intensity, is made as to how it should be represented in the display. For areas of constant value the appearance will be as for a similar halftone pattern, and the effect will only be noticed in areas of changing colour/intensity.}

2.1.2 - Device Categories

In general, drawing may be performed as line drawing (calligraphic) or painting (area fill). Computer graphics uses a terminology which reflects the technology used. Thus line drawing is referred to as vector plotting (a straight line is plotted between two endpoints), while painted - or shaded - images are termed raster plots when produced on particular output devices (which construct a picture through a matrix of discrete points). Although specific devices may be better (or only) suited to one mode of drawing there are usually ways in which both modes may be represented (or, at the least, approximated).

Pen plotters These are the closest device to the draughter’s pencil and drawing board. Here it is generally the case that lines are drawn in the same order as they are generated by the computer program. Thus overlapping lines (or areas of colour) will be drawn as such and the effect will depend on the colours used and the type of ink; with felt-tips, and other pens with translucent ink, line intersections will be noted for their change in colour as for subtractive processes (if the additive primaries are used intersections will - in theory -

be black). Pigmented, opaque inks will leave intersections of lines reflecting the colour of the last drawn line. {The same applies to attempts to shade areas with multiple, parallel strokes of the pen.}

Raster CRT displays are considered next, due to their relevance in discussions concerning other graphic output devices; although they generate transient images rather than the permanence of hardcopy output. The images produced on such devices are arrays of points (the red/green/blue triads) known as pixels and (normally) spaced regularly across the display screen with regularly spaced lines of pixels. The whole image is stored in the display memory with a triplet of values stored for each pixel to represent the red, green, and blue intensity levels. Maximum intensity n all three combined will yield white. Values are assigned to each pixel affected as graphics instructions are processed - the order is crucial. Output vectors are evaluated for their point of intersection with the scanlines, and those points assigned the colour of that vector (a process known as rasterisation). In assigning colour (similar arguments apply to monochromatic displays) to any point there are two basic options: the point may be assigned the colour of the most recently drawn line, or any value already at a point may be merged with that of this vector. The first option is equivalent to the use of opaque inks, while the second has other properties. Although merging of colours is analgous to additive mixing it is not identical. The merging of each component may be additive (although a maximum for each component may not be exceeded) or by taking only the maximum contribution to each component from the intersecting lines. Using the notation (r,g,b) to be the value of a colour triplet some examples of merging can be given.

```
                              Additive overlay

                     line1           line2                result

            i) (0.4,0.5,0.0) + (0.6,0.0,0.0) ====>  (1.0,0.5,0.0)

           ii) (1.0,0.5,0.0) + (0.6,0.0,0.0) ====>  (1.0,0.5,0.0)

          iii) (0.7,0.6,0.1) + (0.7,0.1,0.8) ====>  (1.0,0.7,0.9)


                              Combined overlay

                     line1           line2                result

            i) (0.4,0.5,0.0) + (0.6,0.0,0.0) ====>  (0.6,0.5,0.0)

           ii) (1.0,0.5,0.0) + (0.6,0.0,0.0) ====>  (1.0,0.5,0.0)

          iii) (0.7,0.6,0.1) + (0.7,0.1,0.8) ====>  (0.7,0.6,0.8)


                               Opaque overlay

                     line1           line2                result

            i) (0.4,0.5,0.0) + (0.6,0.0,0.0) ====>  (0.6,0.0,0.0)

           ii) (1.0,0.5,0.0) + (0.6,0.0,0.0) ====>  (0.6,0.0,0.0)
          iii) (0.7,0.6,0.1) + (0.7,0.1,0.8) ====>  (0.7,0.1,0.8)

--------------------------------------------------------------------------------
```

It can be seen that the different techniques give very different results. In no case can the maximum intensity of a component be exceeded so, for example, multiple maximum reds (in additive overlays) would always result in maximum red. {A further complication may occur when the device drives the CRT through a look up table. In this case, the values in the device (pixel) memory are index pointers into look up tables; consequently, it can be possible to combine - logically OR - the overlaying LUT pointer values (in binary arithmetic) yielding a new pointer with the output colour having no relation to the colours held in the look up table for either of the two overlapping colours.} Filled areas may be handled similarly, but often a hardware fill is utilised which will colour all points of the area with the required colour in an opaque mode. Similarly, pixel scan line data passed in graphics data will normally overwrite any existing pixel values (no merging). Subsequent vectors, however, will be handled in whichever mode is currently supported/active. For most purposes the display may be considered as erasable, with the most recently interpreted graphic instruction taking precedence of visibility.

Vector refresh displays rely on buffering plot commands into device memory (as a display list) from where they may be continually scanned and displayed on a long persistent phosphor CRT. This is not the line-by-line scanning of a raster device; each directive is traced across the CRT as a vector (in any direction). There is a limit to the number of directives that may be held and cycled through in the display list before flicker becomes noticeable. Overlapping vectors will all be traced and the effect will be the repeated intensification of the overlapping parts of the image. (Refresh displays are generally monochrome - although bi-colour devices are available.)

Storage tubes, more correctly known as direct-view storage tubes, are in the category of "additive" displays. Each plot instruction remains visible until the whole display is erased. As an instruction is "plotted" it is written by an electron beam on a fine dielectric mesh behind the phosphor on the CRT; a flood gun continually transfers this image onto the CRT. The image on the wire mesh is additive, but the range of viewable intensities is small so overlapping plot elements show little hotspotting - the most intense of the overlapping elements will predominate.

