User Defined Ganzfeld Script Files

Overview

From release V3 of the Espion software, the program has incorporated the ability to read and display Ganzfeld colors and luminances from a user defined script file rather than relying on the built in facilities. This allows users to design their own stimuli, color cycling, beat frequencies etc that are not catered for normally. This user defined file can be created in any text editor or programs such as Excel or Sigmaplot as it is based on a rows/columns format and stored in plain ASCII making it easy to read/debug and modify. To select this option, simply choose 'User Defined' from the 'Flash Mode' parameter on the Ganzfeld stimulator menu.

Features

- ASCII files rather than binary allowing creation by any text editor
- Creation by Standard programs including Excel and Sigmaplot
- Complicated formula can be written directly as the program has a built in mathematical Parser
- Functions can entered directly including ABS, ATAN, COS, EXP, LN, ROUND, SIN,SQRT, SQR and TRUNC. It also includes the built in constant PI
- 4 user defined variables which can be changed from within the program
- Simple string substitution allowing change to be made in real-time to formula, values etc
- Built in incrementing/decrementing variable for creating for loops in formulae
- User definable column delimiter (default tab character)
- Ability to set colors either as RGB, or CIE values
- Luminances can be defined in Scotopic or Photopic units
- Unrestricted length of file (producing stimuli that can last many seconds even minutes)
- 1ms resolution
- Pulse widths up to 65535 ms as standard (more can be achieved)
- Colors can be picked from the standard Ganzfeld color table by name
- The background color can be defined to be the programmable background
- Ability to drive two stimulators in parallel or independently

How it works

Every time the program runs the user script, it loads and parses the file creating a series of data blocks. These blocks control the Ganzfeld directly and are decoded at a frequency of 1kHz. This allows changes of color or luminance to happen with a resolution of 1ms. This script file consists of a series of text lines each line begins with the word 'GLOBAL' or 'BLOCK'. This informs the parser what to expect when parsing the line. The line should be thought of as a series of columns separated by a column delimiter (default tab character, but can be changed in the 'Configure System' menu). The parser splits the line into pairs of columns where the first one contains the parameter name and the following column contains the parameter value. For Example:

```
BLOCK       1           CIEX$       0.4         CIEY$       Sin(1)*0.5  LUM$        100
```

All parameter names end with the $ sign to distinguish them from normal text. The following column must contain the parameter value (normally numeric) and this may contain mathematical functions e.g. 0.4+sin(PI*4)/2+SQR(2). The order in which parameters are defined in the line is unimportant and all parameters have defaults if not included.

NOTE:
The file is case insensitive. Parameter names and values may be in upper or lower case.

Creating files

Script files can be created in any ASCII text editor, but we recommend using a spreadsheet such as Microsoft Excel which allows the easy use of columns. If you use Excel, you should close it before running the script, otherwise the Espion program will produce error messages. This is due to not being able to file share with Excel.

User Defined Ganzfeld Script Files

Parameter Parsing

All numeric parameter values are parsed to allow complicated formula to be entered directly. The program supports the following functions and operators:

Operator	Range/Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
^	Exponential (only positive number for base)
(	Open parenthesis
)	Close parenthesis
MOD	Integers only. Float value will be rounded first
ABS	(Range: -1e4932..1e4932)
ATAN	(Range: -1e4932..1e4932)
COS	(Range: -1e18..1e18)
EXP	(Range: -11356..11356)
LN	(Range: 0..1e4932)
PI	3.1415926535
ROUND	(Range: -1e9..1e9)
SIN	(Range: -1e18..1e18)
SQRT	(Range: 0..1e4932)
SQR	(Range: -1e2446..1e2446)
TRUNC	(Range: -1e9..1e9)

E.g.

```
 BLOCK        RED$              0.652+(((0.301-0.652)*(sin(6*PI/4+2*PI*%0/1000)*0.5+0.5))*%1)
```

