Laser-Scan Ltd.

LITES2

Reference Manual

Issue 4.2

Chapter 13 - Raster

Science Park, Milton Road, Cambridge, England CB4 4FY tel: +44 (0) 1223 420414

Document "LITES2 Reference"		Category "REFERENCE"
Document Issue 1.0	Paul Hardy	7-Mar-1985
.		
.		
.		
Document Issue 3.0	Ron Russell	9-Oct-1989
Document Issue 3.1	Ron Russell	5-Jan-1990
Document Issue 3.2	Ron Russell	19-Jul-1990
Document Issue 3.4	Clarke Brunt	22-Mar-1991
Document Issue 3.5	Clarke Brunt	30-Sep-1991
Document Issue 4.0	Paul Hardy	22-May-1992
Document Issue 4.2	Clarke Brunt	21-Jul-1993
Document Issue 4.3	Clarke Brunt	18-Jan-1994

13 **Display Overlays and Raster Images**

These two topics are described together because they are related in some ways.

Display overlays allow the bit planes of a graphics display to be divided into independent groups called 'overlays'. Subsets of vector IFF data and raster images can be displayed in each overlay, with the advantage that the overlays can be made visible/invisible and be moved to foreground/background etc. independently and instantaneously. Overlays are created by the WORKSTATION OVERLAY command and are subsequently manipulated by the OVERLAY command. Although these commands are present in all version of LITES2, the ability to display overlays is confined to certain models of graphics workstation. See the appropriate Workstation Guide for details.

The ability to read in raster images is a licensed option in LITES2. It is only available in versions of LITES2 for displays capable of displaying overlays.

13.1 **Display Overlays**

Display overlays are created using the WORKSTATION OVERLAY command. This command allocates a number to the overlay - this number is used to refer to the overlay in future. The number of separate overlays which may be created is limited only by the number of pixel planes on the display, since each overlay must consist of a distinct contiguous set of planes. The particular planes to use is normally allowed to default, but an optional argument to the command allows the user to specify this if required. Remember that the number of colours (including a background colour) which may be displayed in an overlay is 2**number-of-planes i.e. 8 for 3 planes, 16 for 4 planes etc. In the case of workstation with primary and secondary displays both supporting overlays, then the overlays are shared between the two displays - creating an overlay, or setting its attributes, will automatically perform the same operation in the other.

Once an overlay has been created, it is advisable to set all colours in it. In addition to a colour value, specified by the OVERLAY COLOUR command, each colour has an attribute to specify how it interacts with the colours of overlays beneath it. It may, for instance, be TRANSPARENT, in which case the underlying colour will show through unchanged, and the setting of this colour will be irrelevant. Another attribute is OPAQUE, in which case the underlying colour will be totally obscured by this colour. The INVERSE attribute produces a complimentary colour to the underlying colour - the setting of this colour is irrelevant. Other attributes are ADD, SUB, and MERGE, which allow the underlying colour to combine with this colour in various ways. Attributes are set by the OVERLAY ATTRIBUTE command, which may be used to set the attributes of all the colours in an overlay simultaneously or individually. Note that if the background colour (colour 0) is made opaque, it will never be possible to see any overlays beneath this one.

An overlay may be made invisible or visible. This is done using the OVERLAY CONCEAL and OVERLAY REVEAL commands. An overlay is CONCEALed when created - it must be REVEALed before anything will be visible in it.

The final colour appearing at a given point on the screen is determined as follows. A ray of light is assumed to set off in the colour set by the OVERLAY BACKDROP command (this is not associated with any particular overlay). It is

then modified as it passes through each visible overlay in turn, the colour of the ray interacting with the colour in the overlay according to the OVERLAY ATTRIBUTE for the colour in the overlay.

The exact behaviour of the different colour attributes is as follows. Assume that 'C' represents the amount of red, green, or blue in the ray of light passing through an overlay, and 'c' represents the colour in that overlay. The way that 'C' is modified on passing though the overlay is:

TRANSPARENT	C = C
OPAQUE	C = c
INVERSE	C = 1 - C
ADD	C = C + c - C*c
SUB	C = C*c
MERGE	C = (C + c)/2

The behaviour of TRANSPARENT and OPAQUE is fairly obvious. INVERSE produces a complimentary colour, for instance black<->white, yellow<->blue. ADD behaves like additive colours (rays of light) so that for instance red + green would give yellow. In ADD mode, the resulting colour is always brighter and nearer to white than the two component colours so that adding black to a colour leaves it unchanged, while adding white to a colour always produces white. SUB behaves like subtractive colours (paint pigments) so that for instance red + green would give black, while yellow + cyan would give green. In SUB mode, the resulting colour is always darker and nearer to black than the two component colours so that adding black to a colour produces black, while adding white to a colour leaves the colour unchanged. MERGE produces an effect intermediate to ADD and SUB - the two colours are averaged producing something intermediate in colour and brightness to the originals.