Non-photographic, colour hardcopy devices include many types of output technology. These may include electrostatic plotters, ink jet plotters/printers, thermal plotters/printers (e.g. wax/dye transfer), inked ribbon systems (matrix printers), and laser printers. Each of these is characterised in the way the picture is produced. Whether it is a single or multi-pass operation, the image is drawn by a series of closely spaced lines composed of many points (which may overlap). The resolution of these devices is expressed in points per scan line unit and lines per unit measure (sometimes points per unit area are given). Typical resolutions are between 100 and 560 points per inch (although some ribbon printers may be lower). For all these devices a whole image is (notionally) created as an array of all pixel points with a colour/intensity value at each. Thus the graphics instructions are pre-processed before plotting (that is, rasterised) to generate the appropriate array of pixel data, which is then output line by line. (For colour applications, it is generally the case that three - or four, if a separate black is provided - complete passes of the array are necessary, one for each ink. This requires careful registration of the plot medium between passes to ensure that scan lines overlap correctly. To avoid this some plotters have a complex mechanism to enable a single pass across the image area; some ribbon printers shift the ribbon and overstrike each line before advancing the paper, and some ink jet devices perform a similar one pass action switching between their jets.) For bi-level monochrome plotters one of the dithering techniques may be employed to give multi-level halftone pictures. For colour plotters, most employ subtractive inks and do not need to rely on spatial integration for the basic mixing of colours. Dithering may be used to increase the colour range, although this will often depend on the source of data (i.e. the rasteriser). This source may be a raster display system feeding the hardcopy RGB signals at its display resolution and limited to a small colour range (by virtue of the number of pixel planes); such signals must be mapped onto CMY space for the subtractive process. In rasterising data there is one further complication - efficient rasterising, without a large buffer, will require pre-sorting of vectors. This may mean that vectors are scan converted in an order different from that in which they were generated by the user. However, this problem is less likely to occur with the large capacity rasterisers now associated with many hardcopy units (or with the raster displays used to buffer the data for lower resolution devices).

Film recorders are of two sorts. Both function by recording, on film, the trace of a beam on a precision CRT through an optical system. Those with a vector capability (i.e. a vector may be drawn on the CRT) function in an additive manner. The shutter remains open and film is exposed continuously as the graphics data is interpreted and plotted on the CRT. Such film recorders may be considered as active. This image (on the CRT) is transient and is not refreshed in the way the display of a raster graphics terminal (or television) is. The whole image is not built up and then captured, rather each element is recorded as it is created (and the CRT image of that element would appear to the eye as no more than a spot, of varying intensity, traversing the shape of the element). This leads to an additive or cumulative exposure on film - multiple intersections of vectors, or overlapping solid fill areas, will result in an exposure equivalent to the sum of all the coincident intensities. Indeed, the nominal device maximum may be exceeded many-fold at any point. The effect of this then becomes a function of the physics/chemistry of film. Essentially, film can only be rendered clear (reversal stock) by maximum exposure. However, an effect known as halation then becomes apparent and the ’bright’ point/area leeches into surrounding areas with a fuzzy appearance. Colour film is produced on film recorders by splitting each element to be drawn into its constituent colour components, which are then plotted separately onto the CRT (near-white spot) and recorded through the corresponding colour filter (normally red, green, and blue are used). Generally the whole image is pre-streamed for each individual filter thereby minimising filter changes. With colour, multiple exposures through any filter will be additive and once a colour is fully saturated on film further exposure may adulterate the image to the extent that not only halation occurs but the colour will be (ultimately) washed to white (all colour layers clear, on reversal stock). The additive nature will be seen dramatically where two maximum intensity primaries overlap (thus, red and green lines crossing will show a very predominant yellow point of intersection). This gives problems when a coloured background is required behind a vector picture; colours must be selected so their combination results in the required vector colour! The other class of film recorder handle only raster data. This may be through pre-processing (i.e. rasterising) via software on a host computer, or they may fed RGB signals from a graphics raster display (which then acts as the rasteriser). The image is recorded on film line by line, and it is useful to consider such film recorders to be passive. This type then will function as a raster device, and generally the last plotted entity will determine the colour at any point.

Appendix 3 - Device Specifications

This Appendix describes the characteristics and capabilities of those devices supported by DIMFILM. As additional devices are made available in DIMFILM their descriptions will be added here. The previous Appendix should be consulted for comparison of various device categories in so far as their mode of operation affects the appearance of graphical output.

3.1 - DICOMED Colour Film Recorder D(1)48C

The Dicomed D48C graphic COM system is a high precision colour film recorder. In D148C configuration it includes a Digital Equipment Corporation controlling mini-computer. The name D48C refers to the graphic recorder, comprising firmware controller, precision CRT, film transports and optical assemblies.

The D48C is capable of image recording in vector, raster and alphanumeric modes, and may be used for either colour or monochrome recording dependent on film/optics used. It is ideally suited to many applications, including scientific plotting, business graphics (vector or raster), animation, engineering and design, and alphanumeric COM (Computer Output Microform, e.g. microfiche output - which may incorporate other graphics).