Adding Comments

Comments may be added to any line in the script file by simply starting the comment with a semicolon. All text after the semicolon is ignored up to the end of the line. For example:

```
GLOBAL    V1NAME$     LUM (cd/m2)   V1DEFAULT$     100         ;This is a comment
```

User defined variables

The script can contain up to 4 variables which can be assigned from the main Ganzfeld stimulator menu allowing the user to produce formula which can be varied at run-time without having to change the script. When a variable is enabled in the ‘GLOBAL’ section, a variable box is displayed on the stimulator menu allowing the variable to be changed. The default value for this parameter and a description can be set using the GLOBAL variables.
The four variables are called ‘%1’, ‘%2’, ‘%3’ and ‘%4’ (or ‘&1’, ‘’&2’, ‘&3’, ‘&4’). You can add these variables to any column entry for any ‘BLOCK’ line as many times as required. The program when it parses the script file, simply substitutes the text read from the main stimulus menu variable entry box instead of the variable name. Thus values may be changed or even whole functions from within the program.

NOTE:
Because Microsoft Excel tries to format ‘%1’ as a real percentage value in general cell format (%1->1%) rather than strings, you should either format the cells as strings or use the ‘&1’ form.

User Defined Ganzfeld Script Files

GLOBAL Parameters

The lines beginning with the word 'GLOBAL' contain information about the script file including a descr iption and any user variables required. More than one 'GLOBAL' line can be included.

Parameter Name	Value/Description
TITLE$	A description of the program displayed on the stimulator menu
V1NAME$	The name/description of variable 1
V1DEFAULT$	The default value for variable 1
V2NAME$	The name/description of variable 2
V2DEFAULT$	The default value for variable 2
V3NAME$	The name/description of variable 2
V3DEFAULT$	The default value for variable 3
V4NAME$	The name/description of variable 3
V4DEFAULT$	The default value for variable 4

Example:

```
GLOBAL     V1NAME$      LUM (cd/m2)   V1DEFAULT$      100
```

NOTE:
The variable is enabled if the parser detects the VxNAME$ parameter. The default value is 0 if not explicitly defined.

BLOCK parameters

All lines that start with the word 'BLOCK' define a set of colors/luminances for the Ganzfeld for a period of time (minimum 1ms, maximum 65535ms). At the end of a block period, the program reads the next block. Thus for a 1 second stimulus where the color changes every ms there will be 1000 'BLOCK' lines.

Parameter	Default	Range	Value/Description
AMBER$	0	0..1 or cd/m^2	Define the amber LED value directly
BLUE$	0	0..1 or cd/m^2	Define the blue LED value directly
CIEX$	0	0..1	Define a color using CIE. CIE x value
CIEY$	0	0..1	Define a color using CIE. CIE x value
COLOR$			Define a color using Espion defined color table. This will automatically set the CIEx, CIEy parameters
FLAGS$	0	0..2^{31}	A selection of bit values which should be added together See FLAGS$ section
GREEN$	0	0..1 or cd/m^2	Define the green LED value directly
INC$	1	±1E38	Loop counter increment value
LUM$	0	1E38	Luminance of color in cd/m^2
MS$	1	1..65535	Number of ms to output this block
RED$	0	0..1 or cd/m^2	Define the red LED value directly
REPEAT$	0	±1E38	Loop counter initial value
UNTIL$	1	±1E38	Loop counter end value
XENON$	0	1E38	Luminance of Xenon in cd.s/m^2

Color Selection

Not all parameters should be combined together. For example there are four ways of defining the color for a block all of which are mutually exclusive.

