GRASS Display Commands

(pages with * need updating)

d.3d
d.ask
d.barscale
d.colormode
d.colors
d.colortable
d.db
d.display
d.dm
d.erase
d.extend
d.font
d.fix.ortho
d.frame
d.geodesic
d.graph
d.grid
d.his
d.histogram
d.icons
d.label
d.labels
d.legend
d.linegraph
d.mapgraph
d.measure
d.menu
d.mon
d.paint.labels
d.pan

d.points
d.profile
d.rast
d.rast.arrow
d.rast.edit
d.rast.labels
d.rast.num
d.rgb
d.rhumbline
d.save
d.scale
d.site.labels
d.siter
d.sites
d.sites.qual
d.text
d.text.freetype
d.title
d.vect
d.vect.area
d.vect.line
d.vect.labels
d.what.db
d.what.rast
d.what.sites
d.what.vect
d.where
d.zoom

SG3d
nviz
nviz tutorial

xganim

Other Commands

help home, database, display, drivers, general, grid3d, imagery, import, misc, models, paint, photo, postscript, raster, scripts, sites, vector

NAME

d.3d − Displays three−dimensional images based on raster map layers.
(GRASS Display Program)

SYNOPSIS

d.3d
d.3d help
d.3d [**−lan**] **map**=*name* **elevation**=*name* [**from_coordinate**=*x,y,z*] [**to_coordinate**=*x,y,z*] [**exaggeration**=*value*] [**lines**=*value*] [**field**=*value*] [**color**=*name*] [**box**=*name*]

DESCRIPTION

d.3d displays three−dimensional graphic images based on GRASS raster map layers. The user identifies the viewing point, the line of sight, a vertical exaggeration factor, the viewing angle (field of view), the frequency and color of vector grid lines to appear in the display, the map to be displayed in 3−d, and the map whose category values are to be used as elevation values in the 3−dimensional image.

The program will be run non−interactively if the user specifies all needed parameter values and flag settings on the command line. Alternately, the user can simply type **d.3d** on the command line; in this case, the program will prompt the user for parameter values and flag settings using the standard interface described in the manual entry for *parser*.

Three−dimensional images can also be generated through the GRASS *d.display* program.

OPTIONS

Flags:

−l

Display *lines* only, without displaying the raster base *map*.

−a

Each grid−cell is rendered using four elevation coordinates: one for each corner. By default, each corner takes the value of the grid−cell to the lower right. With the −a option each corner is calculated to be an average of the four bordering cell elevation values.

−n

Show null elevations.

Parameters:

map=*name*
The raster map layer used to generate the color shown in the 3−d output (i.e., the map whose x and y values will be displayed, using the z values present in the *elevation* map layer.)

elevation=*name*
The raster map used to generate the texture in the 3−d image (i.e., the map whose category values will become the elevation values in the 3−d map displayed). This need not be an elevation map layer, although elevation is commonly used.
Default: elevation

from_coordinate=*x,y,z*
Coordinates of the viewing point, given as: *northing,easting,elevation*. The default *from_coordinate* value is appropriate for the spearfish sample data set.
Default: Calculated at run−time just south−west of the south−west corner.

to_coordinate=*x,y,z*
Coordinates of center of view, given as: *northing,easting,elevation*. The default *to_coordinate* value is appropriate for the spearfish sample data set.
Default: Calculated at run−time as the center of the image

exaggeration=*value*
Vertical exaggeration factor. This value is multiplied by the "elevation" values (category values) in the *elevation* map layer. As the vertical coordinates are exaggerated, the elevation (z) value of the center of view (the *to_coordinate*) should be increased.
Default: 2.0

lines=*value*
North−south and east−west trending lines can be drawn on the 3−d image output, to enhance the 3−d effect. The *lines* value specifies the number of rows and columns to skip between lines. 0 means "display no lines". Lines will be drawn in the *color* stated. The user can elect to only display these lines, without displaying the *map*, by setting the −l flag.
Default: 1

field=*value*
The field of view (viewing angle) in the image, stated in degrees.
Default: 30

color=*name*
Color of vector *lines* drawn on the output.
Options: color, white, red, orange, yellow, green, blue, indigo, violet, gray, black
Default: gray

box=*name*
Color of bounding box
Options: red,orange,yellow,green,blue,indigo,violet, white,black,gray,brown,magenta,aqua,grey,none
Default: none

The resulting three−dimensional image is drawn in the active frame on the graphics monitor. The user should select and erase the full graphics monitor frame before running *d.3d* to prepare the screen for graphics. Refer to the GRASS program *3d.view.sh* for a demonstration of the *d.3d* program.

INTERACTIVE MODE

The interactive mode prompts for the raster map to be rendered, the raster map to be queried for elevation information, and an optionally saved set of saved viewing parameters. The user is then presented with a form in which the options can be modified.

It is suggested that the following procedure be used to efficiently find the correct viewing parameters. Start with a gross grid (by default, a resolution is chosen which limits the total number of rows and columns), display lines only, set line colors to "color" (lines take their color from the map), set the box color to something other than "none", and use the default viewing coordinates. Experiment with the viewing coordinates until the map is positioned in a desireable configuration. When the map is lined up correctly, set the display resolution as low as you like for a final image.

BUGS

Several additions to this program have been suggested. These include the following:

1. Border the image with a visually appealing curtain.
2. Add x,y,z coordinates and scale information to the displayed 3−d image.
3. Provide a graphic−oriented user interface for identifying viewing parameters.
4. Display output more quickly by doing mathematics in integers.
5. Add the option of plotting vector map layers.

SEE ALSO

3d.view.sh,
d.display,
d.erase,
d.frame,
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/10/31 08:40:47 $

NAME

d.area − Draws selected polygons to the GRASS display
(GRASS Display/Vector Program)

SYNOPSIS

d.area [**−f**] **map**=*name* [**fillcolor**=*name*] [**linecolor**=*name*] [**catnum**=*value[,value,...]*]

Flags:

−f
Fill all areas with color (cycle through standard GRASS colors)

Parameters:

map
Name of existing vector map to be displayed
fillcolor
Color desired for filling polygons
Default: white
linecolor
Color desired for polygons borders
Default: none
catnum
List of area category number(s) to display

DESCRIPTION

Use *d.area* to draw filled polygons from a vector map to the GRASS display. Using, the "*−f*" flag, the program will cycle through the standard GRASS named colors, overriding any *fillcolor* argument. Otherwise, the color used for filling polygons is as specified.

The arguments for *fillcolor* and *linecolor* can be any named X11 color (as found in grass5/etc/rgb.txt), a hexidecimal color string (like "AFDF00"), or an RGB triplet specified as "rgb(*red green blue*)". Hexidecimal color strings must be six characters where each pair represents the red, green and blue colors, respectively. RGB values must be between 0 and 255 for each component. Do not use any separator besides a space between the RGB numbers.

The user can optionally choose to restrict drawing operations to selected polygons by specifying one or more area category numbers using the *catnum* argument.

NOTES

When run on the command line, any X11 color name containing spaces or any rgb(...) specification must be quoted to prevent the shell from splitting it into separate arguments.

The routine for handling holes in polygons is relatively slow. Therefore, large maps containing a large number of holes will suffer a significant performance penalty.

It is the intention of GRASS developers to create a better mechanism for persistent storage of vector drawing instructions in the future.

BUGS

The program does not properly clip areas to match the current display extent (**HELP WANTED**).

There is a slight disagreement between the output of d.vect and *d.area* regarding the map coordinate to screen coordinate conversion. This leads to undesirable results when both are used.

AUTHOR

CERL
Modified by Eric G. Miller

Last changed: $Date: 2003/03/16 12:25:19 $

NAME

d.ask − Prompts the user to select a GRASS data base file from among files displayed in a menu on the graphics monitor.
(GRASS File Management Program)

SYNOPSIS

d.ask help
d.ask element=*name,description* [**prompt=**"*message*"]

DESCRIPTION

d.ask is designed for shell scripts that need to prompt the user for the name of a data base file in the user's current GRASS mapset search path. After *d.ask* is invoked with needed parameters, the mouse becomes active and a menu containing files of the specified *element* type is displayed on the user's graphics monitor. The user is prompted by the prompt "*message" to select one of the listed file names with the mouse. Specifically, the query that appears to the user takes the form:*

> ***Double** click on the prompt "message"*
> ***Double** click here to cancel*
>
> *(A list of files of the specified element type from each of the mapsets listed in the user's mapset search path is also displayed.)*

After the user responds, the mouse is deactivated and the displayed menu is erased from the screen (leaving any underlying materials on display intact). Three lines are written to standard output (the user's terminal screen):

> ***name**=file_name*
> ***mapset**=mapset_name*
> ***fullname**=file_name@mapset_name*

Parameters:

***element**=name,description*
> *Name of a GRASS data base element, followed by a one word description of the element. GRASS data base elements are mapset subdirectories; these include: bdlg, cats, cell, cell_misc, cellhd, colr, colr2, dig, dig_ascii, dig_att, dig_cats, dig_plus, dlg, group, hist, icons, noise, paint, site_lists, and windows. The description will be used to display an error message to the screen if no files of the named element type exist in the user's mapset search path. The prompt "message" will appear in the pop−up menu displayed on the user's graphics monitor. The files listed in the menu will be of the specified element type, and exist in mapsets listed in the user's current mapset search path.*

prompt=*"message"*

> *A brief message with which the user will be prompted. If this message contains more than one word, it should be enclosed within double quotes (""). It is a non−selectable message displayed in the menu bar of a pop−up menu. Selectable items in the displayed menu include a cancel option and any of the files in the mapset subdirectory name in the user's mapset search path.*

EXAMPLE

Given the following input,

> ***d.ask element***=*cell,raster* ***prompt***=*"raster map layer to be used"*

d.ask will prompt the user to select a raster (cell) file from among those listed in the "cell" directories of the mapsets listed in his current mapset search path. A pop−up menu will be displayed on the user's graphics monitor, containing the prompt:

> ***Double*** *click on the raster map layer to be used*
> ***Double*** *click here to cancel*

OUTPUT

If the user selects a file name from the displayed menu, d.ask writes three lines to standard output:

> ***name***=*file_name*
> ***mapset***=*mapset_name*
> ***fullname***=*file_name@mapset_name*

The output is in the form of /bin/sh commands to set the variable *name* to the file name specified by the user (of the *element* type requested by *d.ask), mapset* to the GRASS mapset in which this file resides, and *fullname* to the name of the file and its mapset.

If the user elects not to select a file but instead chooses the "Double click here to cancel" option, the mouse will be deactivated and no variable assignments will be returned to standard output.

If no files of the specified *element* type are found in the user's current mapset search path, the mouse is activated and the following message is displayed on the user's graphics monitor:

> ***No element description files found***
> ***Click here to continue***

NOTES

Parameter values (the element name and description, and prompt message) cannot be supplied to d.ask interactively; they must be supplied on the command line or from a file.

SEE ALSO

g.ask,
g.filename,

g.findfile,
g.gisenv,
g.mapsets

BUGS

This program calls other GRASS programs which may generate confusing error messages. Example: an error message from the program d.menu will appear if a nonexistent map element is chosen.

AUTHOR

Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:32 $

NAME

d.barscale − displays a barscale on GRASS monitor
(GRASS Display Program)

SYNOPSIS

d.barscale [**−m**] [**−f**] [**−t**] [**bcolor**=*name*] [**tcolor**=*name*] [**at**=*x,y*]

DESCRIPTION

d.barscale displays a barscale on GRASS monitor either with mouse or with given screen coordinates.

OPTIONS

Flags:

−m
Use mouse to interactively place scale

−f
Use feet/miles instead of meters

−t
Write text on top of the scale, not to the right

Parameters:

bcolor
Color used for the background, or "none"
Options: red, orange, yellow, green, blue, indigo, white, black, brown, magenta, aqua, gray, grey
Default: black

tcolor
Color used for the text
Options: red, orange, yellow, green, blue, indigo, white, black, brown, magenta, aqua, gray, grey
Default: white

at
The screen coordinates for top−left corner of label
Options: 0−100
Default: 0.0,0.0

SEE ALSO

d.scale
d.legend
d.measure
d.what.rast
d.what.sites
d.what.vect
d.where
g.region
parser

AUTHOR

unknown.

Last changed: $Date: 2003/08/25 07:43:08 $

NAME

db.columns − list all columns for a given table
(GRASS Database Program)

SYNOPSIS

db.columns [**driver**=*name*] [**database**=*name*] [**location**=*name*] **table**=*name*

Parameters:

driver
Driver name

database
Database name

location
Database location

table
Table name

SEE ALSO

db.createdb, db.databases, db.describe, db.drivers, db.dropdb, db.droptable, db.execute, db.tables

AUTHOR

?

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

db.connect − Set up connection to the database through DBMI.
(GRASS DB Program)

SYNOPSIS

db.connect
db.connect help
db.connect [**−p**] [**driver=***name*] [**database=***name*] [**location=***name*] [**user=***name*] [**password=***name*] [**key=***name*]

DESCRIPTION

db.connect allows the user to set parameters for connection to database. These parameters are then taken by modules as default values and user do not need enter parameters each time. Values are stored in user file.

OPTIONS

When invoked simply as **db.connect**, the program prompts the user for parameters. The user can run the program non−interactively, specifying the (optional) flag setting and parameters values on the command line. Program flag and parameters are described below.

Flags:

−p
Print only current settings.