As a film recorder the Dicomed D48C captures the image of a variable intensity spot as it traverses a precision cathode ray tube. This CRT uses the P48 phosphor (wide spectral response), and colour is achieved by insertion of coloured filters in the optical path. For each colour filter the corresponding components of the image are traced - that is, recording is in an additive manner (filters being used singly, not in combination). The filter assembly comprises seven colours (red, green, blue, yellow, magenta, cyan, and neutral) and is used in conjunction with specially designed colour corrected optics. (Note: colour output at ULCC is currently produced by accessing the red, green, and blue filters alone and streaming the plot data accordingly.) For use with black and white film the neutral filter may be utilised; however, for the highest resolution in black and white on the aperture format a separate narrow spectral response lens is used (without filter assembly). Three optical assemblies are in use at ULCC for graphic COM: F300 for 16mm cine (colour and black and white), F304 for 35mm slide (i.e. comic - colour), and F309 for 35mm aperture card format (black and white).

The film transport (F333A - manufactured by Marron Carrell), which sits atop the optical assemblies, is able to handle 16mm (perforated) and 35mm (both perforated and unperforated) with 400 feet film magazines. This is accomplished through changes to the transport mechanics and use of appropriate aperture plates.

In vector mode , the D48C has an addressability of 32,768 x 32,768 upon the CRT (the optics/film format will result in a [maximised] subset of this being available for each film format). The D48C is capable of very fast vector plotting: a minimum length vector (single point) will take 15 microseconds, while a full-screen, maximum intensity vector can be drawn in 45 milliseconds (single pass). The time required is roughly proportional to the vector length. Exposure may be at any of 256 levels (for any one filter). In drawing a vector, the D48C deflection system is used to accurately position the beam at selected points on the 32K x 32K matrix when the exposure logic turns the beam on for the requisite time. The points are selected so that the vector is drawn as a continuous and uniform intensity line (through ensuring adjacent points touch).

In raster mode , the D48C is able to plot on a matrix of 4096 x 4096 addressable points (pixels - i.e. picture elements) on the CRT (again, a subset is available to each film format) at any of 256 exposure values. The difference between vector and raster addressability is due to the spot size generated; the raster spacing ensures a continuous image through the touching of adjacent points. The Dicomed provides the user with the means to construct display pixels as composite points with control over point and element spacing. This enables other raster "resolutions" to be imaged. Point plotting in raster mode may be accomplished at speeds of 100K-250K points per second, enabling maximum resolution raster images in full colour (three passes) to be plotted in around 3 minutes. The maximum flyback time for repositioning the beam to the next scan line is 100 microseconds; for certain applications a short end of line delay of (about) 15 microseconds may be introduced.

In point mode , the D48C may be positioned to any of the CRT’s 32,768 x 32,768 addressable points, which may then be exposed at any of 256 levels. The maximum time to position to any point is 100 microseconds, while exposure may be from (about) 4 to 9 microseconds.

In character mode , the D48C has a high speed hardware character generator. This is particularly relevant to alphanumeric COM applications. The character fonts are stored in local memory, and depending on complexity a number of fonts may be accommodated simultaneously, with less than 50 milliseconds required for font changes. Characters may be generated at different sizes and in four orientations, as standard. Speed depends on font complexity, but for typical alphanumeric COM applications speeds in excess of 30,000 characters per second may be achieved. (In practice, throughput is dependant on input rates to the D48C.) The hardware character generator functions at only one exposure level and differs from other drawing modes through operating as a stroke generator. Each character is drawn at high-speed as a series of short vectors and the CRT beam is unblanked throughout the duration of each stroke.

Intensity/exposure control is over an 8-bit (i.e. 256 level) range. However, this range is selected according to the plot mode in use: vector, raster, point, and character modes all access different CRT intensity scales in order to achieve compatible film exposure when modes are intermixed on a frame. [It should be noted that hardware characters may only be generated at a single exposure level.] The D48C incorporates further internal exposure tables that are selectable according to the optics and emulsion characteristics to ensure consistent film exposure under the different operating conditions. Separate table references are made according to the selected filter. These tables are programmable so that alternative emulsions may be used and the system calibrated accordingly. Indeed, calibration is required for each film type and for each optical assembly with which it may be used: this calibration is for each plotting mode and (for colour assemblies) each filter. In vector mode the internal intensity value is dynamically compensated according to the angle at which a vector is drawn. The exposure technique is time modulation which controls the density of individual points directly through the duration for which the CRT is excited. This ensures precise control and a high degree of repeatability with consistent line width.

The geometrics of the D48C offer a very high precision with minimal distortion and a high degree of repeatability. With regard to the major axis, trapezoiding , rectangularity , linearity , and line curvature (or pin cushion distortion) are all maintained to less than (plus or minus) 0.1%, while orthogonality of the horizontal and vertical axes is within (plus or minus) 0.15%. The spatial repeatability is within (plus or minus) 0.02% over a ten-minute interval, and (plus or minus) 0.05% over a thirty-minute interval (after a one-hour warm up).

The photometrics of the D48C offers a dynamic exposure range of 256 levels. In terms of diffuse density units (D), the range on a typical panchromatic film (e.g. Kodak Plus-X) will be 1.5D when processed to gamma 1.0 (or 2.0D for gamma 1.5). The exposure uniformity over the entire plotting area has a maximum deviation of (plus or minus) 0.15D (measured through a circular aperture of 3 millimetres diameter).

The film resolution is a function of optics and film type. Measurements are in line pairs per millimetre and are referenced to black and white film. For the optics in use at ULCC the target resolutions are:

Optics	resolution (lpm)
F300 (16mm)	90
F304 (35mm slide)	40
F309 (35mm aperture)	55

[It is important to appreciate that these are the resolutions that may be achieved at the film plane on high resolution film, such as DACO E. It is possible to resolve 3000 lines across the image area on the CRT of the D48C. The 4K raster of the D48C is designed so adjacent lines merge and a continuous image is perceived.]