- RGB LED values entered directly using parameters RED$, GREEN$, BLUE$, AMBER$, FLAGS$
- CIE x, y, luminance values entered directly using parameters CIEX$, CIEY, LUM$, FLAGS$
- Color, Luminance chosen from the Espion color table. Using parameters COLOR$, LUM$, FLAGS$
- Background color only. This uses only the parameter COLOR$ and LUM$

RGB (MANUAL MODE)

The first method allows direct LED manipulation by setting RGB values in the range 0..1 where 1 is maximum intensity. The ring can also be set using the FLAGS$ parameter. Currently the choice of ring is limited to the DIM or BRIGHT ring. You do not have access to the LOW DIM ring from user defined script files. For stimulator 1, selecting the dim ring is a FLAGS$ value of 1024 and for stimulator 2 its 2048. If the dim ring is not selected then the ring is bright by default. You can combine this mode with the add background mode (FLAGS$ value of 16384) but depending on the luminance of the background a manually selected dim ring may be promoted to the bright ring automatically.

```
BLOCK     RED$        0.4          GREEN$     0.4          BLUE$       0.4          FLAGS$    1024
```

As an alternative to entering values between 0-1 and a ring, you can enter values directly in cd/m^2. This requires a FLAGS$ values of 65536. In this mode the ring is automatically selected (and overrides any manual selection) depending on the intensity set. You can also use the Scotopic flag with either entry method.

```
BLOCK     RED$        100          GREEN$     200          BLUE$       50           FLAGS$    65536
```

CIE

The second method allows a color to be directly input if its CIE coordinates are known. The color will be converted internally into RGB values using the stimulator color matrix. By default the luminance will be in Photopic units unless the FLAGS$ scotopic bit is set.

```
BLOCK       CIEX$        0.4            CIEY$      0.6          LUM$        100
```

To specify the CIE coordinates of a particular color defined in the Espion color list, simply use its name

```
BLOCK       CIEX$        RED            CIEY$      RED          LUM$        100
```

Color

The third method allows a color name to be used which must appear in the Stimulator color table (the allowable colors available for the stimulator as shown in any of the Color selection boxes on the menu). If the color is found in the table, then the program uses this color's CIE x, y value directly. If no luminance is specified, then the current background luminance is used.

```
BLOCK       COLOR$       RED            LUM$       1000         FLAGS$      8192         ;Scotopic
```

Background

The fourth method allows the background color as defined on the stimulator menu to be used as the block color. This uses the inbuilt variable '%B' for the COLOR$ parameter. Th e background luminance is also specified by the variable '%B'. If no luminance is specified, then the current background luminance is used.

```
BLOCK       COLOR$       %B             LUM$       %B+10
```

Adding the background to a specified color

The background RGB values can also be added to the current specified BLOCK color by using the FLAGS$ parameter. For example, this could be used to add a red flash to a blue background. Suppose the background is set (from the Espion menu) to be BLUE, 200cd/m^2 then the following will add a red flash of 10cd/m^2 to produce a flash of 1ms of color RED+BLUE with luminance of 210cd/^{2} followed by steady state BLUE background. Program will produce a RED flash of 10cd/m^2 followed by a steady state background of BLUE 200cd/m^2

```
BLOCK      COLOR$       RED            LUM$      10          FLAGS$       16384       ;Add Back
```

However, if you use this feature with manual entry of color mode (RED$, GREEN$, BLUE$ or AMBER$) any ring specified by the FLAGS$ parameter may be overwritten depending on the combination of background and foreground. In the following example, the dim ring is chosen (FLAGS$ bit 10 (1024) is set). This may be converted to bright ring depending on the intensity of the background set in the main menu.

```
BLOCK      RED$         0.5            GREEN$    0.5         FLAGS$       16384+1024  ;Add Back
```

Xenon Variables

When designing xenon flash script files, you have two xenon variables; XENON$ and FLAGS$ to control the xenon system. The XENON$ variables denotes the luminance to be used. If FLAGS$ is not specified then the desktop will flash either the LO xenon tube, or the HI xenon tube or both together depending on the luminance chosen and the current xenon calibration. However, you can manually select one or both tubes using the FLAGS$ variable (See the FLAGS$ section).