Parameters:

driver=*name*
The name of a DBMI driver. Currently only the **odbc** driver is available. You can use *db.drivers* to list available drivers.

database=*name*
The name of a database. You can use *db.databases* to list available databases.

location=*name*
The location of the database.

user=*name*

The database user name (login).

password=*name*

The database user password. **Warning: password is saved in .grassrc in readable form!!!**

key=*name*

The key column name. Key column is column corresponding to categories.

EXAMPLE

db.connect driver=*odbc* **database**=*gtest* **key**=*id*

SEE ALSO

db.drivers db.databases

AUTHOR

Radim Blazek

Last changed: $Date: 2002/04/09 09:19:22 $

NAME

db.createdb − create an empty database
(GRASS Database Program)

SYNOPSIS

db.createdb **driver**=*name* **database**=*name* **location**=*name*

Parameters:

driver
Driver name

database
Database name

location
Database location

NOTE

Module does not work with ODBC driver.

SEE ALSO

db.columns, *db.databases*, *db.describe*, *db.drivers*, *db.dropdb*, *db.droptable*, *db.execute*, *db.tables*

AUTHOR

?

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

db.databases − list all databases for a given driver
(GRASS Database Program)

SYNOPSIS

db.databases [**−l**] [**driver**=*name*] [**location**=*name[,name,...]*]

Flag:

−l

Output database location also

Parameters:

driver

Driver name

location

Database location(s)

SEE ALSO

db.columns, db.createdb, db.describe, db.drivers, db.execute, db.tables

AUTHOR

?

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

db.describe − describe a table (in detail)
(GRASS Database Program)

SYNOPSIS

db.describe [**driver**=*name*] [**database**=*name*] [**location**=*name*] **table**=*name*

Parameters:

driver
Driver name

database
Database name

location
Database location

table
Table name

SEE ALSO

db.columns, db.databases, db.dropdb, db.droptable, db.execute, db.tables

AUTHOR

?

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

db.drivers − list all installed DBMI drivers
(GRASS Database Program)

SYNOPSIS

db.drivers [**−f**]

Flag:

−f
Full output

AUTHOR

?

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

db.dropdb − remove a database
(GRASS Database Program)

SYNOPSIS

db.dropdb **driver**=*name* **database**=*name* [**location**=*name*]

Parameters:

driver
Driver name
database
Database name
location
Database location

NOTE

Module does not work with ODBC driver

SEE ALSO

db.createdb, *db.databases*, *db.describe*, *db.droptable*, *db.execute*, *db.tables*

AUTHOR

?

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

db.droptable − remove a table from database
(GRASS Database Program)

SYNOPSIS

db.droptable [**driver**=*name*] [**database**=*name*] [**location**=*name*] **table**=*name*

Parameters:

driver
Driver name

database
Database name

location
Database location

table
Table name

NOTE

Module does not work with ODBC driver

SEE ALSO

db.columns, db.cteatedb, db.databases, db.describe, db.dropdb, db.execute, db.tables

AUTHOR

?

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

db.execute − execute any SQL statement
(GRASS Database Program)

SYNOPSIS

db.execute [**driver**=*name*] [**database**=*name*] [**location**=*name*] [**input**=*filename*]

Parameters:

driver

Driver name

database

Database name

location

Database location

input

Filename with sql statement

SEE ALSO

db.columns, *db.createdb*, *db.databases*, *db.describe*, *db.drivers*, *db.dropdb*, *db.droptable*, *db.tables*

EXAMPLE

```
echo 'create table soils ("cat" int, "soiltype" text)' | db.execute
```

```
cat file.sql | db.execute
```

AUTHOR

CERL

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

db.select − Select data from database.
(GRASS DB Program)

SYNOPSIS

db.select
db.select help
db.select [**−c**] [**−d**] [**−h**] [**input**=*file*] [**fs**=*character(s)*] [**vs**=*character(s)*] [**nv**=*character(s)*]

DESCRIPTION

db.select prints result of selection from database based on SQL statement read from input file or from standard input to standard output.

Parameters:

−c

  Include column names in output.

−h

  Horizontal output (instead of vertical).

input=*file*

  Name of a *file* containing SQL select statement.

fs=*character(s)*

  Output field separator, default: |.

vs=*character(s)*

  Output vertical record separator.

vn=*character(s)*

  Null value indicator.

SEE ALSO

db.connect

AUTHOR

?
Modifications and man: Radim Blazek, Radim.Blazek@dhv.cz

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

db.tables − list all tables for a given database
(GRASS Database Program)

SYNOPSIS

db.tables [**−s**] [**driver**=*name*] [**database**=*name*] [**location**=*name*]

Flag:

−s
System tables instead of user tables

Parameters:

driver
Driver name
database
Database name
location
Database location

SEE ALSO

db.columns, db.createdb, db.databases, db.dropdb, db.droptable, db.execute

AUTHOR

?

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.colormode − Allows the user to establish whether a map will be displayed using its own color table or the fixed color table of the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.colormode
d.colormode help
d.colormode mode=*name*

DESCRIPTION

d.colormode establishes what color table will be used to display maps to the graphics monitor.

Parameters:

mode=*name*
Options: *fixed* or *float*

Each time a new raster map layer is drawn on the screen, a color table associated with that map is loaded into the graphics display monitor. The command **d.colormode mode=fixed** requests that the monitor's color look−up table be *fixed* (i.e., static). The effect is that colors drawn on the screen by graphics calls will not change when subsequent maps are drawn to the screen. When the *fixed* option is used, the colors in the map's color table are mapped to the nearest colors available in the monitor's fixed color table. When the *float* option is used, the map's color table is loaded directly into the monitor's color look−up table.

There are advantages to using each color display mode. The *d.display* and *d.colors* programs allow the user to interactively modify the color tables of maps displayed to the graphics monitor. This is done by allowing the user to directly modify the monitor's color look−up table. Hence, to accommodate this option in *d.display* and *d.colors*, the *float* option is chosen. However, this option has the sometimes undesirable effect of changing the colors in which other maps are subsequently displayed to the graphics monitor (although it does not actually change the *color tables* of these latter maps). The *float* colormode is therefore best used when the user wishes to interactively change a map's color table, or when the user wishes to display one or more maps having the same color table. The *fixed* option allows any number of maps to be displayed to the screen, where each map uses different colors, but all use the same fixed color look−up table. You cannot toggle a map's color table when running in fixed mode.

This program will be run non−interactively if the user specifies the color *mode* on the command line (e.g., by typing **d.colormode mode=fixed** or **d.colormode mode=float**). Alternately, if the user simply types **d.colormode** on the command line, the program will prompt the user for the color mode using the standard GRASS *parser* interface.

NOTES

Some devices support only a fixed color look−up. Whether or not XDRIVER supports float mode depends upon the X display in use; modes with more than 256 colors usually only support fixed mode. The number of color categories capable of being displayed is also device−dependent; if your colors fall outside this range when in *float* mode, they may not be *displayed* in the colors indicated in the map's color table file. However, the color table files themselves will accurately represent the user's changes.

Color table files associated with raster map layers are stored in the user's current mapset under the *colr* and *colr2* directories.

BUGS

It is strongly recommended that the user erase the graphics monitor screen (e.g., by running *d.erase*) immediately after changing the mode between *fixed* and *float.*

SEE ALSO

d.colors,
d.colortable,
d.display,
d.erase,
d.rast,
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2003/08/20 08:10:12 $

NAME

d.colors − Allows the user to interactively change the color table of a raster map layer displayed on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.colors
d.colors help
d.colors map=*name*

DESCRIPTION

A color table file associates specific colors with the categories of a raster map layer. The user can change these map category color assignments (i.e., change the map's color table) interactively, by first displaying the raster map to the graphics monitor and then running the program *d.colors.* If you have first set *d.colormode* to "float" before displaying the map to the graphics monitor, any color changes that you subsequently make while in *d.colors* will immediately (and interactively) appear on the graphics display.

While running *d.colors*
the cursor must be inside the window where the execution takes place. When in "float" colormode, the correct colors appear on the graphics monitor only if the cursor is inside the monitor. To see the correct colors, move the cursor from the execution window to inside the graphics monitor, then move back to the window to continue making changes.

If *d.colormode* is "fixed", any color changes made using *d.colors* will not immediately be shown on the graphics display; however, any color changes saved will still alter the map's color table and will appear next time the raster map layer is redisplayed (see *d.colormode* and *d.colortable).* Note that some graphics display monitors or drivers cannot support interactive color change.

The user must first display the relevant raster map layer to the active frame on the graphics monitor (e.g., using *d.rast* or *d.display)* before running *d.colors.* The user can then either enter the name of the raster map layer whose color table is to be changed on the command line (e.g., by typing: **d.colors** map=soils), or type **d.colors** without program arguments. If the user simply types **d.colors** without program arguments on the command line, *d.colors* will ask the user to enter the name of an existing raster map layer using the standard GRASS interface described in the manual entry for *parser*.

In either case, the user is then presented with the *d.colors* command menu, shown below. This menu is the same as the category and color changing portion of the *d.display* menu. The *d.colors* commands are listed beneath the Category Pointer Movement, Color Modification, Replotting Screen, and Quitting sections below. Commands are invoked by typing in the single−key response shown to the left below. (Longer descriptions of these commands appear to the right.) Results from invoking these commands will be reflected in the Category and Category Number sections of the *d.colors* screen. On the *d.colors* screen menu, commands appear in the right half of the screen, and the current status of categories appears in the left half of the screen.

```
CATEGORIES
   0    No Data
   1    (Category 1 description)
   2    (Category 2 description)
   .    ....
   .    ....
CATEGORY NUMBER:
   RED          0   0%
   GREEN        0   0%
   BLUE         0   0%
Shift Incr:   10   3%
CATEGORY POINTER MOVEMENT
   D/d  down (cats)        Move pointer to next category
   U/u  up   (cats)        Move pointer to previous category
COLOR MODIFICATIONS
   R/r  RED                Increase/decrease RED intensity
   G/g  GREEN              Increase/decrease GREEN intensity
   B/b  BLUE               Increase/decrease BLUE intensity
   I/i  increment          Increase/decrease increment
                           (of intensity shift)
   h    highlight          Highlight current color
   +/−  shift colors       Shift entire color table (up/down)
   c    save color         Save color table
   t    toggle table       Toggle to different color table
REPLOTTING SCREEN
   *    Replot screen      Replots the screen
QUITTING
   Q    quit               Quits program
```

Changing categories − The keys **"d, D, u,"** and **"U"** are used to move to a different category. The lower case letters move up, **u,** and down, **d,** the category list one category at a time. The upper case letters move 10 categories at a time for fast movement. The cursor does wrap between the first and last categories. The current category is noted on the text screen with an arrow, and is indicated on the graphics screen by a box around the current color.

Changing colors − The color associated with the current category can be changed with the **"R, r, G, g, B,"** and **b** keys. The upper case letters increase the intensities of red **R,** green **G,** and blue **B** for the current category; the lower case letters decrease the intensities of these same colors for the current category. Video devices make all the colors of the spectrum by mixing red, green, and blue. For those accustomed to red, yellow, and blue being the primary colors, this can be confusing. For starters, yellow is made by mixing red and green. The intensities are listed on the text screen in as percentages.

Keys **I** and **i** increase and decrease the percentage change that each keystroke of one of the color keys (*R, r, G, g, B, b*) causes in its respective color. The increase increment is initially set to 10%. Thus, pressing the *R* key would will increase the red component of the current category by 10%.

Highlight − The **h** key toggles between the current category color and the current highlight color. This color is initially black but can be modified as above while in highlight mode. Blinking can be accomplished by repeatedly striking the **h** key. When changing to different categories using the movement keys as described above, while in highlight mode the category colors will be always left showing their actual colors. Only one category is highlighted at any one time.

Saving the current color table − Pressing the **c** key will save the current color table as you have modified it. This table will then be used next time you display or paint this raster map layer.

Color table toggle − Different types of color tables are suitable for different raster map layers. The key **t** flips between the following color tables: red, green, blue color ramp; gray scale; smooth changing color wave; random colors; and the saved color table.

Color table shift − The entire table is shifted up and down using the **+** and **−** keys.

Quitting the *d.colors* program − Pressing the **Q** key will cause you to quit the *d.colors* program. If colors have been modified but not saved, *d.colors* will ask:

```
Colors changed
Save the changes? (y/n)
```

The user should type in **y** to save changes, or **n** to not save changes, before quitting the program. If the user types **n**, the program will ask:

```
Quit anyway? (y/n)
```

NOTES

To see map color changes reflected on the display monitor as you make them, you must put the monitor in *float* color mode before running *d.colors*. It is also wise to erase the display screen after changing the color mode.

The map whose color table is to be altered with *d.colors* must already be on display in the active display frame on the graphics monitor before *d.colors* is run. This can be done using the command **d.rast** **map=***name* (where *name* is a raster map layer whose color table the user wishes to alter).

The user might type the following sequence of commands to interactively change the colors of a raster *soils* map:

d.colormode **mode=***float*
d.erase
d.rast **map=***soils*
d.colors

After the user has saved any color changes made with *d.colors* and exited the program, the user could then redisplay the *soils* map with the new colors by typing:

d.colormode mode=fixed
d.erase
d.rast map=soils

Some color monitors may not support the full range of colors required to display all of the map's categories listed in the map's color table. However, regardless of whether the user can see the color changes he is effecting to a map's color table, any changes to a map's color table made with *d.colors* that are saved will appear in the map's color table.

Some monitors may not support an interactive color change capability.