Registration accuracy on the F333A film transport is maintained within (plus or minus) 0.005 millimetres. This is of significance in generation of cinematographic film and is adequate for producing steady images.

In practice, measurements on the D48C at ULCC (for geometrics, photometrics resolution and registration) have consistently indicated performance well within these limits.

There are a number of aspects concerning use of the D48C that are generally applicable to all devices. These are dealt with in the following sections, after which each film mode is considered separately (i.e. these being the different devices supported on the Dicomed).

3.1.1 - Prime

An image is recorded from the excitation of phosphor on the CRT (through a controlled exposure of an electron beam). After the excitation - and associated luminescence - of the phosphor it must stabilise at a neutral level. If this stabilisation is not allowed to occur the next phosphor excitation may result in greater luminescence. That is, the exposure on the film may be greater than that desired. This effect may be observed as the "ghosting" of earlier images superimposed upon the current picture, and is known as image retention . It occurs most commonly on raster images (particularly in those areas of relatively low intensity) and is more usually noticed on grey scale images (on black and white film). The retained image is often a previously drawn vector picture or components from several. Usually lines drawn at high intensity will be subject to retention, although repeated tracing across the same area of the CRT [i.e. identically positioned vectors, which might be user data or the ULCC identification frame(s)] may also be retained.

Consider the case of a high intensity vector (redrawn several times) followed by a low intensity uniform raster. Each point in the raster image will be subject to the same length of electron beam excitation on the CRT, and it would be expected that even exposure of the film across the raster would result. However, those areas of the CRT on which the vector was drawn may not have relaxed to the appropriate neutral level. At those points although excited for the same length of time phosphor luminescence will be at a different level than elsewhere: the result is a uniform raster with a vector superimposed. The effects on vector images are virtually unnoticeable. This is due partly to the very small overlap of vectors which might exhibit retention (intersections of line width slightly enhanced would not be perceived), but mainly to the fact that vectors are produced via higher exposures than rasters (dictated by relative visual perception) and so retention effects are proportionately lower. Similarly, with colour rasters exposures are at a higher level than for black and white due to film characteristics and spectral filtering (i.e. the filter cuts the transmission of phosphor luminescence considerably; conversely a longer exposure is required for a similar effect on film). Again, the effect may be observed on low intensity colour rasters. Similarly, high intensity raster images occasionly result in noticeable image retention on subsequent lower intensity rasters. The effect is not confined to adjacent images and may become apparent some considerable time later. Image retention is dependent on phosphor characteristics (in the case of the D48C this is P48). At ULCC the Dicomed is periodically primed ; that is, the CRT is saturated with a uniform high intensity raster. This reduces image retention to a very high degree. However, image retention may become a problem during graphic generation from stacked data (the method by which ULCC combine the plot data of many individual users into a single, continuous input to the Dicomed). To help obviate this DIMFILM users are provided with a means to prime the CRT during their plot run.

```
      DPRIME(<device>)
```

- this will cause the CRT to be primed for the designated device, provided it is currently active. If the display surface is non-empty it will be cleared before the prime is initiated. The frame will always be advanced after the prime. The single parameter is of type CHARACTER and should be a string comprising the device designation:

’D35’ - will prime device D35 (35mm black and white aperture card format)

’D35P’ - will prime device D35P (35mm black and white portrait mode format)

’D16’ - will prime device D16 (16mm black and white movie)

’D35C’ - will prime device D35C (35mm colour slide)

’D16C’ - will prime device D16C (16mm colour movie)

’ALL’ - will prime all active Dicomed devices

Specific references exist for each individual device if such mode is preferred (these being detailed under the section for relating to each device).

It should be noted that priming will expose a frame of film (the film transport having no shutter). Thus, priming is inappropriate to cinematographic output except preceding sequences.

Most users will not find image retention a problem. The facility is included for users who encounter such effects. Priming is a relatively slow process and excessive use could degrade throughput of the Dicomed system. The length of this section is not intended to emphasise the problem, but to explain and reassure users when it is first encountered.

3.1.2 - Frame repetition

In certain applications it is useful to repeat (either singly or multiply) the generation of each successive frame. In particular, for animations which require twnety-four images for every second of screen time (twenty-five if destined for PAL video) great economies of computation may be made by "double framing", i.e. repeating each image. Thus for every second only twelve images will have been computed. Much animation can be successfully created in this way with little, or no, jitter or judder apparent to the viewer.

Most animations will require action to be held for periods of time; it is then sensible to instruct the system to replicate a particular frame the relevant number of times (avoiding repeatedly re-computing the same image).

Two device specific routines are provided in DIMFILM for these purposes.

```
      DMULTI(<device>,MFRAM)
```

- this will set the multi-frame count for the specified device (or, optionally, for all active Dicvomed devices).

<device> - is of type CHARACTER, and should be a string containing the device designation (i.e. any of 'D35', 'D35P', 'D16', 'D35C',or 'D16C' for a specific device or 'ALL' for all active Dicomed devices).

MFRAM - is the required multi-frame count (in range [0,5]) for all subnsequent frames.

It must be noted that DMULTI is a deferred action: it does not take effect until after the next frame advance is executed. Subsequent to that advance, following frames will be replicated to generate the specified number of copies of each.