This will flash the xenon system at 100cd.s/m^2 on the current background.

```
BLOCK      XENON$       100            COLOR$    %B
```

The second example will flash xenon 4, using time for a duration of 0.2ms and no conversion. Knowing that the longest duration is 2ms, then a luminance value of 1 will produce 2ms, 0.5 will produce 1ms etc

```
BLOCK      XENON$       0.1            FLAGS$    (8+4)
```

Block Period

The default block period (the time the block is output to the stimulator) is 1ms, but this can be extended to any time up to 65535ms. This facility removes the need to create a new block for every ms of the stimulus especially if it doesn't change.

```
BLOCK      MS$          100
```

NOTE:
It should be noted that each block takes 1 ms to perform its action. If you define a block as a delay between other blocks, then you must subtract 1 from the block period e.g.

```
BLOCK      COLOR$       RED            LUM$      1000
BLOCK      MS$          10-1                     ;Subtract 1 because block takes 1ms to perform
BLOCK      COLOR$       RED            LUM$      500
```

This script will produce a red flash of 1000 cd/m^2, for 1ms then a blank for 10 ms and then another red flash of 500 cd/m^2 for 1 ms. No color will be produced between the flashes as no color was specified for the delay block.
You don't need to do this in the following case:

```
BLOCK      COLOR$       RED            LUM$      1000        MS$          10
BLOCK      COLOR$       GREEN          LUM$      500         MS$          10
```

This script will produce a red flash of 1000 cd/m^2, for 10ms then a green flash of 500 cd/m^2 for 10ms.

FLAGS$ parameter

This variable consists of a set of bit values which have the following meaning:

Bit	Value	Meaning
0..4	0-31	If these values are anything other than 0, then the XENON$ values must be in the range 0 to 1. otherwise the XENON$ intensity is specified in cd.s/m^2 **Value Description** 0 Fire Lo tube, use Lo capacitance feedback 1 Fire Lo tube, use Hi capacitance feedback 2 Fire Lo tube, use time (no feedback) 3 N/A 4 Fire Hi tube, use Lo capacitance feedback 5 Fire Hi tube, use Hi capacitance feedback 6 Fire Hi tube, use time (no feedback) 7 N/A 8.15 Fire external xenon tube 1..8
10	1024	If this bit=1 then stimulator 0 will use the DIM range for LED outputs otherwise it will use the bright or dim range depending on color RGB values
11	2048	If this bit=1 then stimulator 1 will use the DIM range for LED outputs otherwise it will use the bright or dim range depending on color RGB values
13	8192	If this bit=1 then the color will be set in Scotopic units. Default is Photopic units
14	16384	If this bit=1 then the background RGB values will be added to the block RGB values as specified by COLOR$, LUM$, CIE$, or RED$/GREEN$/BLUE$/AMBER$
15	32768	If this bit=1 then the block will trigger the acquisition system to start recording
16	65536	If this bit=1 then if you are entering RED$, GREEN$, BLUE$, AMBER$ values directly, instead of specifying an intensity in the range 0-1 where 1 is maximum and a ring, you enter a value in cd/m^2. The program will calculate a value of 0-1 and a ring. This will automatically override any ring selected manually. You can also use the add background option

To create a FLAGS$ parameter with the correct bits set, just add the values together. The following example sets the colors using RGB manual mode, values are entered in cd/m^2 (65536) and the background is added (16384). Finally this block provides the trigger (32768).

```
BLOCK       RED$        100           GREEN$    200          FLAGS$       16384+32768+65536
```

Acquisition Trigger

Any block can be made to trigger the acquisition system simply by setting the FLAGS$ parameter. If no trigger is explicitly found in the script, then the default trigger block is the first. A special case is made when a block contains a Xenon flash. All Xenon flash blocks will produce a trigger regardless of whether one was explicitly specified.