SEE ALSO

d.colormode
d.colortable
d.display
d.rast
p.colors
r.colors
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2003/08/20 08:10:12 $

NAME

d.colortable − To display the color table associated with a raster map layer.
(GRASS Display Program)

SYNOPSIS

d.colortable
d.colortable help
d.colortable map=*name* [**color**=*name*] [**lines**=*value*] [**cols**=*value*]

DESCRIPTION

The GRASS program *d.colortable* is used to display the color table associated with a raster map layer in the active frame on the graphics monitor. The map *name* should be an available raster map layer in the user's current mapset search path and location.

Parameters:

map=*name*
: Name of a raster map layer in the user's current mapset search path whose color table is to be displayed.

color=*name*
: Color of the lines separating the colors of the color table.
Options: red, orange, yellow, green, blue, indigo, violet, gray, white, and black.
Default: *white*

lines=*value*
: Number of lines to appear in the color table.
Options: 1 − 1000

cols=*value*
: Number of columns to appear in the color table.
Options: 1 − 1000 If the *values* of both *lines* and *cols* are not specified by the user, *d.colortable* divides the active frame equally among the number of categories present in the named raster map layer. If one option is specified, the other is automatically set to accommodate all categories. If both are specified, as many categories as possible are displayed.

The user can specify all needed parameters on the command line using the form:

> **d.colortable map**=*name* [**color**=*name*] [**lines**=*value*] [**cols**=*value*]

If the user specifies the name of a map on the command line but does not specify the values of other parameters, parameter default values will be used. Alternately, if the user types simply **d.colortable** on the command line without any program arguments, the program will prompt the user for parameter settings using the standard GRASS parser interface described in the manual entry for *parser*.

EXAMPLE

The user running the command:

> **d.colortable map**=*soils* [**color**=*red*] [**lines**=*1*] [**cols**=*3*]

would see the active graphics frame divided into three columns extending the full frame height. The lines dividing the color table associated with the *soils* map would be displayed in red. The user would see, at most, only three of the colors from the *soils* color table displayed in the active frame (because the user requested that this frame be divided into three sections).

NOTES

If the user wishes to display the entire color table associated with a map, the user should either stipulate a number of lines (rows) and columns (cols) sufficient to accommodate the number of categories in the map's color table, or fail to assign values to one or both of *lines* and/or *cols*. If the user runs *d.colortable* using the default number of lines and columns (the full graphics frame), all categories from the map's color table will be displayed. However, if the user requests that the color table associated with a map which has 10 data categories be displayed in a graphics frame with only 3 lines (rows) and 2 columns (a total of six cells), only six of the ten map categories will be displayed.

The user should run the command **d.colormode mode=**float before running *d.colortable*, for accurate display of the map's color table.

The user should run the GRASS program *d.erase* between runs of *d.colortable* to avoid confusion.

SEE ALSO

d.colormode
d.colors
d.display
d.erase
d.rast
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2003/08/20 08:10:12 $

NAME

dcorrelate.sh − Graphically displays the correlation among from two to four raster map layers in the active frame on the graphics monitor.
(GRASS Shell Script)

SYNOPSIS

dcorrelate.sh *layer1 layer2* [*layer3* [*layer4*]]

DESCRIPTION

dcorrelate.sh is a C−shell (csh(1)) script that graphically displays the results of an *r.stats* run on two raster map layers. This shell script is useful for highlighting the correlation (or lack of it) among data layers (scattergram).

The results are displayed in the active display frame on the user's graphics monitor. *dcorrelate.sh* erases the active frame before displaying results.

OPTIONS

Parameters:

layer1 layer2 [layer3 [layer4]]
The names of from two to four existing raster map layers to be included in the correlation.

NOTES

This is a shell script that uses *r.stats* and the UNIX *awk* command to calculate the correlation among data layers, and uses *d.text* and *d.graph* to display the results.

If three or four map layers are specified, the correlation among each combination of two data layers is displayed.

This command is written for */bin/csh*. If your system doesn't support this shell, don't install this script.

FILES

This program is simply a shell script. Users are encouraged to make their own shell script programs using similar techniques. See **`$GISBASE/scripts/dcorrelate.sh`**.

SEE ALSO

The UNIX *awk* command

d.text, *d.graph*, *r.coin*, *r.stats*

AUTHOR

Michael Shapiro, U.S.Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/06/16 15:29:18 $

NAME

d.db − Displays sites from DB table in the active graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.db
d.db help
d.db table=*table_name* **x=***column_name* **y=***column_name* [**where=***SQL_where_clause*] [**color=***name*] [**size=***value*] [**icon=***name*]

DESCRIPTION

d.db displays sites from table in database with coordinates in x, y columns using site markers of a color, size, and type specified by the user. Output is displayed in the active graphics monitor.

Parameters:

table=*name*
Name of a *table* in the database.

x=*column_name*
Name of a *column* containing x coordinates.

y=*column_name*
Name of a *column* containing y coordinates.

where=*SQL_where*
SQL WHERE clause without 'where'.

color=*name*
Sets the current color to *name*.
Options: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white, and black
Default: *gray*

size=*value*
Size, in pixels, in which the site icons are to be drawn.
Options: 0−1000
Default: 5

icon=*name*
The type of icon to be displayed at site locations.
Options: x, diamond, box, +
Default: x

SEE ALSO

db.connect

AUTHOR

Radim Blazek, Radim.Blazek@dhv.cz

Last changed: $Date: 2002/01/25 05:45:32 $

NAME

d.display − A menu−driven, highly interactive display program for viewing maps and producing final map products
(GRASS Display Program)

SYNOPSIS

d.display

DESCRIPTION

The GRASS program *d.display* is used to display maps on a graphics monitor and prepare final map products for printing. It is advisable to first run *d.erase* before each run of *d.display* to prepare the screen for graphics.

After invoking the command **d.display**, the DISPLAY MAIN MENU appears on the monitor. Options in this and subsequent *d.display* menus are selected through use of the pointing device (mouse). When the mouse cursor is over the desired menu selection, the user simply presses any button on the mouse.

d.display is a macro that allows the user to access a wide range of GRASS display functions in a menu−driven and highly interactive environment. Most of the functions accessible through *d.display* can also be run independently of one another by running the GRASS display (*d.) commands listed in the SEE ALSO section, below. However, in many cases, these other d. commands give the user greater flexibility and make available more options than are available through the d.display menus. For example, the d.display label file option runs the d.label command using defaults. Users running d.label directly from the command line have the choice of stipulating specific label background and text colors, as well as text sizes and font types. Users running the label file option through d.display cannot specify these options; instead, default values are used (see d.label for default values used).*

The *d.display* tool is best learned by playing with its options. Refer to the individual *d.* and other commands in the SEE ALSO section for a detailed examination of these functions.

NOTES

The user can also modify the current region settings from the DISPLAY MAIN MENU. Using the mouse, the user can choose to *zoom* in on a portion of the displayed raster map layer to create a smaller geographic region. Once the user has defined the corners of this area using the mouse, the smaller area is enlarged and replotted on the screen. The user can similarly choose to *zoom* out. In this case, the user uses the mouse to encompass that portion of the reduced map that the user wishes to make the current geographic region and display in the center of the screen. The larger map is then replotted and displayed. Finally, the user can enter geographic coordinates for the region's corners from the keyboard using the GRASS command *g.region*. This is done by selecting the *type* coordinates option.

The main menu also offers the user the option of hiding the main menu for ten seconds (*hide menu for 10 seconds*).

The *d.display* command must be run in conjunction with a program which converts the GRASS graphics commands generated by *d.display* to the graphics commands that the current device understands. This device will almost always be a graphics monitor. On most systems the user is required to start up this other program separately from (and prior to) *d.display*. In such cases, a locally−defined program, usually "monitor", is used (see your GRASS manager). (For example, on MASSCOMP machines, the *d.mon* program must be running in the foreground on the graphics monitor. On SUN machines, the *d.mon* program can be started from any terminal and run in the background.)

d.display sends program output to the current graphics frame (set, by default, to the full graphics screen unless modified by the user). You can use *d.frame* to create and choose a smaller display frame on the graphics monitor for use by *d.display*.

See the GRASS macro *3d.view.sh* stored under $GISBASE/scripts for an example of how you can create a *d.display−like* macro using a shell script.

SEE ALSO

See *3d.view.sh* or *slide.show.sh* under **`$GISBASE/scripts`** for examples of *d.display−like* macros.

d.3d
d.colormode
d.colors
d.frame
d.label
d.legend
d.measure
d.mon
d.rast
d.scale
d.sites
d.vect
d.zoom
g.region
d.menu
v.digit

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:32 $

NAME

d.dm − Display manager. GUI tool for easy display/zoom/query of map sets. (*GRASS Display Program*)

SYNOPSIS

d.dm

DESCRIPTION

d.dm may be started by typing "d.dm &" in GRASS shell. **MOUSE−OVER HELP IS AVAILABLE FOR MOST OF THE WINDOW PARTS!** Window of d.dm is devided into more frames:
Sets frame: in upper left corner, contains list of map sets
Maps frame in upper right corner, contains list of maps in selected set
Set/map buttons on the left side below Sets/Maps frames, used for manipulating with sets and maps
Save/Close buttons on the right side below Sets/Maps frames
Monitors below Set/Map, Save/Close buttons
Display/query buttons on the bottom of the d.dm window

Manipulating Sets

After first start of d.dm in GRASS mapset, empty Set named 'Set0' is inserted into Sets frame. User can **rename** set by double click on set name and retyping text. **New set** may be added by pressing the button **Add set**. Existin Set may be **selected** by single click on set name. Selected Set may be **removed** by pressing **Del set** button.

Manipulating Maps

List of maps in selected Set is displayed in Maps frame. New map may be added by pressing the button **Add map**, and selecting map type from displayed list of types. Map row appeares in Map frame. Each map row starts by map type button. The map may be selected by pressing the map type button. Next item is text field **map name** which is name of map as is used in GRASS, for example roads@PERMANENT or rivers. Map name may be selected from list of available maps opend by right mouse button click on the map name field. Following items in the map row depend on the map type. Selected map may be removed by pressing **Del map** button. Selected maps may be moved up and down by pressing buttons with up and down arrow symbols.

Selecting monitors

Monitor is selected in Monitors frame by radio button. Monitor does not need to be opend and will be opened automaticaly.

Display, Zoom

Display button is used to display all maps in selected map Set. **All** button resets region to full default region. **Zoom** and **Pan** buttons are used for zooming in monitor.

Query

Query button is used to query selected map.

PNG

PNG button creates a file named 'map.png' containing the selected maps in the current directory. If such a file already exists it will be overwritten ! Does not handle legend files. Issues such as true color support and background color have to be set with environmental variables (see PNG−driver for details).

Save, Close

d.dm session may be saved by **Save** button. The session is saved in $LOCATION/$MAPSET/.d.dmrc file. This file is read in on next session. d.dm is quit by **Close** button (without saving).

OPTIONS

No options.

BUGS

Error codes returned by GRASS commads are not handled.

SEE ALSO

d.mon

AUTHOR

Radim Blazek

Last changed: $Date: 2003/02/09 12:07:23 $

NAME

d.erase − Erases the contents of the active display frame on the user's graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.erase
d.erase help
d.erase [**color=***name*]

DESCRIPTION

d.erase erases the contents of the active graphics frame, and replaces it with the color black (by default) or by whatever color is specified by the user. *d.erase* will not alter the assignment of the active frame.

Parameter:

color=*name*
Color with which active frame will be erased.
Options: red, orange, yellow, green, blue, yellow, indigo, violet, black, white, gray, brown, and magenta
Default: black

SEE ALSO

d.frame
d.mon

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:32 $

NAME

d.extend − Set window region from currently displayed raster, vector and sites maps with largest map region.
(GRASS Display Program)

SYNOPSIS

d.extend
d.extend help
d.extend [−v]

DESCRIPTION

d.extend can be used to display all objects in currently displayed maps.

OPTIONS

Flag:

−v

Run verbosely.

NOTES

If no raster map is displayed on a monitor, the default region resolution is calculated to fill window by 512 * 512 cells. Otherwise, it will use the minimum resolution of displayed raster maps.

SEE ALSO

g.region

AUTHOR

Huidae Cho <grass@geni.cemtlo.com>
Hydro Engineering Research Laboratory, Kyungpook National University
South Korea

Last changed: $Date: 2002/05/10 07:00:00 $

NAME

d.font − Selects the font in which text will be displayed on the user's graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.font
d.font help
d.font font=*name*

DESCRIPTION

d.font allows the user to select use of a specific text font for display of text on the graphics monitor. The GRASS program *show.fonts.sh* is a UNIX Bourne shell macro which names and displays the fonts that can be selected using *d.font*. If the user does not specify a font when using other GRASS programs that display text, the font type *romans* is used by default.

The user can run this program either non−interactively or interactively. If the user specifies a font type name on the command line the program will run non−interactively. Alternately, the user can simply type **d.font** on the command line; in this case, the program will prompt the user for a display text font type using the standard GRASS interface described in the manual entry for *parser*.

Parameter:

font=*name*

Name of a font type, from among the font types italicized below.
Default: *romans*
Options: (italized)
cyrilc Cyrillic
gothgbt Gothic Great Britain triplex
gothgrt Gothic German triplex
gothitt Gothic Italian triplex
greekc Greek complex
greekcs Greek complex script
greekp Greek plain
greeks Greek simplex
italicc Italian complex
italiccs Italian complex small
italict Italian triplex
romanc Roman complex
romancs Roman complex small
romand Roman duplex

romanp Roman plain
romans Roman simplex
romant Roman triplex
scriptc Script complex
scripts Script simplex

NOTES

The font type *romans* is the fastest font type to display to the graphics monitor.

SEE ALSO

d.text
d.title
show.fonts.sh
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

d.font uses the public domain version of the Hershey Fonts created by Dr. A.V. Hershey while working at the U.S. National Bureau of Standards.