In particular, double framing will be achieved by setting MFRAM = 2. While for MFRAM = N a total of N frames will be generated for each following image. MFRAM = 1 is single framing, and is equivalent to MFRAM = 0. In each case the action will commence following the next reference to FRAME (it is, therefore, sensible - as an aid to clarity - to position the refernce to DMULTI to immediately precede a frame advanmce (i.e. reference to FRAME): the image following that advance will then be the one for which the DMULTI action first takes effect. It will remain active until reset.

(Single and double framing may also be achieved through reference to the DSET routine, described below.)

DHOLD(<device>,NFRAMS)

- this will repeat the next frame the specified number of times, for the designated 16mm device, provided it is currently active.

<device> - is of type CHARACTER, and should be a string containing the device designation (i.e. any either ’D16’, or ’D16C’). This option is only available for 16mm animations on the Dicomed film rcorder.

NFRAMS - is the numbers of times the following complete frame is to be generated.

It must be noted that DHOLD is available only for 16mm film devices. The action is deferred until the next frame advance is executed. The following frame will then be generated the requisite number of times, after which framing will revert to single or multi framing (as presently set). In particular, NFRAMS is the number of frames in total that will be generated for the hold. That is, if NFRAMS = 1 a single frame only will be produced (even if currently multi-framing). In general, for NFRAMS = N the original frame will be generated followed by (N-1) repeats, yielding a total of N frames. As DHOLD is a deferred action it is sensible to position the reference to it to immediately precede a frame advance (i.e. reference to FRAME): the image following that advance will then be the one to which the DHOLD action refers.

3.1.3 - Dicomed device control

There are other ways in which Dicomed devices can be controlled or reset. A routine, DSET, has been provided for some such requirements. In particular, this can be used to globally change the orientation of a device or to control single or double framing (in this it is a simple alternative to the facility provided in the DMULTI routine).

DSET(<device>,KEY)

- this will select particular operational modes for the specified device (or, optionally, for all active Dicvomed devices).

<device> - is of type CHARACTER, and should be a string containing the device designation (i.e. any of ’D35’, ’D35P’, ’D16’, ’D35C’,or ’D16C’ for a specific device or ’ALL’ for all active Dicomed devices).

KEY - is a character string specifying the required action. It may be one of the following:

SINGLE - turn on single framing, deferred until after the next frame advance.

DOUBLE - turn on double framing, deferred until after the next frame advance.

COMIC - turn on comic mode orientation.

CINE - turn on cine mode orientation.

LANDSCAPE - turn on landscape mode orientation.

PORTRAIT - turn on portrait mode orientation.

For a fuller discussion of single and multi framing refer to the description of DMULTI.

Orientation of an image may be considered in two ways. Typically one refers to a picture as being in either landscape or portrait orientation. A picture is said to be landscape if its longer dimnension is horizontal, while it is portrait if its longer dimension runs vertically.

For film the equivalent term is comic or cine; for comic mode the picture base is longitudinal with respect to the (edge of the) film, while in cine mode the base of the image is transverse (i.e. across the film). For each film format there is a different default and imaging is done relative to this. Through reference to DSET it is possible to change that orientation. The effect is of rotating the image through ninety degrees before recording it on the film. The default mode for 16mm film is CINE and for 35mm is COMIC, but due to the aspect ratios of each both correspond to LANDSCAPE images. (For 16mm the longer dimension is transverse, while for 35mm it is longitudinal.) Thus the default image layout is LANDSCAPE for all the Dicomed film modes. The routine DSET enables the production of, for example, portrait mode slides or portrait mode images on 16mm (the latter would be unsuited for projection, but would be quite suitable for archive purposes).

For each device the equivalent combinations may be shown as:

DEVICE:	D35	D16	D35P	D35C	D16C
ASPECT RATIO					
LANDSCAPE	COMIC	CINE	COMIC	COMIC	CINE <--DEFAULT
PORTRAIT	CINE	COMIC	CINE	CINE	COMIC

It is strongly advised that changes of orientation are made only at the beginning of a frame; they take immediate effect and various device transformations will be reset to reflect the selected mode.

3.1.4 - Raster elements

As intimated above, the raster element as used on the D48C may be composed of a grouping of points with both point and element spacing being variable. DIMFILM raster facilities do not require the user to address this topic. All mappings from the cell mapped area to device raster elements are performed within the device drivers. However, it is recognised that for certain serious applications the user will require more detailed understanding of the raster construction. Additionally the user is provided with the means to respecify the raster characteristics (and DIMFILM will take account of the actual raster structure when transmitting cell data to the Dicomed).

The Raster element is the unit by which the D48C plots a raster picture (i.e. it is the basic picture element, or pixel). When a raster is transmitted to the D48C a series of required pixel intensity values are transferred. Successive raster elements are plotted with those successive intensities, the same intensity being applied to each point in the element. When one element has been plotted the beam is spaced to the next; when a scan line is complete the beam is repositioned ready for the next scan. The element spacing (both horizontally and vertically) may be set. Each raster element is a square or rectangular array of one to sixteen points in each of the horizontal and vertical directions. The point spacing (both horizontally and vertically) within the element may also be set. The points correspond to the 32K grid addressability which may be used to expose a single spot on the CRT - these points should not be confused with pixels, of which they are a component. Only in the limiting case of elements consisting of single points will points be equivalent to pixels. The raster elements are the pixels of which the image is composed, and it will be seen that the D48C offers the user the ability to specify both the size and spacing of pixels.