The order in which the overlays are traversed may be altered at will - it is not fixed by the particular planes allotted to the overlays. The OVERLAY POP command moves an overlay to the front of all the others, while OVERLAY PUSH moves it to the back. Using these two commands, it is possible to arrange any number of overlays in any order.

The default state of an overlay when created is that it is CONCEALed, POPped to the foreground, has a TRANSPARENT black colour 0 (background), and has all other colours OPAQUE white. The minimum which must be done to see anything in the overlay is to OVERLAY SELECT something to be drawn in it, OVERLAY REVEAL it, and re-draw.

Any changes to overlays affecting the appearance of the screen normally take place at once (this is actually achieved by re-computing a colour table and writing it to the display). If many changes are being made, it is inefficient to update the display after each, so the OVERLAY DEFER command prevents updating of the display. Once the changes have been made, the OVERLAY UPDATE command will update the display to the current state.

If no overlays are created, then LITES2 will continue to function as previously - the IFF vector data will be written into all available planes of the workstation. Colours are specified initially using a colour table file on disc, and subsequently using the WORKSTATION COLOUR command. Once any overlays have been created, LITES2 will use the overlays in preference to its default behaviour. In this case, the colour table file is irrelevant, and the WORKSTATION COLOUR command should not be used (it will still have an effect, but

this will be overridden by the next OVERLAY command). Nothing will be displayed in an overlay until it has been explicitly selected. This is achieved using the OVERLAY SELECT/DESELECT commands. The original SELECT/DESELECT commands still function as before - they control which features may be found, drawn, or output. A feature must be selected both by ordinary SELECT commands, and OVERLAY SELECT commands to be drawn into an overlay. Although it is possible to select a feature for display in more than one overlay, it will only appear in the lowest numbered overlay. Having made appropriate selections, it is necessary to DRAW, WINDOW, ZOOM etc. to create the new display.

Raster images may also be displayed in overlays. These are selected using OVERLAY SELECT/DESELECT IMAGE commands.

The SHOW OVERLAYS command may be used to display details of all overlays, including the current selections for them.

To remove all selections from an overlay use OVERLAY CLEAR. It is then possible to delete the definition of the overlay using OVERLAY DELETE, freeing its planes for use by new overlays. If all overlays are deleted, then LITES2 will revert to its original non-overlay behaviour.

13.2 **Raster Images**

Up to 8 DTI files may be accessed simultaneously by LITES2. Only one LSI file may be accessed - it uses two of the slots. Up to 8 LSR files may be accessed, but depending on the format of the LSR file, it may require 1 or 2 slots, which could reduce the limit to 4. These are specified at any time using the IMAGE DTI, IMAGE LSI, or IMAGE LSR commands. An image is allocated a number from a preceding IMAGE NUMBER command - this number is used to refer to the image in future. An image may be closed (possibly to make room for another) using the IMAGE CLOSE command. Images should not be opened and closed unless absolutely necessary - this may eventually result in LITES2 running out of computer memory.

The SHOW IMAGES command may be used to display details of images.

In the case of DTI files, information on the pixelsize and origin of the file is taken from the projection record in the file header (if present). If this information is not available (this includes LSI files), then the information must be supplied using IMAGE ORIGIN and IMAGE PIXELSIZE commands. For DTI files, the orientation of the file may be specified using IMAGE CORNER and IMAGE DIRECTION commands. This information is normally also obtained from the file header. Because some displays are more efficient at displaying raster images in certain directions, it may be advisable to use the MATRIX package utility DTIROTATE to alter the file to the optimum orientation.

The origin of a DTI file is taken to be at the centre of the bottom left pixel, while for LSI files it is at the bottom left of this pixel. LSR files can behave either way, according to whether they have point or area type pixels.

LITES2 will proceed from INITIAL to READY state when the required number of IFF files have been specified, or the MAPS 0 command is given (in this case, at least one image file must be open). Any image files open at this time will be used in the calculation of the 'working area' (the maximum x and y extents of image and IFF files + 5% all round). Although image files lying outside this

area may be specified subsequently, it will not be possible to access any parts of these files lying outside the working area. The RANGE LIMITS command may be used to set a larger working area initially. Images are not drawn during initial read-in. A DRAW, WINDOW, or ZOOM command must be given before they appear.