Loop Counter

The script parser has a built in loop counter for creating multiple blocks on the fly. This removes the need to define every block in the script file explicitly. By simply building formulae for block parameters which include the loop counter as a variable, a whole stimulus can be created in one line. There are three parameters which control the loop counter; REPEAT$, UNTIL$ and INC$. The REPEAT$ parameter is the starting value of the loop counter, the UNTIL$ parameter is the loop counter terminate value and the INC$ value determines the loop counter increment similar to one of the C statements below depending on the sign of INC$

```
for(I=REPEAT;I<=UNTIL;I+=INC) // (INC>0)
for(I=REPEAT;I>=UNTIL;I+=INC) // (INC<0)
```

Every iteration of the loop counter creates a new block internally.
To reference the loop counter, in your formula use '%0' The following example creates a stimulus of 1000 blocks (1 second) ramping up the RGB values from 0 to maximum luminance:

```
BLOCK  REPEAT$    0  UNTIL$    999  INC$   1  RED$    %0/999  GREEN$    %0/999  BLUE$    %0/999
```

NOTE:
The loop counter is floating point and all parameters may be positive, negative or fractional and include mathematical equations.

Support for Dual Stimulators

The script files have built in support for dual stimulators e.g. dual ColorBursts. By default if the stimulator 'Output Stimulator' parameter is set to 'Both Stimulators' then both simulators will be made to outp ut the same stimulus. If the block color is set by any means other than direct RGB values, the RGB values calculated for each stimulator will be based on the individual stimulator's calibration parameters correctly producing the correct color and luminance regardless of stimulator being used. However if RGB colors (RED$, GREEN$, and BLUE$ parameters) are specified directly then the stimulator colors and luminances will almost definitely be different.
The script can contain individual parameters for each stimulator allowing independent control of the stimulators. To do this, simply separate the parameter values by a colon (':') character e.g.

```
BLOCK      RED$         0.5:0.3        CIEX$      0.64:0.45
```

If both stimulators are selected, and the script file only contains one set of parameters then these will be used for both stimulators. If the script file contains parameters for both stimulators and only one stimulator is selected, then the relevant set of parameters will be chosen.

NOTE:
Not all parameters can be split for obvious reasons. These include MS$, FLAGS$ and loop variable parameters.

Debugging Script Files

The Espion program has the ability to produce a debug output file generated after parsing a user script file. This file will be called 'FILENAME.DBG' where filename is the name of the script file being parsed. To enable this feature, the Debug mode parameter in the GLOBAL parameters (Configure System menu) needs to be set to 4 or a value with bit 2 set. The output file will contain lines with the following parameters:

Parameter Name	Value/Description
BLOCK	The block number. Starting at 1
RED$	The Red value in range 0-64000 (64000 is max)
GREEN$	The Green value in range 0-64000 (64000 is max)
BLUE$	The Blue value in range 0-64000 (64000 is max)
MS$	The block period in ms
DIM$	0=Bright, 1=Dim range
FLAGS$	Flags Bits see FLAGS$ section

Conversion of cd/m^2 to cd.s/m^2 for flash calculations

All LED luminance values must be entered in cd/m^2 format regardless of pulse duration. To convert to cd.s/m^2 you must divide the cd/m^2 value by 250. This is because the maximum pulse period is 4 ms or 1/250 of a second. To actually produce this luminance, you must set the block period to 4ms e.g:

```
BLOCK      COLOR$      RED            LUM$      1000        MS$         4
```

This will produce a luminance of 4cd.s/m^2. If you do not specify a block length, then the default is 1ms, which will produce a luminance of 1cd.s/m^2.

Double Xenon flashes

Script programs make it very easy to produce double xenon flashes (or any other number). The following program produces two xenon flashes or variable intensity with variable period between them.

```
GLOBAL     description$  Double XENON pulse with variable interval between pulses. Triggers on
                         each pulse
GLOBAL     V1name$       X1 cd.s/m^2      V1Default$  100
GLOBAL     V2name$       X2 cd.s/m^2      V2Default$  1000
GLOBAL     V3name$       Interval ms      V3Default$  100

BLOCK      xenon$        &1
BLOCK      ms$           &3-1             ;ms-1 because block takes 1ms to perform
BLOCK      xenon$        &2
```

The minimum time between flashes is 3 ms as the flash takes a maximum of 2 ms and the interval between the end of one flash and the start of the next is 1 ms. Note also that the period between the flashes has 1ms subtracted from it as the block takes 1ms to decode. To flash the xenons on a background color, simply set the background color and intensity from the Espion program directly.