The raster structure is determined by a number of parameters. The raster element consists of an array of points, arranged with a specified number of points horizontally and a number of points vertically , these being in the range [1,16]. Spacings are expressed in raster units. That is, in terms of the 4K x 4K maximum raster grid of the D48C. However, the spacings may be stepped by eighths of the raster unit which is equivalent to units on the 32K x 32K addressable grid of the CRT. (It is these latter units which are used in DIMFILM.) The spacing of points within this array may be set independently in each direction via the horizontal point spacing and the vertical point spacing , each of these being in the range 1 to 7-7/8 raster units (i.e. in the range [8,63] CRT addressable units). Similarly, elements may be spaced in each direction by specifying the horizontal element spacing and the vertical element spacing , again in the range 1 to 7-7/8 raster units (i.e. [8,63] CRT addressable units). The element spacing is measured between the lower left corners of adjacent elements (in the appropriate direction), and should be such that adjacent elements do not overlap (i.e. in either direction the element spacing should exceed the product of the number of points and the point spacing).

```
--------------------------------------------------------------------------------

                     The element structure may be shown:


                          ----------------------------------
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          ----------------------------------
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          O----O---------X----X-------------
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          ----------------------------------
                          |    |    |    |    |    |    |
                          O    O    |    X    X    |    |
                          |    |    |    |    |    |    |
                          ----------------------------------
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          O----O---------X----X-------------
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          ----------------------------------
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          ----------------------------------
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          ----------------------------------
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |
                          X----X---------X----X-------------
                          |    |    |    |    |    |    |
                          |    |    |    |    |    |    |


--------------------------------------------------------------------------------
```

In the foregoing illustration, the lines represent a part of a 4K x 4K grid on the CRT (i.e. raster units). The ’O’s are the points comprising one raster element, while the ’X’s are points in adjacent elements. The grid divisions (i.e. 1/4096 of image dimension) are termed raster units . In the example a raster element consists of six points arranged as 2 horizontally by 3 vertically. The horizontal point spacing is 1 raster unit, and the vertical point spacing is 1.5 raster units. The spacing between elements is determined by the horizontal element spacing of 3 raster units, and a vertical element spacing of 7 raster units.

DIMFILM provides the means whereby the user may change the specification of the raster structure.

```
      DRAST(<device>,HES,HPS,HPE,VES,VPS,VPE)
```

- this will set the raster structure for the designated device, provided it is currently active.

<device> - is of type CHARACTER, and should be a string containing the device designation (i.e. any of 'D35', 'D35P', 'D16', 'D35C', 'D16C' for a specific device or 'ALL' for all active Dicomed devices),

HES - is the horizontal element spacing in CRT addressable units (= 8 x raster unit), as an INTEGER in the range [8,63], but subject to exceeding HPExHPS.

HPS - is the horizontal point spacing in CRT addressable units (= 8 x raster unit), as an INTEGER in the range [8,63].

HPE - is the number of horizontal points per element , as an INTEGER in the range [1,16].

VES - is the vertical element spacing in CRT addressable units (= 8 x raster unit), as an INTEGER in the range [8,63], but subject to exceeding VPExVPS.

VPS - is the vertical point spacing in CRT addressable units (= 8 x raster unit), as an INTEGER in the range [8,63].

VPE - is the number of vertical points per element , as an INTEGER in the range [1,16].

Specific references exist for each individual device if such mode is preferred.

On initialisation, each Dicomed device is initialised with a 2048 x 2048 raster with elements consisting of 2 x 2 unit spaced points. Element spacing is 2 raster units. The equivalent parameters to DRAST would be HES = 16, HPS = 8, HPE = 2, VES = 16, VPS = 8, and VPE = 2. For most purposes this will yield eminently acceptable results.

3.1.5 - Line Width/Spot Size

The Dicomed devices are able to produce thickened lines and (hardware) spots of various sizes. These are controlled through the DIMFILM routines for line width and spot size scale factors. The supported range is from 1 to 16 in each case - the default being for a single width line and minimal point. Scale factors below 1.0 will equate to the (default) of 1.0, while values above 1.0 will select the nearest integral value up to a maximum of 16.

NOTE : At present multi-width lines on the Dicomed are produced on a vector by vector basis and no account is made of any preceding or following vector.

Unacceptable results will be produced for thick lines forming parts of curves (in particular characters). This will be investigated in due course; until it has been resolved users are advised to restrict their use of thickened lines to lower values of the scale factor (a maximum of 3.0 is suggested).

3.1.6 - Plot Dataset Length

DIMFILM attempts to utilise the Amdahl queues in an economic and efficient manner. To this end a limit is applied to the permitted size of a user’s plot dataset. At present, the Dicomed drivers will terminate when a dataset length of 100 MBytes is exceeded. We shall review this limit in the light of experience, and would be pleased to hear from users who encounter difficulties as a result of this limit. It is possible that the limit may be reduced if operational difficulties result - every effort will be made to give notice of such change, but this cannot be guaranteed.

3.1.7 - Device Sizes

The physical image size on film for each of the Dicomed modes is maintained, to a close tolerance, according to various international standards. The default coordinate system for each film device is intended to produce images of the defined size (within the physical device limitations). While these have been carefully determined, the right is reserved to amend them should the need arise. There will be no effect upon users specifying their own coordinate system through reference to BOUNDS.

