NAME

spyview − program for visualizing 2d data

SYNOPSIS

spyview files...

DESCRIPTION

spyview is a program designed for viewing two dimensional data. It's inspiration is an old window's program called Spyglass Transform, whose origins lie in an even older unix X11R4 program called XImage.

Upon file loading, data is first loaded into a double precision floating point array. A sequence of image processing operations is optionally performed on the floating point data, and then the data is quantized to 16 bit integers and loaded in to the spyview data buffer. From there, the data is then colorized according to the settings in the main window and draw in the image window.

Exact control over the quantization step is possible through the "Load options" subwindow. The user can here control exactly how the floating point data is quantized, either by setting how much of the 16-bit dynamic range is used, or by manually setting the min and max of the quantization conversion. Convienent buttons are provided for picking a min and max based on the current slider settings in the spyview window.

Currently, spyview support the native SPM200 .Stm file format, as well as PGM files and MTX files (see MTX section below) and regular ASCII DAT files (see DAT section below). For PGM files, X, Y, and Z scaling factors can be included as a comment in the pgm file. These scaling factors will be used to display physical units on crossection plots. Unit names can also be included. The following is an example of how to set these comment fields:

P5
491 401
#zmin -7.146000e-04
#zmax 4.742000e-04
#zunit Capacitance (a.u)
#xmin 0
#xmax 9.8
#xunit Mag Field (B)
#ymin -2
#ymax 2
#yunit Bias (V)
65535
(raw data....)

These can also be set using the gui using the "Image Units" window. Changes to the units in ranges using the gui can also be saved in an MTX file using the "Save MTX" button on the controls window.

Spyview supports a wide range of colormaps, and gives the user complete control over the transfer function that maps the 2d 16 bit data onto 8,8,8 bit RBG data for drawing to the screen.

Colormaps are read in as a 24 bit color PPM file, that is 1 column by X rows, where X is the number of colors in the map. (Default colormaps are 256 colors.) System wide colormaps are stored in /usr/share/spyview/cmaps/. Colormaps are also loaded by default from a directory ˜/cmaps/ and are appended to the colormap list in the GUI.

The basic transfer function for mapping the data is:

imagedata = 0 if (dataval<min)

= X*((dataval-min)/width) ˆgamma if (min<dataval<max)

= X if (dataval>max)

where X is the number of colors in the colormap. These numbers are then mapped onto a 8,8,8 bit RGB color using a colormap lookup table.

The user can also normalize the image. When normalizing, exactly "bpercet" of the pixels will get mapped

to black (0), and "wpercent" will get mapped to white. bpercent and wpercent can be controlled by the settings in the spyview control window GUI.

Plane subtraction support is also included. A plane can be fit to the data, and the plane parameters can be adjusted in real time by the user.

The software also has support for plotting horiztonal, vertical and arbitrary angle cross sections of the data, as well as the histogram of the currently displayed data (ie. min<dataval<max) and the colopmap. Arbitrary angle cross sections use bilinear interpolation.

Separate x and y integer zooms are also implemented (the zoom is implemented by replicating pixels so that there is no distortion of the data). The zooms can be set using the keyboard shortcut. Spyview also supports "negative" integer zoom, inplemented by dropping datapoints. If you resize the spyview image window, spyview will automatically pick the closest integer zoom possible. The window frame can then be "snapped" to the image size by using the "Shift S" keyboard shortcut.

GUI SHORTCUTS

Button 1 Clicking in the image window will give you a horizontal cross section. Type 'c' in the image window to clear the cross section. (Type q in the gnuplot window to close the plot.)

Button 2 Same as button 1, but gives a vertical cross section.

Ctrl-Button 1 Draw a new arbitrary angle cross section.

Ctrl-Button 2 Adjust the nearest arbitrary angle cross section endpoint.

Ctrl-Button 3 Drag around the currently selected arbitrary angle crossection.

c Clear the cross section.

Button 3 Toggle hiding/showing of controls window.