Images must be displayed in overlays. The OVERLAY SELECT/DESELECT IMAGE commands are used to specify which images are to be displayed in which overlays. Several images may be displayed in a single overlay (this is not useful if the images overlap - the higher numbered image will overwrite the others), and the same image may be displayed in several overlays (uses of this are not obvious, but there may be some!).

The colour displayed for a given pixel value is obtained by one of two methods. The default method, which is the fastest, is to take the pixel value as the colour index directly. The only control is provided by the IMAGE BITS command, which allows a set of bits to be selected from the pixel value. Another way of looking at the IMAGE BITS command is that the displayed colour is obtained from the data value as follows:

display_colour = (data_value/2**first_bit) modulo 2**number_of_bits

For instance, if a 16 bit (WORD format) DTI file was to be displayed in a 4 plane overlay, the default action would be to display the 4 least significant bits. The IMAGE BITS command could be used to display some other set of contiguous bits instead. The number of bits specified need not be the same as the number of planes in the overlay: extra high order bits are ignored, or the value is padded with 0 at the high order end if insufficient bits are specified (thus one could for instance display a single bit in a multi-plane overlay).

The second method of producing a colour index from the pixel value is called 'classification'. This is switched on by the IMAGE STEP command, and further controlled by the IMAGE RANGE, IMAGE FIRSTCOLOUR, IMAGE BAND, and IMAGE SEA commands. Although slower than the IMAGE BITS method, classification allows greater versatility is displaying the image.

In order for several image files to be displayed simultaneously, they must have the same pixelsize. If files with different pixel sizes are selected for display, then any which do not have the same pixel size as the lowest numbered file will not be displayed.

It is not possible to display image files at arbitrary zoom factors, since each image pixel must correspond to an integer number of screen pixels. In particular, the maximum number of pixels which may be displayed is limited by the number of pixels on the display screen. If the working area is larger than this, it will be impossible to display the whole area. In this case, for DTI or LSR files, the image is subsampled, taking every n'th pixel, but otherwise the window will be automatically reduced. When WINDOWing or ZOOMing into the picture, the nearest integer zoom factor displaying at least the required area will be chosen. A result of this is that the area drawn may not correspond exactly to the area expected. The SHOW WINDOW command may always be used to determine the area currently on the screen. LSI files may contain 'reduced views', with less pixels at a degraded resolution compared to the original view. These will be used whenever possible to allow a larger area to be displayed than would otherwise be possible. A reduced view may only be used if all selected image files contain the particular reduced view. Note that DTI and LSR files do not contain reduced views - subsampling is used instead.

The command ZOOM n IMAGE may be used to attempt to draw the image at a particular zoom factor, for instance ZOOM 2 IMAGE will draw with 2 screen pixels (in each direction) for each image pixel, while ZOOM 0.5 IMAGE (or ZOOM 1/2 IMAGE) will sub-sample with each screen pixel representing 2 image pixels in each direction.

For LSR files containing bit data, the command IMAGE SUBSAMPLE PRIORITY n may be used to specify that rather than missing out pixels completely when subsampling, pixel value 0 or 1 be given priority. This can help prevent the break up of subsampled images, with the loss of fine detail. IMAGE SUBSAMPLE FAST reverts to the normal behaviour.

The image data value at the current cursor position may be obtained using the $IMAGEVALUE system variable. An estimate of the gradient and aspect (direction of maximum gradient) may be obtained using the $IMAGEGRADIENT and $IMAGEASPECT system variables. It is not necessary for the image to be displayed in order to do this, indeed it is not necessary to have a graphics display at all - the 'cursor' may be positioned by keyboard commands. Since several images may cover the same point, it is necessary to specify the set of images to consider for this purpose using the IMAGE SELECT command. More than one image may usefully be specified in the case when they 'tile' together to form a single image. In the case when selected images overlap, the value will be taken from the highest numbered image.