Last changed: $Date: 2002/01/25 05:45:32 $

NAME

d.frame − Manages display frames on the user's graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.frame
d.frame help
d.frame [**−cepslD**] [**frame=***name*] [**at=***bottom,top,left,right*]

DESCRIPTION

This program manages display frames on the user's graphics monitor. GRASS display programs at run−time connect with graphics rendering programs. While the display programs are identical on every hardware platform, the graphics rendering programs are (essentially the only GRASS programs) designed for individual hardware devices. These rendering programs are managed with the GRASS program *d.mon*. Graphics are displayed in rectangular frames on whatever graphics monitor the user is currently directing GRASS display output to. These frames are created and managed with this program, **Note** that GRASS frame contents are not retained when one frame covers another. You cannot shuffle frames from top to bottom and then back again. They simply define rectangular areas on the screen where subsequent drawing will occur.

Flags:

−c

Creates a new display frame on the graphics monitor.

−e

Removes all existing display frames and reinitializes the entire graphics screen (the full−screen display frame).

−p

Prints the name of the active frame, in which GRASS display output will appear.

−s

Selects a frame for the display of GRASS graphics. This frame is then known as the "active frame".

−l

List map names displayed in GRASS monitor

−D

Prints the status of the user's graphics monitor and active display frame to standard output. Information includes the name and the dimensions of the current frame on the graphics monitor, given in the form *bottom* top left right. This function is useful for debugging output, and for determining display screen coordinates.

Parameters:

frame=*name*
The name of the display frame to be created/selected.

at=*bottom,top,left,right*
Where to place the frame (implies **−c**). Frame coordinates are stated in the form: *bottom,top,left,right*. The lower−left corner of the graphics monitor always is at location 0,0 while the monitor's upper−right corner is always at 100,100 (the values are in percent).

NOTES

If the user has created multiple display frames that overlap one another, whatever the user displays in the active frame will overwrite those portions of the underlying frame where these frames overlap.

SEE ALSO

d.erase
d.mon

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory
Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2003/03/16 12:25:19 $

NAME

d.geodesic − Displays a geodesic line, tracing the shortest distance between two geographic points along a great circle, in a longitude/latitude data set.
(GRASS Display Program)

SYNOPSIS

d.geodesic
d.geodesic help
d.geodesic [**coor=***lon1,lat1,lon2,lat2*] [**lcolor=***name*] [**tcolor=***name*]

DESCRIPTION

d.geodesic displays a geodesic line in the active frame on the user's graphics monitor. This line traces the shortest distance between two user−specified points on the curved surface of a longitude/latitude data set. The two coordinate locations named must fall within the boundaries of the user's current geographic region.

OPTIONS

This program can be run either interactively or non−interactively. If the user types **d.geodesic** on the command line without other program parameters, the mouse will be activated; the user is asked to use the mouse to indicate the starting and ending points of each geodesic line to be drawn. The default line color (white) and text color (red) will be used.

Alternately, the user can specify the starting and ending coordinates of the geodesic, line color, and text color on the command line, and run the program non−interactively.

Once the user indicates the starting and ending coordinates of the geodesic, the line and its length (in miles) are displayed to the user's graphics monitor.

Parameters:

coor=*lon1,lat1,lon2,lat2*
Starting and ending coordinates, in longitude and latitude values, of the geodesic line to be drawn.
lcolor=*name*
Line color in which geodesic will be displayed.
Options: red, orange, yellow, green, blue, magenta, indigo, violet, gray, white, black
Default: white
tcolor=*name*
Text color in which the length of the geodesic will be displayed.
Options: red, orange, yellow, green, blue, magenta, indigo, violet, gray, white, black. Default: red

NOTES

This program works only on GRASS data bases using a longitude/latitude coordinate system.

SEE ALSO

d.rhumbline

AUTHOR

Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:32 $

NAME

d.graph − Program for generating and displaying simple graphics to the graphics display monitor.
(GRASS Display Program)

SYNOPSIS

d.graph
d.graph help
d.graph [**input=***name*] [**color=***name*]

DESCRIPTION

d.graph draws graphics that are described either from standard input (default), or within a file (if an input *file* name is identified on the command line). If graphics commands are entered from standard input, a *ctrl−d* is used to signal the end of input to *d.graph*.

The program can be run interactively or non−interactively. The user can run the program completely non−interactively by specifying the name of a graphics *file* containing *d.graph* graphics commands and the values of all needed parameters on the command line. The user can instead elect to run the program partially interactively, by specifying any/all of the parameters *except* the graphics **file=***name* parameter on the command line. In this case, *d.graph* will expect the user to input *d.graph* graphics commands from standard input (i.e., the keyboard) and will (silently) prompt the user for these graphics commands.

Alternately, the user can simply type **d.graph** on the command line, and be prompted for the values of all parameters (the user can still input graphics commands from either an input file or standard input using this form). In this case, the user is presented with the standard GRASS *parser* interface.

Parameters:

file=*name*
Name of file containing graphics commands. This is a UNIX file name; the file must be located in the user's current working directory or be specified by its full path name. If no filename is specified commands are taken from the standard input.

color=*name*
Starting color desired for graphics.
Options: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white, and black.
Default: *white*

The coordinate system used is 0−100 in x and 0−100 in y, regardless of the graphics monitor display frame size and aspect. The (0,0) location is the lower left corner of the active graphics monitor display frame. All values may be floating point.

The graphics language is simple, and uses the following commands:

# comment

A line of comment which is ignored in the processing.

move *xpos ypos*

The current location is updated to *xpos ypos. Values are stated as a percent of the active display frame's horizontal (xpos) and vertical (ypos) size, and may be floating point values. Values are between 0−100.* ***Note.*** *A space must separate xpos and ypos.*

draw *xpos ypos*

A line is drawn in the current color from the current location to the new location *xpos* ypos, which then becomes the current location. Values are stated as a percent of the active display frame's horizontal (*xpos) and vertical (ypos) size, and may be floating point values. Values are between 0−100.* ***Note.*** *A space must separate xpos and ypos.*

color *color*

Sets the current color to that stated; subsequent graphics will be drawn in the stated color, until the current color is set to a different color. Options are *red, orange, yellow, green, blue, indigo, violet, brown, magenta, gray, white,* and *black.*

size *xper yper*

Subsequent text will be drawn such that the text is *xper* percent of the graphics monitor display frame wide and *yper* percent of the frame high. By default, the text size is set to 1 percent of the active frame's width and 1 percent of the frame's height if unspecified by the user (this may be too small to be seen by the user).

Note. A space must separate *xper* and *yper.*

text *line−of−text*

The stated text is drawn at the current location using the current color, and the new current location is then positioned at the end of the text string.

icon *type size x y*

Draws an icon of types *o, x,* or *+* with specified *size* at location *x,y*. Note: type *o* designates a square.

polygon
xpos ypos
xpos ypos
...

The coordinates appearing beneath the word *polygon*, one pair per line, circumscribe a polygon that is to be filled with the current color.

EXAMPLE

For an example use of *d.graph*, examine the contents of the macro command *$GISBASE/bin/grass.logo.sh* located in the GRASS command bin (which draws a GRASS logo by inputting *d.graph* graphics commands that are stored in a shell file). Note that the coordinates in the *grass.logo.sh* macro were taken directly off an image drawn by hand on graph paper.

NOTES

d.graph remembers the last screen location (xpos ypos) to which the user moved, even after the user erases the display frame. If the user runs *d.graph* repeatedly, and wishes to start anew with the default (xpos ypos) screen location, the user should *clear* the display frame between runs of *d.graph*.

LIMITATIONS

There are no automated ways of generating graphic images. It is anticipated that GRASS user sites will write programs to convert output from a resident graphics editor into GRASS *d.graph* format.

SEE ALSO

d.font
d.mapgraph
d.text
grass.logo.sh
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:32 $

NAME

d.grid − Overlays a user−specified grid in the active display frame on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.grid
d.grid help
d.grid size=*value* [**color**=*name*] [**origin**=*easting,northing*]

DESCRIPTION

d.grid overlays a grid of user−defined size and color in the active display frame on the graphics monitor. The grid will overlay, not overwrite, the contents of the active display frame.

d.grid can be run non−interactively or interactively. If the user specifies the grid *size* and (optionally) the grid *color* on the command line the program will run non−interactively; if no grid *color* is given the default will be used. Alternately, the user may simply type **d.grid** on the command line; in this case, the program will prompt the user for parameter values using the standard GRASS user interface described in the manual entry for *parser*.

Parameters:

size=*value*
Size of grid to be drawn in the active display frame, in current map coordinate system units. It should be noted that, depending on the scale of the map layer displayed in this frame, the grid may fall outside the user's viewing range.
Options: 0−100000

color=*name*
Sets the current grid color to *name*.
Options: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white, and black.
Default: *gray*

origin=*easting,northing*
Lines of the grid pass through this coordinate. The coordinate need not be within the current frame.

NOTES

d.grid will not erase grids already displayed in the active graphics display frame by previous invocations of *d.grid*; multiple invocations of *d.grid* will therefore result in the drawing of multiple grids inside the active graphics frame. (A command like *d.erase*, which erases the entire contents of the active display frame, must be run to erase previously drawn grids from the display frame.)

Currently, the grid size can only be drawn in the units of the map coordinate system used by the current GRASS location.

SEE ALSO

d.display
d.erase
d.frame
d.legend
d.rast
d.scale
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:32 $

NAME

d.his − Displays the result of combining hue, intensity, and saturation (HIS) values from user−specified input raster map layers.
(GRASS Display Program)

SYNOPSIS

d.his
d.his help
d.his [**−n**] **h_map**=*name* [**i_map**=*name*] [**s_map**=*name*]

DESCRIPTION

his stands for hue, intensity, and saturation. This program produces a raster map layer providing a visually pleasing combination of hue, intensity, and saturation values from two or three user−specified raster map layers.

The human brain automatically interprets the vast amount of visual information available according to basic rules. Color, or *hue*, is used to categorize objects. Shading, or *intensity*, is interpreted as three−dimensional texturing. Finally, the degree of haziness, or *saturation*, is associated with distance or depth. This program allows data from up to three raster map layers to be combined into an image which retains the original information in terms of *hue*, *intensity*, and *saturation*.

OPTIONS

This program can be run non−interactively or interactively. It will run non−interactively if the user specifies on the command line the name of a map containing hue values (**h_map**), and the name(s) of map(s) containing intensity values (**i_map**) and/or saturation values (**s_map**). The resulting image will be displayed in the active display frame on the graphics monitor.

Alternately, the user can run the program interactively by typing **d.his** without naming parameter values on the command line. In this case, the program will prompt the user for parameter values using the standard GRASS parser interface.

Flags:

−n

Respect NULL values while drawing

Parameters:

h_map=*name*
Name of input raster map layer to be used for HUE values.
i_map=*name*
Name of input raster map layer to be used for INTENSITY values.
s_map=*name*
Name of input raster map layer to be used for SATURATION values.

While any raster map layer can be used to represent the hue information, map layers with a few very distinct colors work best. Only raster map layers representing continuously varying data like elevation, aspect, weights, intensities, or amounts can suitably be used to provide intensity and saturation information.

For example, a visually pleasing image can be made by using a watershed map for the *hue* factor, an aspect map for the *intensity* factor, and an elevation map for *saturation*. (The user may wish to leave out the elevation information for a first try.) Ideally, the resulting image should resemble the view from an aircraft looking at a terrain on a sunny day with a bit of haze in the valleys.

THE PROCESS

Each map cell is processed individually. First, the working color is set to the color of the corresponding cell in the map layer chosen to represent *HUE*. Second, this color is multiplied by the *red* intensity of that cell in the *INTENSITY* map layer. This map layer should have an appropriate gray−scale color table associated with it. You can ensure this by using the color manipulation capabilities of d.display, d.colors or r.colors. Finally, the color is made somewhat gray−based on the *red* intensity of that cell in the *SATURATION* map layer. Again, this map layer should have a gray−scale color table associated with it.

NOTES

The name is misleading. The actual conversion used is

```
  H.i.s + G.(1−s)

where

  H   is the R,G,B color from the hue map
  i   is the red value from the intensity map
  s   is the red value from the saturation map
  G   is 50% gray (R = G = B = 0.5)
```

Either (but not both) of the intensity or the saturation map layers may be omitted. This means that it is possible to produce output images that represent combinations of *his, hi,* or *hs*. Users wishing to store the result in new raster map layers instead of displaying it on the monitor should use the program *r.his*.

SEE ALSO

d.colormode, *d.colors*, *d.colortable*, *d.display*, *r.colors*, *d.his*, *i.his.rgb*, *i.rgb.his*, *hsv.rgb.sh*, *rgb.hsv.sh*, *parser*

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/06/16 15:29:18 $

NAME

d.histogram − Displays a histogram in the form of a pie or bar chart for a user−specified raster file.
(GRASS Display Program)

SYNOPSIS

d.histogram
d.histogram help
d.histogram [**−qnC**] **map=***name* [**color=***name*] [**style=***name*] [**nsteps=***value*]

DESCRIPTION

d.histogram displays the category−value distribution for a user−specified raster map layer, in the form of a bar chart or a pie chart. The display will be displayed in the active display frame on the graphics monitor, using the colors in the raster map layer's color table. The program determines the raster file's category value distribution by counting cells.