Left/Right Arrow Go to next/previous image.

Up/Down Arrow Go to next/previos colormap.

n Normalize the image.

, Decrease the x and y integer zoom by one.

. Increase the x and y integer zoom by one.

Shift-, Decrease the x integer zoom by one.

Shift-. Increase the x integer zoom by one.

Ctrl-, Decrease the y integer zoom by one.

Ctrl-. Increase the y integer zoom by one.

Shift S Snap the window frame to the current image size.

1 Make the x an y zooms equal: the smaller one will be adjusted to match the larger one.

s Toggle the "square" setting (see below)

Alt-s Save the image as a ppm.

p Toggle plane subtraction.

Shift-Button 1 Drag over a section of the image to zoom in on.

Shift-Button 2 Set the center of the zoom window, and turn it on if necessary

z Toggle the zoom window

Shift-. in Zoom Increase the x zoom in the zoom window by one.

Shift-, in Zoom Decrease the x zoom in the zoom window by one.

Ctrl-. in Zoom Increase the y zoom in the zoom window by one.

Ctrl-, in Zoom Decrease the y zoom in the zoom window by one.

. or + or = in Zoom Increase the x and y zoom in the zoom window by one.

, or - in Zoom Decrease the x and y zoom in the zoom window by one.

GUI SETTINGS

square If the width of the image is an integer multiple of the height, make the image square by averaging pixels together horizontally.

This option is designed for our scanning capacitance experiment, where we oversample (ie. 1 or more points per TC) the image horizontally so that we can do a boxcar average instead of increasing the lockin TC to average, which gives us better signal to noise (since the exponential in the TC average throws away more data information than the boxcar average).

Negate Invert the data before applying the colormap (flips colormap transfer function left to right).

Invert Invert the data after applying the colormap (flips colormap transfer function top to bottom).

IMAGE PROCESSING

Spyview also has support for many image processing operations. These are controlled by the "Image Processing" window. On the left of the window, there is a list of the image processing operations currently built into spyview. By clicking on one of the operations, you can read the description of the operations in the status line at the bottom of the image processing window.

The box on the right of the window shows the current queue of image processing operations. Each time a file is loaded, the specified sequence of image processing operations is performed, and the resulting data is loaded into the spyview image window. Added a new operation, or changing the parameters of any of the operations will cause the data to be reloaded.

Image operations can be deleted, moved up, or moved down in the queue by selecting them and using the buttons in the middle of the window. A given operation can also be temporarily disabled by delecting the "Enable filter" check box in the lower left of the window.

As a long queue of image operations can take a long time to set up, and can also take a long time to execute, it is convenient to save the processed data to disk. This can be done using the "Export MTX" button in the main window, which will save the processed data to a new file. (The original data will not be overwritten! See below for more info on Export MTX.)

ASCII DATAFILE SUPPORT

ASCII data files can now be loaded directly into spyview. The data can be arranged in a matlab-compatible "matrix" like format or in a gnuplot "datblock" and "datindex" format (see gnuplot documentation for more details).

When loading gnuplot files, 3D data can be loaded. Here there are two options: if your data has only data blocks (2D gnuplot format), spyview will load each of the columns in the dataset into the third dimension of a 3D dataset (so that Z=0 is the 2D data made from column 1, Z=1 from column 2, etc).

The second gnuplot file option is to use datablocks as one axis, and data indices as the second axis. In this case, the user must select the column in the gnuplot file that will be used to generate the data.

MTX DATAFILE SUPPORT

mtx (short for matrix) is a simple binary floating point file format written for 3D data matrices. Spyview supports loading and displaying of floating point data from these files. An MTX file consists of a text

header followed by binary "float" or "double" data (written from fwrite() in ia32 format). The header consists of two lines. The first (optional) line contains comma-delimited information about the physical dimensions of the data set:

"Units", Dataset name, xname, xmin, xmax, yname, ymin, ymax, zname, zmin, zmax

The next line contains four numbers:

nx ny nz length

where nx, ny and nz are the sizes of the x, y and z dimensions, and legnth is the number of bytes in each datapoints (4 for float, 8 for double).