3.1.8 - D35 - monochrome aperture card format (35mm)

Medium: 35mm unperforated film (reversal)

Imaging: monochrome, vector and raster

Physical image size: 36.41x27.94 millimetres

Default horizontal coordinates: 0.0,32236.0

Default vertical coordinates: 0.0,24708.0

Default image/film mode: landscape/comic

Default (initial) intensity: 0.8

3.1.9 - D35P - monochrome 35mm portrait mode

Medium: 35mm unperforated film (reversal)

Imaging: monochrome, vector and raster

Physical image size: 27.94x36.41 millimetres

Default horizontal coordinates: 0.0,24708.0

Default vertical coordinates: 0.0,32236.0

Default image/film mode: portrait/cine

Default (initial) intensity: 0.8

3.1.10 - D16 - monochrome 16mm (movie)

Medium: 16mm double perforated monochrome film (reversal)

Imaging: monochrome, vector and raster

Physical image size: 10.26x7.49 millimetres

Default horizontal coordinates: 0.0,32767.0

Default vertical coordinates: 0.0,23920.0

Default image/film mode: landscape/cine

Default (initial) intensity: 0.8

3.1.11 - D35C - colour slide 35mm

Medium: 35mm double perforated, reversal colour film

Imaging: colour, vector and raster

Physical image size: 36.3x24.5 millimetres

Default horizontal coordinates: 0.0,32148.0

Default vertical coordinates: 0.0,21698.0

Default image/film mode: landscape/comic

Default (initial) intensity: 0.8

Default (initial) colour: (0.8,0.8,0.8)

3.1.12 - D16C - colour 16mm (movie)

Medium: 16mm double perforated, reversal colour film

Imaging: colour, vector and raster

Physical image size: 10.26x7.49 millimetres

Default horizontal coordinates: 0.0,32767.0

Default vertical coordinates: 0.0,23920.0

Default image/film mode: landscape/cine

Default (initial) intensity: 0.8

Default (initial) colour: (0.8,0.8,0.8)

Appendix 4 - Conversion to the new DIMFILM

This chapter tabulates user accessible routines in the original DIMFILM and indicates, where relevant, the new name and any change in arguments. The major parameter change is associated with character strings. Implementation under Fortran 77 has required the replacement of (hollerith string,character count) by a single parameter of type character.

```
Thus, for example, the old coding
     CALL SYMTEXT(6HABCDEF,6)
becomes
     CALL SYMTXT('ABCDEF')
```

The other general change affecting parameters is the change from an integer intensity to a real value (normalised in the range 0. to 1.).

Other than subroutine name changes, the major difference the user will encounter is the replacement of the single character alternators (user assignable) by a predetermined set of escape sequences. This has the advantage of only removing two characters from normal usage in strings and simultaneously offering even greater flexibility to the user in formatting and controlling his textual output.

New features are not generally described in this Appendix unless they directly replace an original feature. Thus, for instance, colour specification and raster plotting is not described here - nor are the device nomination routines appropriate to them.

4.1 - Routine name conversion

Notes: n/a - indicates routine or equivalent is no longer available

blank in second column if no change

+C+ Hollerith string, count parameter pair(s) replaced by character type parameter(s)

+R+ integer intensity value replaced by real value

Original	New routine name	
ACCENT	n/a - but see new text description	
AUTODIV	AUTD	
AUTOR		
AUTORTH	AUTORT	
AUTOTH	AUTOT	
AUTOX		
AUTOXY		
AUTOY		
AUTRDIV	AUTORD	
AXISCUT	AXCUT	
AXTIKON	AXTIK	
BLANK		
BLNKFRM	OBLANK	
BNDRY	INQB	
BOUNDS		
BOX	RECT	- see also new routine BOX
BRKSYM		
CALPNT	n/a	
CAM16MM	D16	
CAM35MM	D35	
CASEALT	n/a - but see new text description	
CFMT		+C+
CHAR	MARKER	- but see new description
CHARA	n/a	
CHARL	n/a	
CHECK		
CHHALF		
CINE	n/a	
CINTERP	CINTER	
CIRCLE		
CIRCSEG	CIRARC	
CLABEL		
CLABHT		
COMIC	n/a	
CONTOUR	CONTR	
CONTR1		- see description re last parameter
CONTR2		
COORDS	INQC	
CQUAD		
CSCALE		
CVHALF		
CWHOLE		
DASH		
DASHDOT	DSHDOT	
DASHOFF	DSHOFF	
DEFALTS	n/a	
DEFINT	n/a - but see general description of intensities	
DEFMATH	n/a - but see new text description	
DEGREES	DEGREE	
DEGSEP	RADSEP	

DIVUNIT	DIVUNI
DOT	
DRAWXAX	DRAWXA
DRAWYAX	DRAWYA
DTHETA	
EDGES	
ELLIPSE	ELLIPS
ELLSEG1	ELARC1
ELLSEG2	ELARC2
ENDBLNK	ENBLNK
ENDCONT	ENSYMC
ENDFILM	DIMEND
ENDWIND	ENPANE
FNPLOT	
FRACTNS	n/a - but see new text description
FULL16	n/a
GRAFCON	GRCON
GRAFDEF	GRDEF +C+
GRAPHER	GRAPHE
GRAPHFN	GRAPHF
GRAPHIC	GRAPH
GREEK	n/a - but see new text description
GRFRAME	GRFRAM
GROFF	
GRON	
GRSLIDE	GRCONA
GRTOAC	
HATCH	
HISTGRM	HISTGR
HPER	
HREP	
HVYALT	n/a - but see new text description
HVYINT	+R+
HVYTYPE	HVYTYP
IBACK	n/a - but see general description of intensities
IBCOUNT	IBCNT
IDEF	n/a - but see general description of intensities
INTERP	
INTREND	INTRND
INTRUPT	INTRPT
INTSTY	INTSY +R+ - single real parameter (0.,1.), see description
ITALIC	
ITALT	n/a - but see new text description
IVAL	INQR/INQI
IXAX	
IYAX	
IZVAR	INTVAR +R+
JOIN	
LIMIT	n/a
LINALT	n/a
LINESYM	n/a
LINK	
LINR	
LINX	
LINXY	