OPTIONS

The user can run this program either non−interactively or interactively. The program will be run non−interactively if the user specifies the name of a raster map layer and (optionally) any other desired parameters and flags on the command line, using the form:

> **d.histogram** [**−zqnC**] **map=***name* [**color=***name*] [**style=***name*] [**nsteps=***value*]

If a map name is given on the command line, any other parameter values left unspecified on the command line will be set to their default values (see below). Alternately, the user can simply type **d.histogram** on the command line, without program arguments. In this case, the user will be prompted for needed inputs and option choices using the standard GRASS *parser* interface.

Flags:

−q
: Gather the histogram quietly, without printing messages on program progress to the user's terminal.

−n
: Display information for null cells.

−C
: Report for ranges defined in cats file (fp maps only).

Parameters:

map=*name*
: The name of an existing raster map layer in the user's current mapset search path for which a histogram is to be displayed.

color=*name*
: The name of the color to be used for the axis, text−labels, and tic−marks on the pie or bar chart.
Options: *white*, *red*, *orange*, *yellow*, *green*, *blue*, *indigo*, *magenta*, *violet*, *brown*, *gray*, and *black*
Default: *white*

style=*name*
: The style of the histogram. If **style=***bar*, then *d.histogram* will displays the statistics in the form of a bar chart. If **style=***pie*, then *d.histogram* will display the statistics in the form of a pie chart.
Options: *bar* or *pie*
Default: *bar*

nsteps=*value*
: Number of steps to divide the data range into (FP maps only).
Default: *255*

NOTES

d.histogram uses the current geographic region settings and the current mask (if one exists).

d.histogram uses the colors in the map's color look−up table (i.e., the map's *colr* or *colr2* file). To view results correctly on the display monitor, make sure the color mode is correctly set before running *d.histogram* (i.e., execute **d.colormode** **mode=float**).

d.histogram does not erase the active frame before displaying output.

SEE ALSO

d.colormode *d.colors* *d.colortable*
d.display *d.erase* *g.region*
r.mask *r.stats* *parser*

AUTHOR

Dave Johnson
DBA Systems, Inc.
10560 Arrowhead Drive
Fairfax, Virginia 22030

Last changed: $Date: 2002/01/25 05:45:32 $

NAME

d.icons − Displays points, as icons, at user−defined locations in the active display frame on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.icons
d.icons help
d.icons [**−r**] **icon=***name* [**color=***name*] [**size=***value*] [**points=***name*]

DESCRIPTION

d.icons graphically displays point (site) locations as icons in the active frame on the graphics monitor. Geographic coordinates are read either from standard input or from an input file whose name is stated by the user. At each site location, a user−defined icon is displayed.

OPTIONS

The user must enter at least the name of an *icon* file storing a graphic representation of the icon to be displayed, and the geographic coordinates of the points at which they will appear.

The user can run this program either non−interactively or interactively. The program will be run non−interactively if the user specifies the name of an *icon* file and (optionally) any other desired parameters on the command line, using the form:

> **d.icons** [**−r**] **icon=***name* [**color=***name*] [**size=***value*] [**points=***name*]

If the user fails to specify the name of a *points* file on the command line, the program will prompt the user to enter geographic coordinates from standard input. If the user specifies at least the name of an *icon* file on the command line, any other parameter values left unspecified by the user will be set to their default values (see below).

Alternately, the user can simply type:

> **d.icons**

on the command line, without program arguments. In this case, the user will be prompted for needed parameter values using the standard GRASS user interface described in the manual entry for *parser*. The geographic coordinates at which icons are to be displayed can still be input from an input file (*points*) in interactive use.

Flag:

−r
: Coordinates are input in reverse order (i.e., as *northing easting*).

Parameters:

icon*=name*
: The name of an existing file containing a graphic representation of the icon to be drawn. *icon* files can be created by the user using the *p.icons* command and are stored under the *icons* directory under the user's current mapset.

color*=name*
: Sets the current icon color to the *name* stated.
: Options: red, orange, yellow, green, blue, indigo, violet, gray, white, and black
: Default: white

size*=value*
: The icon scaling factor.
: Options: 1−1000
: Default: 1

points*=name*
: The name of a UNIX file containing the geographic coordinates of sites at which icons are to be drawn. Since this is a UNIX file, the user should specify the name using standard UNIX file naming conventions. If this file is not in the user's current working directory, its pathname should be specified. The file contents should consist of a series of geographic coordinates that fall within the boundaries of the current geographic region. Each site location should be stated on a separate line as an easting and northing (in that order) separated by a single blank space. If no *points* file is specified by the user, input is taken from standard input and should be given in the same form; to end standard input, type *end* (or press control−d).

EXAMPLE

An *icons* file contains a graphic representation of the icon to be displayed. Here, spaces represent areas of no color, and x's represent areas of color. For example, the user might construct an icon resembling a cross in the following way:

```
  x
  x
xxxxx
  x
  x
  x
```

This icon might be stored in a file called *cross* (under the user's $LOCATION/icons directory). If the user specified that this be the *icon* file used while running *d.icons*, this cross would then appear in the *color* and *size* specified by the user, at each site location named in the *points* file.

Note: icons are created with the *p.icons* command.

The *points* file lists the geographic coordinates of site locations (at which icons will be displayed). This file should take the form:

```
easting northing
easting northing
```

```
        easting northing
        ...
```

If the user sets the **−r** flag, the order of these coordinates should be reversed (i.e., coordinates should be given as *northing easting*).

If the points file is not specified, then the corrdinates are read from the keyboard or across a pipe. This feature allows users to enter the coordinates by hand, or, more usefully, to get then from another program. For example, to display icons at locations specified in a sites list:

> **s.out.ascii** *name* | **d.icons icon**=*cross*

where *name* is the name of a site list, and *cross* is the name of an icon.

SEE ALSO

d.points
d.sites
d.where
p.icons
s.db.rim
s.out.ascii
v.db.rim
parser

AUTHOR

Contributed by:

Dave Johnson
DBA Systems, Inc.
10560 Arrowhead Drive
Fairfax, Virginia 22030

Modified by:

Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:32 $

NAME

d.igraph − Graphics interface for GRASS on the Intergraph.
(Map Development Tool)

SYNOPSIS

d.igraph

DESCRIPTION

d.igraph is the GRASS interface to the GRASS graphics driver on the Intergraph workstations. GRASS can be run from an ascii terminal or from a window on the Intergraph workstation. There are some pitfalls to running GRASS from a window; these are discussed in the NOTES section below.

When *d.igraph* is run a pull−down menu will appear on the workstation screen and some information will be displayed on the ascii terminal. The pull−down menu, which works the same way other Intergraph pull down−menus work, will look like this:

```
_______________________
|  GRASS Commands       |
|  start/select         |
|  stop                 |
|  GRASS Graphics       |
|  resize/move          |
|  virtual screens    >|
|  colors             >|
| EXIT                  |
|______________________|
```

The menu is divided into two categories: *GRASS Commands* and *GRASS Graphics*. *GRASS Commands* in the pull−down menu have GRASS command−line equivalents. These command−line equivalents can be used without running *d.igraph*, although users may find the pull−down menu commands easier and faster to use. On the other hand, there are no command−line equivalents for the *GRASS Graphics* commands in the pull−down menu. These commands, which must be run from inside *d.igraph*, enable you to modify the size, placement, and number of colors, of the graphics window.

The first command under *GRASS Commands* is *start/select*; selecting this command with the mouse will start the graphics driver *IGRAPH* and then select it for output. This pull−down menu command is equivalent to typing the following commands at the GRASS command line:

d.mon start=IGRAPH
(start and select the IGRAPH graphics driver)

or

d.mon −s start=IGRAPH
(start but don't select the IGRAPH driver)
d.mon select=IGRAPH
(select the already started IGRAPH driver)

Using the *start/select* option saves the user from having to type the above command(s). After the graphics driver is started, the pull−down menu will disappear and the *d.igraph* program will exit, leaving the user at the GRASS prompt. When the second command (*stop*) under *GRASS Commands* is selected, the graphics driver *IGRAPH* will stop and the GRASS window will disappear. The equivalent GRASS command−line command is: **d.mon stop=IGRAPH**.

The second category (*GRASS Graphics*) in the pull−down menu contains three commands written for the GRASS graphics driver *IGRAPH*. These commands establish parameters for size, location, and number of colors the graphics driver will use to create a GRASS graphics window. Therefore, they should be run *before* starting the graphics driver with the *start/select* command. These parameters will not affect a GRASS window that has already been started, but they will be saved and used the next time the graphics driver is started.

The first command in the second category is *resize/move*. When the *resize/move* command is selected, the pull−down menu will disappear and a special *Resize* window, colored grey, will be created on the graphics monitor. The *Resize* window will be the exact size and at the same location as the GRASS graphics window would be if the graphics driver had been started. Use the resize and move icons on the *Resize* window to modify the window. When the desired size and position has been achieved, **delete** the *Resize* window. To do this, move the mouse to the left side of the icon strip at the top of the *Resize* window and click the mouse on the small box containing an **X**. Note that the window size is displayed on the ascii terminal in which the *d.igraph* program was started. Now, each time the graphics monitor is started, the window will be the same size and at the same location as the *Resize* window.

The second command is *virtual screens*. This command has a sub−menu with a listing of virtual screen numbers (from 0 to 3). Of the virtual screen numbers listed (0 to 3), only one or two of those numbers will be active and selectable from the sub−menu. The number of selectable virtual screen numbers will depend on the type of workstation and its virtual screen configuration. When the *IGRAPH* driver is started, it will create a GRASS window on the virtual screen whose number was picked by this command. Note that the current virtual screen number is displayed on the ascii terminal n which the *d.igraph* program was started.

The third graphics command is *colors*. This command has a sub−menu with a listing of the possible range of active colors the workstations will support (16 to 512). The number of active colors is determined by the number of planes the graphics processor supports (9 planes will give 512 active colors). We do not suggest running GRASS with fewer than **256** active colors.

Each workstation can be configured differently with regard to the number of virtual screens and the available number of colors. A certain amount of testing will have to been done to determine what works best with a particular workstation.

The last command is the *EXIT* command. This command is used to leave the *d.igraph* program.

USING THE GRAPHICS WINDOW

Once the GRASS graphics driver is started, a window will appear on the screen. The GRASS graphics window is mouse−activated. The mouse must be within the GRASS graphics window for the true GRASS colors to be displayed on the graphics window. The mouse doesn't have to be clicked inside the window; just moving the mouse within the window is enough to activate the window. While the mouse is outside the GRASS graphics window, the colors in the graphics window will be random.

NOTES

Obsolete for LINUX, of course!

When using the GRASS program *d.colors* or the *interact* selection in the *d.display* program, the mouse must be within the GRASS graphics window when toggling or highlighting categories. If the mouse is not within the graphics window, the colors will not represent the true category colors. Using the two programs mentioned above while running GRASS from an Intergraph window (ascii window) creates a problem; The mouse must be within the graphics window at all times for those programs to work properly, but the mouse also must be moved to the ascii window to activate the ascii window so that commands can be typed. This is not a problem when GRASS is run from an ascii terminal, because the mouse can be left within the GRASS graphics window at all times.

AUTHOR

Michael Higgins, U.S.Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:32 $

NAME

d.label − Creates and displays text labels in the active display frame on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.label
d.label help
d.label [**size=***value*] [**backcolor=***name*] [**textcolor=***name*] [**font=***name*]

DESCRIPTION

d.label allows the user to create and display text labels in the active frame on the graphics monitor. It interactively requests that the user type in text and use the pointing device (mouse) to identify where this text is to be placed within the active graphics frame. The program will prompt the user for label size, background color, text color, and text font type, if the user fails to specify these values on the command line. Program prompts use the standard GRASS parser interface.

Parameters:

size=*value*
Sets the label text size to the specified number. Values are stated as a percentage of the frame height; e.g., a size of 10 will make each line of text equal to one−tenth the height of the display frame.
Options: 0−1000
Default: 10

backcolor=*name*
Sets the color of the label background to the *name* stated.
Options: red, orange, yellow, green, blue, indigo, violet, gray, white, and black.
Default: *black*

textcolor=*name*
Sets the color of the label text to the *name* stated.
Options: same as for *backcolor*.
Default: *white*

font=*name*
Sets the font type used for the label's text to *name.*
Options: romand, romanp, romant, romans, scriptc, scripts, romancs, italicc, italiccs, gothitt, gothgrt, and gothgbt. The user can view these available fonts by running show.fonts.sh.
Default: romans (Roman simplex font type), or whatever font the user has set the font type to before entering d.label.

NOTES

This program will allow the user to display labels anywhere in the active display frame, even in areas lying outside of the current geographic region.

d.label will only allow the user to type in one line of text. The fully interactive program *d.labels* can be used to create text labels having as many as four lines.

d.label does not create paint labels files displayable by *d.paint.labels* −− *d.labels* does. These programs should be integrated in future GRASS releases.

BUGS

If the user is running GRASS under X Windows, and the user enters a text label longer than 80 characters (i.e., allows text to wrap onto a second line), X Windows will stop the user's GRASS graphics monitor without throwing the user out of GRASS. If this happens, the user should simply restart a graphics monitor, using *d.mon*.

SEE ALSO

d.display
d.font
d.labels
d.legend
d.mon
d.paint.labels
d.rast
d.scale
d.text
d.title
d.where
show.fonts.sh
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.labels − To create/edit GRASS paint label files for display on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.labels

DESCRIPTION

The *d.labels* program allows the user to interactively create and modify paint−labels files for use with the *p.map* and *d.paint.labels* programs. The *paint labels* files created will be stored in the user's current mapset under the **`/paint/labels`** directory.

d.labels is fully interactive, and requires use of a graphics monitor. After the user has typed

> **d.labels**

and has typed in the name of a new or existing paint−labels file, the user is asked to indicate label positions on the display frame using the mouse−pointer, and to edit label parameters that define the label's size, color, and other characteristics through use of a visual−ask (VASK) data entry screen. The mouse can also be used to indicate the geographic coordinates of locations within the current geographic region in the active frame. It is helpful to display a relevant data file in the active display frame before running *d.labels* to provide visual clues for label placement.