When producing hardcopy plots which include LSR images, then logical name LSL$LSR_PLOT_MODE provides some control over the way the image data is written to the plot file. The default (equivalent to defining the logical name as 0) is to plot non-subsampled images in the tiles in which they are stored in the LSR file. For very large files, this has sometimes resulted in more tiles than the plotter or rasteriser can cope with. Subsampled images are plotted in swathes the full width of the image, and as tall as allowed by the memory buffers, which are allocated in proportion to logical name LSL$FILL_POINTSMAX (default 8192). If LSL$LSR_PLOT_MODE is defined as 1, then swathes are used even for non-subsampled images, hence the number of swathes can be controlled by the size of the buffers. Long thin swathes have sometimes caused problems if the plotter tries to rasterise the plot with scan lines running in the opposite direction to the swathes. If LSL$LSR_PLOT_MODE is defined as 2, then the plot is drawn as square tiles as large as will fit in the buffer, so rasterising in either direction should be equivalent. If LSL$LSR_PLOT_MODE is defined to be greater than 100, then square tiles with the specified number of pixels on each side will be used (provided they will fit in the buffer). For bit images, tile sizes which are a multiple of 8 may give better performance.

13.3 **Raster Editing**

LITES2 supports some editing of raster images. This facility is at present restricted to LSR type files containing bit data. To enable editing, the command IMAGE EDIT must be given (after IMAGE NUMBER, but before IMAGE LSR). IMAGE READONLY is the opposite of IMAGE EDIT, and is the default.

Once the image is open, its background and foreground values may be specified using IMAGE BACKGROUND (default 0) and IMAGE FOREGROUND (default 1). Since only bit images are supported, the values must be either 0 or 1. These commands control the behaviour or the rest of the image editing commands.

Most of the image editing commands operate on the interior of one of LITES2's REGIONs. The IMAGE REGION command specifies which region number is to be used. The region need not exist when this command is given, and may be redefined subsequently. The region may be created in all the usual ways, plus the more recently implemented REGION WINDOW (quickly constructs rectangular regions using the cursor), and REGION IMAGE (a region constructed from the image data itself, around pixels of the same colour).

IMAGE CLEAR and IMAGE FILL fill the interior of the region with background and foreground colour respectively.

IMAGE COPY and IMAGE MOVE both allow the image data within the region to be attached to the cursor, and moved until an END or ABANDON command is given. IMAGE COPY retains the original data, while IMAGE MOVE fills the original region with background.

IMAGE PAINT and IMAGE ERASE allow the image to be 'painted' in foreground and background colour respectively using the cursor. PAINT state is entered and copies of the current brush are deposited each time the cursor is moved until END is given (or ABANDON, which undoes any painting). The shape and size of the brush is controlled by the IMAGE BRUSH CIRCLE/RECTANGLE commands.

IMAGE SPECKLE CLEAR and IMAGE SPECKLE FILL search the region for speckles of foreground or background colour respectively, smaller than a given size, and remove them by painting with background and foreground colour respectively.

When searching for speckles, or constructing a region with REGION IMAGE, pixels may be taken as connecting if the touch by their sides (IMAGE CONNECT SIDE, the default), or also diagonally (IMAGE CONNECT DIAGONAL).

IMAGE BURN_IN allows any data currently displayed on the screen to be 'burned into' the raster image. This operation affects just the screen area, and the image must be displayed with one screen pixel to one image pixel at the time (the command ZOOM 1 IMAGE facilitates this). Any pixels displaying a colour other than the image background are written into the image in the foreground colour.

IMAGE RECOVER allows the effect of the last image editing command to be undone, restoring the image to its previous state.

In order to point accurately to image pixels, without having to zoom in on the main image, the DRAW IMAGE command may be used to draw the image into the annotation display (3 or 4, if available), and pointing in this display will then track the LITES2 cursor. A cursor may be displayed in the annotation display also using DISPLAY CURSOR.

13.4 **Image Registration**

LITES2 allows an IFF file map to be registered on the screen with a raster image when the two are not in exactly the same coordinate space. One or more images, and an IFF map, are specified and read in as normal. It is advisable that the pixel size and the origin of the images is set so as to make the data line up as well as possible before performing the registration. Once in READY state, the command IMAGE SETUP should be given, and LITES2 will enter SETUP state and

prompt for the position of the NW corner point of the IFF file to be digitised on the screen. WINDOW, DRAW, and ZOOM commands may be given to get the correct area of the raster image on the screen. The cursor should be moved to the position on the image where the NW corner point of the IFF file should be - the command START will digitise the point and move on to the SW corner point. When all 4 corners have been successfully digitised, LITES2 will return to READY state and the next redraw will register the corner points of the IFF file with the image. ABANDON may be used at any time in SETUP state to abort the SETUP and return to default behaviour.

The EXTENDED transformation (see SETUP TRANSFORM) is used for the registration, which means that the 4 corner points will fit exactly to the positions digitised, with the rest of the map being distorted as required.