LINY		
LNUM	n/a	
LOGLIM		
LOGR		
LOGTIK		+R+
LOGX		
LOGXY		
LOGY		
LOWCASE	LOWER	
LRYOPPT	LRYOPT	
LRYTICK	LRYTIK	
LRYVAL		
LTITLE		+C+
LUXOPPT	LUXOPT	
LUXTICK	LUXTIK	
LUXVAL		
LXAXLAB	LXALAB	+C+
LXAXTIK	LXATIK	
LXAXVAL	LXAVAL	
LXLABEL	LXLAB	+C+
LXOPPT	LXOPT	
LXTICK	LXTIK	
LXVAL		
LYAXLAB	LYALAB	+C+
LYAXTIK	LYATIK	
LYAXVAL	LYAVAL	
LYLABEL	LYLAB	+C+
LYOPPT	LYOPT	
LYTICK	LYTIK	
LYVAL		
MARGIN		
MATHEXP	n/a - but see new text description	
MONTHSX	MONSX	
MONTHSY	MONSY	
MONTHX		
MONTHY		
NEWFRAM	FRAME	
NEWORIG	NEWORG	
NEXTONE	n/a - but see new text description	
NOACCNT	n/a - but see new text description	
NOAXTIK	NOAXT	
NOCHECK	NOCHCK	
NOCINT		
NOCLAB		
NOFRACT	n/a - but see new text description	
NOITAL		
NOIZVAR	NOIVAR	
NOMATH	n/a - but see new text description	
NONEXT	n/a - but see new text description	
NOROT		
NOSPEC	n/a - but see new text description	
NOSPOS		
NOSUB	n/a - but see new text description	
NOSUPER	n/a - but see new text description	
NOTXTCN	ENTXTC	
NOUNLIN	NOUNLN	
NRMTYPE	NRMTYP	
NUM	RNUM/INUM - real/integer value, format as	

	character type	
OFFTOXY	OFF2	
OLDORIG	OLDORG	
ONTOXY	ON2	
OPMESS	n/a	
OUTLINE	OUTLIN	
OUTSLDE	n/a	
OWNDASH	OWNDSH	
PARALL1	PARAL1	
PARALL2	PARAL2	
PIECHRT	PIECHT - caption now singly dimensioned character array	
POINT		
POLRDEF	POLDEF	+C+
POLYGON	POLYGN	
POLY3		
POLY3ER	POLY3E	
POLY5		
POLY5ER	POLY5E	
POSN		
PTPLOT		
PTPLTER	PTPLTE	
RADIANS	RADIAN	
RELAXES	RELAX	
RELFRAM	RELFR	
REPEAT	n/a - to be implemented in more versatile manner	
ROMAN	n/a - but see new text description	
ROMGRK	n/a - but see new text description	
ROTATE		
RRANGE		
RTHDIV	RDIV	
RTHETA	POLAR	
RTHFN	POLFN	
RTHFRAM	POLFR	
RTHGRID	POLGRD	
RTHLEVS	TLEVS	
RTHOFF	POLOFF	
RTHON	POLON	
RTHOUT	POLOUT	
RTHPLY3	POLPY3	
RTHPLY5	POLPY5	
RTHPTS	POLPTS	
RTHTOAC	POLTOA	
RTIKS	RTIK	
RVALS	RVAL	
RXAX		
RXTIK		
RXVAL		
RYAX		
RYAXLAB	RYALAB	+C+
RYAXTIK	RYATIK	
RYAXVAL	RYAVAL	
RYLABEL	RYLAB	+C+
RYOPPT	RYOPT	
RYTICK	RYTIK	
RYVAL		
SAMER		

SAMERTH	SAMERT	
SAMETH	SAMET	
SAMEX		
SAMEXY		
SAMEY		
SERIAL	n/a	
SETDASH	SETDSH	
SETITAL	SETITL	
SETUP	n/a	
SHDEGRM	SHDEGR	
SLDSYM		
SLIDE	n/a	
SNUM	SRNUM/SINUM - real/integer value, format as character type	
SPECHAR	n/a - but see new text description	
STEP	n/a	
STEPGRM	STEPGR	
STRAIL		+C+
STRING		+C+
SUB	n/a - but see new text description	
SUBALT	n/a - but see new text description	
SUPALT	n/a - but see new text description	
SUPER	n/a - but see new text description	
SYMANGL	SYMANG	
SYMCONT	SYMCON	
SYMHT		
SYMINT	- to be made available (renamed)	
SYMPOS		
SYMTEXT	SYMTXT	+C+
TEXTCON	TXTCON	
THRANGE	TRANGE	
THTIKS	TTIK	
THVALS	TVAL	
TICKCON	TIKCON	
TITLE	n/a	
TMPORIG	TMPORG	
UNLIN		
UPCASE	UPPER	
UTITLE		+C+
UXAXLAB	UXALAB	+C+
UXAXTIK	UXATIK	
UXAXVAL	UXAVAL	
UXLABEL	UXLAB	+C+
UXOPPT	UXOPT	
UXTICK	UXTIK	
UXVAL		
WINDFRM	OPANE	
WINDOW	PANE	
XAXIS		
XGRID		
XUNIT		
XYGRID		
YAXIS		
YGRID		