USER PROMPTS

You will first be prompted to enter the name of either an old paint−labels file that you wish to modify, or a new paint−labels file that you wish to create. The following is the file name prompt:

OPTION: Label text file name

> key: name
> required: YES
> enter option >
>
> If you enter the name of a new (non−existent) paint−labels file, you are asked to indicate a location falling within the current geographic region in the active display frame at which the label is to be placed.

Mouse Buttons
Left: Place Label Here

Middle: Where Am I ?
Right: Quit

Once you have indicated a position, you will be presented with a data−entry screen (also known as a VASK screen) so that you can enter the paint−label parameters that define how the label will appear on the display or on the color printer. Once you finish entering the label−parameters, and you hit the <ESC> key to proceed, the label will be displayed on the screen. If you like the appearance of the label, enter "y" to the "Look OK?" prompt; if not, enter "n" and you will again be presented with the data entry screen. Once you approve the label's appearance you will be asked to indicate the position of the next label.

If you choose to display an already−existing paint−labels file, you will be shown the labels. You will then have the option of editing a label or adding more labels.

Buttons
Left: Edit Existing Labels
Middle: Quit
Right: Add more Labels

On−line help is available. If you are puzzled about what should go in a data entry field, just enter "help" in that field, hit the <ESC> key, and you will be presented with a short description of the field and what you may enter.

The following is a list of the different fields in the data entry screen and a short description of each.

***TEXT**:*

The user can enter up to four lines of text.

***SKIP**: yes/no.*

If *no*, label will be printed. If *yes*, the label will be retained in the file but not printed.

***LOCATION**:*

Determines where the text will be located. The user specifies the easting and northing in map units, and (optionally) specifies a vertical and horizontal offset (in printer pixels) from the specified easting/northing. (The vertical (y) offset will shift the location to the south if positive, north if negative. The horizontal (x) offset will shift the location east if positive, west if negative.) These offsets are provided to allow finer placement of labels.

***RESET LOCATION**: yes/no.*

If *yes*, the user will be allowed to use the mouse to select a new location for the label. The mouse buttons will function as follows:
Mouse Buttons Left: Place Label Here Middle: Where Am I ? Right: Quit

***PLACEMENT**:*

Determines which part of the label to which the location refers. This may be specified as:
lower left (lower left corner of the text) lower right (lower right corner of the text) lower center (bottom center of the text)

upper left (upper left corner of the text) upper right (upper right corner of the text) upper center (top center of the text)

center (center of the text)

***TEXT SIZE**:*

Determines the size of the lettering used in the label. Text size (height) is stated in units of (ground) meters; e.g., a text size of 500 sets the height of each line of text equal to 500 ground meters. Thus, text will appear to grow or shrink when displayed, depending on the user's current geographic region settings.

TEXT COLOR:

This selects the text color. This color can be specified in one of three ways:

1. By color name:
   aqua black blue brown cyan gray green grey indigo magenta orange purple red violet white yellow
2. As red, green, blue percentages. For example: .5 .4 .7
   (This form is not supported by *d.paint.labels*, but see *p.map* and *p.labels*.)
3. By printer color number, to get the exact printer color.
   (This form is not supported by *d.paint.labels*, but see *p.map* and *p.labels*.)

Note. If the user sets the text color equal to the background color, the text will not appear on the graphics monitor.

WIDTH:

This determines the line thickness of the letters. The normal text width should be set to 1. Larger numbers can be used to simulate bold face. (*d.paint.labels* ignores this value and always uses 1.)

HIGHLIGHT COLOR:

The text can be highlighted in another color so that it appears to be in two colors. The text is drawn first in this color at a wider line width, and then redrawn in the text color at the regular line width. Highlight colors are not shown on the graphics display, they are only shown in the output from *p.labels* and *p.map*.

HIGHLIGHT WIDTH:

Specifies how far from the text lines (in pixels) the highlight color should extend. Highlight colors are not shown on the graphics display, they are only shown in the output from *p.labels* and *p.map*.

BACKGROUND COLOR:

Text may be boxed in a solid color by specifying a background color (see TEXT COLOR above to specify a color). Specify *none* to use no background.

OPAQUE TO VECTORS: *yes/no.*

This field only has meaning if a background color is selected. *yes* will prevent vector lines from entering the background. *no* will allow vector lines to enter the background.

BORDER COLOR:

Select a color for the border around the background. Specify *none* to suppress the border.

NOTES

This program is fully interactive and requires no command line arguments.

If the user types the word **help** in a data entry field while entering label information, acceptable field choices will be listed.

Labels must be placed within the current geographic region. If the user attempts to place a label outside of this region, the program will appear to loop endlessly. It is simply failing to display a label positioned beyond the boundaries of the current region; in this event, the user should simply reposition the label.

Labels created by this program are stored in ASCII files under the directory **`$LOCATION/paint/labels`**.

SEE ALSO

d.label
d.paint.labels
d.rast
d.vect
d.where
p.labels
p.map

AUTHOR

Dave Johnson,
DBA Systems, Inc.

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.legend − Displays a legend for a raster map layer in the active frame on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.legend
d.legend help
d.legend [**−vcnsmf**] **map**=*name* [**color**=*name*] [**lines**=*value*] [**thin**=*value*] [**labelnum**=*value*] [**at**=*bottom*,*top*,*left*,*right*] [**range**=*min*,*max*]

DESCRIPTION

d.legend displays a legend for a user−specified raster map layer in the active frame on the graphics monitor.

The user can run *d.legend* either non−interactively or interactively. If the user specifies the name of a raster **map** layer on the command line, the program will run non−interactively. Default legend text **color** and position will be used unless the user specifies other values on the command line.

Alternately, the user can simply type **d.legend** on the command line; in this case, the program will prompt the user for parameter values using the standard GRASS parser interface.

OPTIONS

Flags:

−v
Do not show category labels

−c
Do not show category numbers

−n
Skip categories with no label (ignored for floating point maps)

−s
Draw smooth gradient (forced automatically for floating point rasters)

−m
Use mouse to size & place legend

−f
Flip legend vertically or horizontally

Parameters:

map=*name*
    Name of raster map.
color=*name*
    Sets the legend's text color.
    Options: red, orange, yellow, green, blue, indigo, white, black, brown, magenta, aqua, gray, grey
    Default: white
lines=*value*
    Number of lines in the legend (useful for truncating long legends)
    Options: 0−1000
    Default: all
thin=*value*
    Thinning factor (thin=10 gives cats 0,10,20...)
    Options: 0−1000
    Default: 1
labelnum=*value*
    Number of text labels for smooth gradient legend
    Options: 2−100
    Default: 5
at=*bottom,top,left,right*
    Placement coordinates, as 0−100% of graphics frame height/width. 0,0 is lower left.
    Options: 0−100
    Default: automatically placed and scaled
range=*min,max*
    Category range
    Default: all

NOTES

The legend's default size is based on the dimensions of the active frame, specifically its height. *d.legend* will only obscure those portions of the active frame that directly underlie the legend.

When using the mouse or **at** to size & place the legend, if the user makes a horizontal box the values run from left to right and no category values are printed (you may want to use *d.label* for that). A horizontal legend is created if the box is wider than it is tall.

Raster maps based on floating point values will display smoothed, from greatest to smallest value, while categorial raster maps will display in order, from top to bottom. If the box is defined with inverted y−values or an inverted **range**, the legend will automatically flip. If this is not the desired result, the **−f** flag may be used to flip it back.

If the user attempts to display a very long legend in a relatively short display frame, the legend may appear in unreadably small text, or even revert to a smooth gradient legend. Use the **lines**, **thin**, **range**, and/or **−n** options to reduce the number of categories to be displayed, or the **−s** flag to force a smooth gradient legend.

The **lines** option will display the first number of categories, as defined by *value*, contained in the raster map. When used with the **−n** flag, it takes on a new meaning: "up to category #". When used with both **thin** and the **−n** flag, its meaning becomes more obscure. When using **lines**, auto−scaled text similar to "4 of 16 categories" will be placed at the bottom of the legend.

The **thin** option sets the thinning factor. For raster maps with a 0th category, **thin=***10* gives cats [0,10,20,...]. For raster maps starting at category 1, **thin=***10* gives cats [1,11,21,...].

The **range** option lets the user define the minimum and maximum categories to be used in the legend. It may also be used to define the limits of a smooth gradient legend created from a raster containing floating point values. Note the color scale will remain faithful to the category values as defined with *r.colors*.

The flag **−n** is useful for categorial maps, as it suppresses the drawing of non−existing categories (otherwise the full range is shown).

Vertical legends produced with *d.legend* will produce text labels to the right of the legend box. This text will be auto−scaled to fit within the frame, reducing the size of the legend if necessary. Legends positioned with the mouse or with the **at** option will not auto−scale text, in order to provide more control to the user. Smaller text may be obtained in this case by reducing the height of the box. The **−c** and **−v** flags may be used to suppress the display of category numbers and labels respectively, or used together to suppress all text of categorial raster maps.

The text produced from floating−point raster maps will automatically create output with a meaningful number of significant digits. For very small values, numbers will be expressed in scientific notation, e.g. "1.7e−9".

Note that old scripts which relied on setting **lines** greater than the number of categories to scale the legend may no longer produce the desired output, although the auto−scaling should still produce something that looks good in this case.

TODO

− Add **use** option for legends of discrete category numbers/values only.
− Text labels for horizontal legends.

SEE ALSO

d.colormode
d.colors
d.colortable
d.display
d.erase
d.grid
d.rast
d.rast.labels
d.rast.leg
d.vect
d.what.rast
r.reclass
parser

AUTHORS

Bill Brown, U.S. Army Construction Engineering Research Laboratories
Late 2002: Rewrite of much of the code. Hamish Bowman, Otago University, New Zealand

Additional improvements from various authors

Last changed: $Date: 2003/03/16 12:25:19 $

NAME

d.linegraph − Generates and displays simple line graphs in the active graphics monitor display frame.
(GRASS Display Program)

SYNOPSIS

d.linegraph
d.linegraph help
d.linegraph xfilename yfilename[,name,...] [directoryname] [ycoloroption[,option,...]] [xtitlevalue] [ytitlevalue] [titlevalue] [titlecoloroption]

DESCRIPTION

d.linegraph is a primitive program to draw simple x,y line graphs based on numerical data contained in separate files.

Parameters:

xfilename
    Name of file with data for the X axis of graph.
yfilename[,name,...]
    Name of file(s) with data for the Y axis of graph. Up to ten Y inputs can be used.

Data file format
The X and Y data files for the graph are essentially a column of numbers in each file, with one input number per line. The program expects that each X value will have a corresponding Y value, therefore the number of lines in each data input file should be the same. Essentially, the X data becomes the X axis reference to which the Y data is plotted as a line. Therefore, the X data should be a monotonically increasing progression of numbers (i.e. "1,2,3,..."; "0, 10, 100, 1000,..."; "...−5,−1,0,1,5..."). If multiple Y data files are used, the Y axis scale will be based on the range of minimum and maximum values from all Y files, then all Y data given will be graphed according to that Y scale. Therefore, if multiple Y data inputs are used with dissimilar units, the graph produced comparing the two will be deceptive.

directoryname
    Path to the directory where the input files are located. If this option is not used, the *d.linegraph* looks for files in the current directory.
    Example format: directory/usr/grass/data/graph
ycoloroption[,option,...]]
    Color to be used for drawing the lines in the graph. If multiple Y data files are used, an equal number of colors may be used to control the colors of the lines. Colors will be assigned to Y data in respect to the sequence of instantiation on the command line. Options are listed below. By default, a series of colors will be chosen by the program if none are provided upon invocation.

Order of default colors: yellow, red, green, violet, blue, orange, gray, brown, magenta, white, indigo).

titlecoloroption

The color to be used for titles, axis lines, tics, and scale numbers.
Default: "white"
Color options: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white, and black.

xtitlevalue

Title to describe X data. Will be centered beneath the graph. Default is no title unless the need for a unit descriptor is computed by the program (i.e. X: title in hundreds). Also, see NOTES section (below) for a format caveat for multi−word titles.

ytitlevalue

Title to describe Y data. Will be centered beneath the X data title. Default is no title unless the need for a unit descriptor is computed by the program (i.e. Y: ttiittllee in thousands). Also, see NOTES section (below) for a format caveat for multi−word titles. In the case of graphs with multiple lines, one may wish to use more specific title placement by using the *d.text* or *d.label* programs.

titlevalue

Title to describe the graph. Will be centered over the top of graph. Default is no title. See NOTES section (below) for a format caveat for multi−word titles.

NOTES

Since the command line parser is not amiable to multiple word inputs, to input titles of more than one word, use the underscore character ("") to represent spaces (" ").

Example: "titleCensusdata1990" would be printed over the graph as "Census data 1990".

The way the program locates and labels tic marks is less than perfect: 1) although distances between Y tics is proportional to the value, they are not proportional on the X axis; 2) decimal values between −1 and 1 can be printed on the X axis, but not on Y. (With respect to the later, the input for Y values can all be multiplied by a factor of 10 before graphing).