OUTPUT OPTIONS

Spyview has many options for saving the displayed data to file. The names output files are controlled by the text in the "basename" box, which by default is set to the current filename minus a .Stm or .pgm extension if present. Upon saving, the appropriate extensions are added to this base name.

For all export options, the format is specified by the drop down box. The available output formats include:

Save PPM (.ppm)

This saves the current color image displayed in the image window to a "portable pixmap" image (equivalent to a uncompressed .bmp). This can be easily converted to more common formats using the netpbm utilities (such as pnmtopng using "pnmtopng file.ppm > file.png").

Save MTX (.mtx)

This will save the current data as a .mtx file, replacing the original data if it exists!!! This is useful if you want to change the axis ranges and names in a datafile.

Export PGM (.pgm)

This will export the current data to a 16-bit pgm file, including the comments in the header for the axis names and ranges.

Export MTX (.export.mtx)

This will export the current data as a .mtx file. It will not overwrite the currently loaded file.

Export GP (.gnu and .gp)

This will generate a ASCII data file from the current loaded data in a format that is compatible with the gnuplot 3d-data input file. The data is saved in a file ${basename}.gp. Spyview will also generate a file ${basename}.gnu, which is a gnuplot script that will generate a 2D colorscale plot of the .gp data using the exact colormap that was display in spyview, using the X and Y axis scales and labels specified in the "Image Units" window. (Run it by typing "load file.gnu" in gnuplot.)

Export DAT (.dat)

This will generate an ascii matrix data file of the current data, which can easily be loaded into matlab using the "load('file.dat')" command, or plotted in gnuplot as well using the "splot 'file.dat' mat" command.

Postscript (.ps)

Spyview also has support for generating some nice, compact postscript from the color scale plot in the image window. By clicking on the "Postcript" button, you will bring up the postscript control panel, where many properties of the postscript output can be controlled.

The postscript generated will include a framed box with X and Y axes. The ranges of the X and Y axes are determined by the ranges set in the Image Units control window, as are the labels on the X and Y axes. A title can be included in the plot, and by default it's value will be the name of the currently loaded file, as indicated in the file chooser widget.

A directory stamp can also be included on the postscript output: in this case, a line of text will be added to the bottom right corner of the page indicating the full path the directory that spyview was launched from.

The generate button will generate the output postscript file. The "View postscript" button will also generate the postscript output, and will also then launch the "gv" postscript viewer to display the output, from which the postcript can easily be printed by typing "p". Note: gv must be installed on your system. "gv" is launched with the --watch option, so that if you remake the postscript file, gv will automatically refresh when it notices the file changed. A "Live Preview" mode is also available on unix, in which gv is sent a HUP signal to force it to reread the file, avoiding the delay in how frequently gv checks the file.

Three pages options are available: letter paper, a4 paper, and EPS. For EPS setttings, the directory stamp and the landscape setting are both ignored. For EPS, a bounding box is also generated in the postscript file from an estimate of the size of the figure. (For a more accurate bounding box, consider using gs -sDEVICE=bbox and manually inserting into the EPS file.)

EXTERNAL COMMANDS

If a keypress not associated with a spyview command is hit in the image window, and the environment variable SPYVIEW_EXTERNAL is set, then the program specified by SPYVIEW_EXTERNAL is run as follows:

$SPYVIEW_EXTERNAL *key_pressed x_coord y_coord x_pixel y_pixel*

i.e.

$SPYVIEW_EXTERNAL r 1.23 -3.232 11 23

If SPYVIEW_EXTERNAL returns success (0), the image is then reloaded.

OPTIONS

Run spyview with no arguments for an option list.

SEE ALSO

spyview3d(1), **spybrowse**(1), **dat2pgm**(1)

AUTHOR

This manual page was written by Gary Steele <gsteele@electron.mit.edu>, for the Debian project (but may be used by others).