BUGS

Unknown

SEE ALSO

d.frame, *d.text*, *d.label*, *parser*

AUTHOR

Chris Rewerts, Agricultural Engineering, Purdue University

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.mapgraph − Generates and displays simple graphics on map layers drawn in the active graphics monitor display frame.
(GRASS Display Program)

SYNOPSIS

d.mapgraph
d.mapgraph help
d.mapgraph [**input=***name*] [**color=***name*]

DESCRIPTION

d.mapgraph draws graphics that are described in standard input (default) or the UNIX input file *name*. If commands are entered via standard input, a *ctrl−d* is used to signal the end of input to *d.mapgraph*. This program performs essentially the same function as *d.graph*; however, point locations are specified to *d.mapgraph* in the geographic coordinate system of the user's current mapset and location (i.e., in map coordinates), rather than in graphics display screen coordinates.

The program can be run interactively or non−interactively. The user can run the program completely non−interactively by specifying the name of a file containing *d.mapgraph* graphics commands and the values of all needed parameters on the command line. The user can instead elect to run the program partially interactively, by specifying any/all of the parameters *except* the graphics **input=***name* parameter on the command line. In this case, *d.mapgraph* will expect the user to input *d.mapgraph* graphics commands from standard input (i.e., the keyboard) and will (silently) prompt the user for these graphics commands. Alternately, the user can simply type **d.mapgraph** on the command line, and be prompted for the values of all parameters (the user can still input graphics commands from an input file using this form.) In this case, the user is presented with the standard GRASS *parser* interface.

OPTIONS

Parameters:

input=*name*
  Name of a UNIX file containing graphics instructions. Specify the full path name of the file if not in the current directory. If no file name is given, commands are taken from standard input.

color=*name*
  Starting color desired for graphics.
  Options: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white, and black.
  Default: *white*

d.mapgraph is used for drawing simple graphics on top of map layers. The coordinate system used by *d.mapgraph* is the same as that of the map layer displayed in the active display frame on the graphics monitor (or that of the user's current region, if no map is displayed).

The graphics language is simple and uses the following commands:

# comment

A line of comment which is ignored in the processing.

move *xpos ypos*

The current location is updated to *xpos ypos* (where these, respectively, are the easting and northing of geographic coordinates stated in the map coordinate system of the user's current GRASS location, falling within the current region and active frame). If unspecified by the user, the current location becomes (0,0). If, as most likely, the point (0,0) falls outside of the user's current region, graphics drawn there will not appear in the graphics frame.
Note: use *g.region* to obtain the coordinates of current location. Use *d.where* to obtain specific map coordinates of various points on the raster map displayed in the active frame.
Note: there must be a space between *xpos* and *ypos*.

draw *xpos ypos*

A line is drawn in the current color from the current location to the new location *xpos ypos*, which then becomes the current location. *xpos* and *ypos* are (respectively) an easting and northing stated in the map coordinate system of the user's current GRASS location, and located within the user's current geographic region and active frame.
Note: there must be a space between *xpos* and *ypos.*

color *color*

Sets the current color to that stated. Color options are: *red*, *orange*, *yellow*, *green*, *blue*, *indigo*, *violet*, *magenta*, *brown*, *gray*, *white*, and *black*.

size *xper yper*

Subsequent text will be drawn such that the text is *xper* percent of the display frame's width and *yper* percent of the display frame height. If not specified by the user, the text size becomes 5 percent of the active frame's width and 5 percent of the frame's height. This is equivalent to entering **size 5 5**.

text *line−of−text*

The stated text is drawn at the current location using the current color and the current size.

icon *type size x y*

Draws an icon of types o, x, or + with specified size at location x,y. Note: type o designates a square.

polygon *xpos ypos*
xpos ypos
.
.
.

The map coordinates appearing on lines beneath the word *polygon*, one pair per line, circumscribe a polygon which is to be filled with the current color.

NOTES

d.mapgraph is identical to the *d.graph* command, except for the difference in coordinate systems used.

d.mapgraph will complain if the user enters something to standard input that it does not understand. Blank lines in the input file will result in this error message.

SEE ALSO

d.frame
d.graph
d.rast
d.zoom
g.region
grass.logo.sh
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.measure − Measures the lengths and areas of features drawn by the user in the active display frame on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.measure
d.measure help
d.measure [**−skm**] [**c1=***name*] [**c2=***name*]

DESCRIPTION

d.measure provides the user with an interactive way to measure the lengths and areas of lines and polygons drawn by the user in the active frame on the graphics monitor. Lines and polygons are drawn using a pointing device (mouse). Each line segment is drawn in colors *c1* and *c2*. A mouse button menu indicates that the user can find out the geographic coordinates of the cursor location, draw line segments between user−specified vertices, and quit *d.measure*. Lines and polygons are defined by the series of vertices marked by the user. If more than two successive vertices are drawn, *d.measure* prints the area encompassed (*d.measure* will assume the area is closed even if the user has not joined the first and last vertices). Line lengths are stated in the same units as those of the current LOCATION. In a XY−LOCATION, where no units are available, *d.measure* prints a warning and gives results without units. Areas are stated in hectares, square miles, square meters and square kilometers. Lines and polygons drawn using *d.measure* will overlay (not overwrite) whatever display currently appears in the active frame on the graphics monitor. The user can, for example, run *d.rast* or *d.vect* prior to running *d.measure*, and use raster and/or vector maps as a backdrop.

OPTIONS

The user can specify the colors in which line segments will be drawn by setting the values of *c1* and *c2* on the command line. Default line colors (see below) will be used if the user does not specify the values of *c1* and *c2* on the command line.

Flags

−s

Suppesses clear screen and writes messages to stderr instead of stdout. Usefull for work with the TCLTKGRASS gui.

−m

Use metric units (meters, kilometers) only

−k

Output results (additionally) in kilometers and square kilometers

Parameters:

***c1**=name*

The first color in which each line segment is drawn, while being positioned.
Options: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white, and black
Default: *white*

***c2**=name*

The second color in which each line segment is drawn, after its vertices are fixed. Options: Same as *c1*.
Default: *white*

NOTES

d.measure uses all segments drawn by the user when computing area. If the user draws an area within another area, the combined area of both regions will be output.

SEE ALSO

d.display
d.frame
d.graph
d.mapgraph
d.rast
d.vect
d.where

AUTHORS

James Westervelt,
Michael Shapiro,
U.S. Army Construction Engineering Research Laboratory

NAME

d.menu − Creates and displays a menu within the active frame on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.menu
d.menu help
d.menu [**bcolor**=*name] [**tcolor**=name] [**dcolor**=name] [**size**=value]*

DESCRIPTION

d.menu allows the user to create a menu containing a TITLE and options, and to display this menu in the active frame on the graphics monitor. After the menu is displayed in the active frame, the pointing device (mouse) is activated and must be used to select one of the menu options. The number associated with the selected menu option is then printed to standard output and the program exits. This program provides GRASS macro writers with a mouse interface for users.

Parameters can be stated on the command line, from within standard input, or (as illustrated in EXAMPLE) from within a script file.

The user can specify the menu's background, text, and line colors (**bcolor**, **tcolor**, and **dcolor**) and the menu size (**size**) on the command line. If the user sets at least one of these values on the command line, any remaining values that are not specified on the command line will be set (automatically) to their default values (see below). Alternately, the user can simply type **d.menu** on the command line; in this case, the program will prompt the user for the menu colors and text size using the standard GRASS parser interface. The user will then be (silently) prompted to enter the menu's location and contents through standard input (see Menu Information, below).

Parameters:

bcolor=*name*
Sets the color of the menu background (the rectangle on which the text sits).
Options: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white, and black.
Default: black

tcolor=*name*
Sets the color of the menu text.
Options: Same as those for bcolor.
Default: white

dcolor=*name*
Sets the line color used to divide text lines in the menu.
Options: Same as those for bcolor.

Default: white

size=*value*

Sets the menu's text size to the specified value. Sizes are stated as percentages of the height of the active graphics display frame.
Options: 1 − 100
Default: 3 (i.e., 3%)

Menu Information:

After the user has (optionally) specified menu colors and size, the program expects the user to enter information about the menu's location and content. The menu will be placed in the lower left corner of the active display frame by default if the user does not position it elsewhere using the .T or .L commands.

The user specifies the menu contents by typing in a menu TITLE followed by the option choices to appear in the menu when displayed. The user *must* enter a menu TITLE and at least one menu option.

.T *value*

Specifies the menu's distance from the active display frame's top edge (as a percentage of the active frame's height).
Note: Not required

.L *value*

Specifies the menu's distance from the active display frame's left edge (as a percentage of the active frame's width).
Note: Not required

menu TITLE

A TITLE that describes the type of options listed in the menu, and that will appear at the top of the menu when it is displayed.

option name(s)

The options that will appear in the menu when displayed. Each menu option should appear on a separate line. The user may enter as many options as desired, but must enter at least one menu option.

Note: The user should choose a menu size and location that will allow all menu options to be displayed in the active frame.

If the user enters the menu TITLE and option(s) from standard input (i.e., at the keyboard rather than from a file), the user should enter *control−d* to end input and display the menu in the active frame on the graphics monitor. (Note: The *d.menu* program can also be incorporated into UNIX Bourne shell script macros. The below example shows how this might be done.)

EXAMPLE

In the following example, the shell script menu2 calls the shell script color.select which contains d.menu commands to display a menu in the current frame on the graphics monitor. After the user selects an option from the display menu, the selection number is available for use by menu2.

Contents of file *menu2*:

```
#! /bin/csh −f
set option = 0
set colors = (red green blue black white )
@ option = `color.select`

if ($option <= 5) then
     set color = $colors[$option]
     echo $color

endif
exit
```

Contents of file *color.select*:

```
#! /bin/csh -f
d.menu bcolor=red tcolor=green dcolor=yellow size=5 << EOF
.T 25
.L 25
Color Choices
Option 1
Option 2
Option 3
Option 4
Option 5
EOF
```

If the user runs *menu2*, a menu will be displayed on the graphics monitor that has red background, green text, with menu options divided by yellow lines, and a text size of 5% of the active display frame height. The mouse cursor will become active, allowing the user to select (by pointing with the mouse) one of the displayed menu options. Here, these menu options are called *Option 1*, *Option 2*, and *Option 3*, etc. The first line of text (here, the words *Color Choices*) contains the TITLE of the menu; this line is **not** a menu option that can be chosen by the user with the mouse. When the user presses one of the mouse buttons while pointing to the desired menu choice, the number of the option chosen will be available for capture by the shell script *menu2*. *menu2* is a simple example that takes this information and only echoes it to the screen.

NOTES

Although the user can vary text size, all text within the same menu is displayed in a single text size (and font). If the user specifies that items included in the menu's text be displayed in different sizes, all text will be displayed in the size stated last.

SEE ALSO

d.display
d.font
d.frame
d.grid
d.label
d.legend
d.paint.labels
d.text
d.title
show.fonts.sh
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.mon − To establish and control use of a graphics display monitor.
(GRASS Display Program)

SYNOPSIS

d.mon
d.mon help
d.mon [**−lLprs**] [**start=***name*] [**stop=***name*] [**select=***name*] [**unlock=***name*]

DESCRIPTION

The GRASS program *d.mon* allows the user to start, select, list, query the status of, release control of, stop, and unlock control of, available graphics monitors. The user can run this program either interactively (through a series of menus), or non−interactively by typing the name of the monitor to start, stop, select, or unlock, and any desired flags on the command line.

Flags:

−l
    List all monitors
−L
    List all monitors (with current status)
−p
    Print name of currently selected monitor
−r
    Release currently selected monitor
−s
    Do not automatically select when starting

Parameters:

start=*name*
    Name of graphics monitor to start
stop=*name*
    Name of graphics monitor to stop
select=*name*
    Name of graphics monitor to select
unlock=*name*
    Name of graphics monitor to unlock

When the user enters the command **d.mon** without specifying parameter values, the below menu appears on the screen:

```
                        MONITOR MENU
Making sure that the graphics monitor is running:

  1      − Start a graphics monitor
         (also automatically selects this monitor)
  2      − Stop a graphics monitor

Choosing a graphics monitor for your graphics:

  3      − Select a graphics device for output
         (currently selected monitor:  name)
  4      − Release control of the graphics driver
         (let someone else use it)
         (option 4 appears only after selection of a monitor)

RETURN   quit
```

These parameters perform the following functions:

1. **Start a Monitor.**
   In order to display on−screen GRASS graphics, the user must *start* and *select* a graphics monitor. By default, the *start* command actually runs two commands, to both start and select whatever monitor is named by the user. (The user can get a list of available monitors by setting the −l or −L flag on the command line.)

   When a monitor is *started*, it is therefore also (automatically) *selected* for output, unless the *−s* flag is set by the user; the user can also explictly *select* a monitor that has been started (see (3) below). After a monitor is started, a blank graphics frame should appear on whatever terminal the user is using to display graphics.

   The desired monitor should be started once and need not be restarted unless it is stopped (option *2)* for some reason. A monitor may continue to run for any length of time, even when no GRASS session is being run. The monitor program runs in the background.

2. **Stop a Monitor.**
   Sometimes the monitor program needs to be stopped (terminated). Choosing option 2 will terminate a user−specified monitor program. A graphics monitor has two different types of status: monitor program *not running*, and monitor *running*. A monitor that has been started and/or selected will be listed as running; a monitor that has been stopped (or not started) will be listed as not running. The −L (list status) flag will list the status of each monitor connected to the system.
3. **Select a Monitor.**
   When the user *starts* a monitor, it is also (automatically) *selected* for graphics output unless the user sets the *−s* flag. In order to use (direct graphics output to) a monitor, the user must *select* that monitor for use, either by simply starting the monitor without the −s flag or by explictly selecting the monitor for output using option 3. Only running monitors can be selected for graphics output. Once the user has *selected* a monitor for output, no other user can use this monitor for graphics output until the monitor driver is either *released* (by the user) or *unlocked* (by any user on the system).

   The user can run multiple graphics monitors by simply starting each of the graphics monitors drivers he wishes to direct output to.

4. **Release a Monitor.**
   Once a user has *selected* a monitor for graphics output, it is locked for use by that user until either: (1) the user voluntarily releases control of the monitor for use by another (option 4), or (2) another GRASS user *unlocks* the user's control of the monitor. Menu option 4 appears only to the person who has selected the monitor (since only that user can release control of his selected graphics monitor.) If another user wishes to *unlock* the user's control of the monitor, that user must run *d.mon* from the command line and set the **unlock=***name* parameter.

You may choose multiple options within the *d.mon* program. To leave (exit) the *d.mon* menu, press the <RETURN> key.

NOTES

The *d.mon* program can regulate control of graphics monitors both in systems using multiple monitors and in systems using a single graphics monitor.

SEE ALSO

d.erase
d.frame
monitorcap

AUTHOR

Michael Shapiro, U.S. Army Construction Engineering Research Laboratory,
Full 24bit support by Roberto Flor, ITC−Irst, Trento, Italy

Last changed: $Date: 2002/04/09 00:31:26 $

NAME

d.paint.labels − Displays text labels formatted for use with GRASS paint (*p.labels*, *p.map*) output to the active frame on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.paint.labels
d.paint.labels help
d.paint.labels file=*name*

DESCRIPTION

d.paint.labels displays the *paint* label file *name* in the active display frame on the graphics monitor. This file can be generated by the *labels* option in the *p.labels* program, the *d.labels* program, or simply created by the user as an ASCII file (using a text editor like *vi*) and placed in the appropriate */paint/labels* directory under the user's current mapset and location (i.e., under $LOCATION/paint/labels).

OPTIONS

Parameter:

file=*name*
: Name of ASCII file containing paint labels.

This program can be run either non−interactively or interactively. If the user specifies the name of a paint labels file on the command line, the program will run without interacting with the user. Alternately, the user can type simply **d.paint.labels** on the command line; in this case, the program will prompt the user for the name of a paint labels file using the standard GRASS *parser* interface.

NOTES

Some of the information stored in the label file is unused by *d.paint.labels*. The unused information found in a label file is printed to standard output. This extra information is used by such programs as *p.labels* and *p.map*.

SEE ALSO

d.display
d.font
d.label

d.labels
d.text
d.title
p.labels
p.map
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.pan − Allows the user to change the current geographic region settings interactively, with a mouse.
(GRASS Display Program)

SYNOPSIS

d.pan
d.pan help
d.pan [**−j**] [**rast=***name*[,*name*,...]] [**vector=***name*[,*name*,...]] [**site=***name*[,*name*,...]] [**zoom=***value*]

DESCRIPTION

d.pan allows the user to interactively adjust the settings (zoom and pan) of the current geographic region using a pointing device such as a mouse. Like *g.region*, *d.pan* re−defines the settings of the geographic region. However, *d.pan* allows the user to change the current region settings interactively, by re−centering/zooming the current map(s). The graphics display is updated automatically.

OPTIONS

Flag:

−j
Just redraw given maps using default colors.

Parameter:

rast=*name[,name,...]*
Name of raster map(s)
vector=*name[,name,...]*
Name of vector map(s)
site=*name[,name,...]*
Name of site map(s)
zoom=*value*
magnification: >1.0 zooms in, <1.0 zooms out
options: 0.001−1000.0
default: 1.0

SEE ALSO

d.display

d.erase
d.rast
d.vect
g.region

AUTHOR

Unkown.
Improvements to auto−redraw maps displayed in GRASS monitor by Huidae Cho and Markus Neteler

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.param.scale − Calculates surface parameters and features from a DEM and displays them in GRASS monitor. It allows interactive indentification of terrain parameters and features with a mouse. Reports membership classes for variable window sizes and their variance or entropy.
(GRASS Raster Program)

SYNOPSIS

d.param.scale
d.param.scale help
d.param.scale [**−c**] [**−f**] [**−t**] **in=***name* **out=***name* [s_tol=value] [c_tol=value] [size=value] [param=name] [exp=value] [zscale=value]

DESCRIPTION

The GRASS program *d.param.scale* can be used to identify interactively terrain parameters and features with a mouse. Reports membership classes for variable window sizes and their variance or entropy.

PROGRAM OPTIONS

d.param.scale can be run either non−interactively or interactively. In non−interactive use, the user must specify the name of the raster map layer to be used for input on the command line, using the form **in=***name name* (where the *name* is the name of a raster map layer to be input.

Flags:

−c
    If set, *d.param.scale* constrains model through central window cell

−f
    If set, *d.param.scale* fixes the axis scale

−t
    If set, *d.param.scale* creates text output of results

Parameters:

in*=name: Raster surface layer to interrogate*

s_tol* =value: Slope tolerance value that defines a `flat' surface (degrees) − default: 1.0*

c_tol=*value: Curvature tolerance value that defines a `planar' surface − default: 1.0*

size =*value: Size of processing window (odd number only) − default: 15*

param=*value: Morphometric parameter to calculate options: elev, slope, aspect, profc, planc, longc, crosc, minic, maxic, feature, resid, moran, geary − default: elev*

exp =*value: Exponent for distance weighting (0.0−4.0) 1 implies a linear decay, 2 an inverse distance squared decay − default: 0.0 (no distance decay).*

zscale =*value: Vertical scaling factor − default: 1.0*

With:

```
elev:     Generalised elevation value.
slope:    Maximum gradient at a point.
aspect:   Direction of maximum gradient.
profc:    Profile convexity (vertical in direction of steepest slope).
planc:    Plan convexity (contour curvature).
crosc:    Cross sectional convexity (tangent to contours, downslope).
longc:    Longitudinal convexity (perpendicular to contours downslope).
minic:    Minimum convexity.
maxic:    Maximum convexity.
feature:  Morphometric features:
            peaks, ridges, passes, channels, pits and planes.
resid:
moran: Moran statistics
geary:
```

SEE ALSO

r.param.scale
parser

AUTHOR

Jo Wood, PROJECT ASSIST,
Department of Geography, University of Leicester

REFERENCE

Wood, J. (1996): The Geomorphological characterisation of Digital Elevation Models. Diss., Department of Geography, University of Leicester, U.K.
online at:
www.geog.le.ac.uk/jwo/research/dem_char/thesis/index.html

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.points − Displays point graphics in the active frame on the graphics display monitor.
(GRASS Display Program)

SYNOPSIS

d.points
d.points help
d.points [**color**=*name*] [**size**=*value*] [**type**=*name*] [**file**=*name*]

DESCRIPTION

d.points displays points whose coordinate locations are specified by the user and located within the current geographic region in the active display frame on the graphics monitor. Point coordinates are read either from standard input or from a file stated by the **file=***name* option. When entering coordinates through standard input, the user presses *control−d* to signal the end of input to *d.points*.

OPTIONS

The program can be run interactively or non−interactively. The user can run the program completely non−interactively by specifying the name of a points **file** (containing the geographic coordinates of icons to be sited) and the values of all needed parameters on the command line. If the name of an input **file** to *d.points* is given on the command line, any other parameter values left unspecified on the command line (e.g., the icon **color**, **size**, and/or **type**) are set to their default values.

The user can instead elect to run the program partially interactively, by specifying any/all of the parameters *except* the points **file=***name* parameter on the command line. In this case, *d.points* will expect the user to enter icon locations from standard input (i.e., at the keyboard) and will prompt the user for these locations. Each icon location should be entered as an easting and northing pair separated by a space, followed by a carriage return. Coordinates should be stated in whatever map coordinate system is used by the user's current GRASS location. The user presses *control−d* to signal the end of standard input to *d.points*.

Alternately, the user can simply type **d.points** on the command line, and be prompted for the values of all parameters (the user can still input icon coordinates from an input file using this form). In this case, the user is presented with the standard GRASS *parser* interface.

Parameters:

color=*name*

Sets the current color to the *name* stated.
Options: red, orange, yellow, green, blue, indigo, violet, gray, white, and black

Default: *gray*

size=*value*

Size of icon in pixels.
Options: 0−1000 (pixels)
Default: 5 (pixels)

type=*name*

Sets the shape (type) of the icon to be drawn to *name*.
Options: x, diamond, box, +
Default: +

file=*name*

Take input from the UNIX file *name*. (Note: This file must be in the user's current directory or given by its full path name.)
Default: Standard input.
The file contents should consist of a series of geographic coordinates that fit within the boundaries of the current geographic region falling within the active display frame (refer to *d.where* or *g.region* for these coordinates). Each icon location should be specified by an easting and northing (in that order) separated by a single blank space, and followed by a carriage return. Eastings and northings should be stated in whatever map coordinate system is being used by the user's current GRASS location.

NOTES

d.points will display a point at the location specified in the active frame, even if the point lies outside the boundaries of current geographic region.

SEE ALSO

d.frame
d.where
d.sites
g.region
s.out.ascii
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.profile − Displays profiles of a user−specified raster map layer.
(GRASS Display Program)

SYNOPSIS

d.profile
d.profile rast=*name* [**drast=***name*] [**plotfile=***name*]

DESCRIPTION

This command works only interactively. It clears the entire graphics screen and provides a graphical interaction allowing the selection of transects for which profiles are then generated. Input parameters can be specified on the command line for quicker start−up.

USER PROMPTS

First, you will be presented with a prompt asking you to choose a raster map layer to be profiled. Then you will be prompted for an optional display raster. The optional display raster will be shown rather than the profiled raster. This is useful to be able to more easily see ground features, such as might be visible in an aerial photo or satellite image. Finally, you will be prompted for the name prefix for output file(s) containing the profile data. This is an optional feature.

After parameters are given, the raster layer will be displayed in the left half of the graphics monitor, and the right half will be divided into four frames. There will also be two frames along the top of the monitor: a mouse−button menu frame on the left, and a status frame on the right.

The mouse−button menu initially offers you three options:

> Mouse Buttons:
> Left: Where am I?
> Middle: Mark FIRST Point of Profile Line.
> Right: QUIT this.

You may query the displayed raster map layer by indicating points with the left mouse−button. The coordinates and category value of each point that you indicate will be displayed on in the status frame. If you mark the first point of the profile line you will be presented with the following mouse−button menu:

> Mouse Buttons:
> Left: Where am I?
> Middle: Mark SECOND Point of Profile Line.

Right: QUIT this.

Once you mark the second point of the profile line, the profile line will be labeled (with a letter from A to D) and displayed in one of the four display frames on the right hand side of the screen. You will then be presented with a third mouse−button menu:

Mouse Buttons:
Left: DO ANOTHER
Middle: CLEAR DISPLAY
Right: QUIT this.

If you would like to view another profile, click on the left mouse−button. If you would like to redisplay the raster map layer and clear out the four profile frames, click on the middle mouse−button. If you would like to quit, then click on the right button.

If you selected the **plotfile** option, you will have up to four files starting with the *name* and followed by a suffix of the plot letter *A*, *B*, *C*, or *D*. The plots that are written to these files reflect the last of each plot done with that letter (i.e. They will match what's in the display when you quit).

The format of the plot files is not geared toward any particular software, but should be easy to coax into a number of other programs. The first three lines contain some header information about the plot, each prefixed with a # sign to denote a comment. A fourth comment line describes the data that follows. It consists of the distance (always in meters), the cell value (in whatever units it is in), the easting in decimal format, and the northing in decimal format. The number of values will vary depending on the length of the profile, cell resolution and the slope angle of the profile line. Below is an example of a plotfile:

```
# Profile A of elev.ft@snows
# From (702879.29679757, 4287317.55920185) to (702722.40973589,
4287061.72632285)
# Stats: Count = 644, Min = 2424.658936, Max = 2513.246094
# dist value east north
0.000000 2513.246094 702879.102364 4287317.516912
0.000000 2513.246094 702878.713496 4287317.118970
0.556395 2513.246094 702878.324629 4287316.721029
1.112790 2513.246094 702878.324629 4287316.323088
...
```

NOTES

You might notice the first two 'dist' values in the profile output above are both zero. This is due to the fact the cell resolution for this file is less than one meter, and so the function that calculates the distance considers the distance between the first cell and the second to be zero. You might also notice, the coordinates given in the header and displayed on screen are slightly different from the first and last coordinates given in the profile data output. This is because the profile data output finds the eastings and northings for the center of the cells while the coordinate transformations from mouse clicks might yield slightly different coordinates which still fall within the same cell boundaries. The difference should always be less than the distance between the center and any corner (not edge!) of the cell at the resolution of the profiled raster.

Only four output plotfiles can be made. Each time a new profile is run, the plotfile is immediately written to the file *name.letter*. One can take advantage of this fact to create an unlimited number of plotfiles simply by renaming the output files before running more profiles. This may not always be the case, but it is at the time of this writing.

Useful enhancements to *d.profile* would include:

1. Adding an option to display profiles using category colors, like a bar−chart.
2. Allowing profile lines to be defined by a series of points, not just two.
3. Allowing profiles to be saved in a file, for later viewing by GRASS.
4. Allowing the user to enter profile line points by typing coordinates.

Perl Script to convert output to a site_list

The script below will convert the **plotfile** output to the site_list format. It probably won't work 100% for lat/lon datasets.

```
#! /usr/bin/perl −w
# prof2sites.pl: converts the output of d.profile to a site_list

my ($dist, $elev, $east, $north);

while (<STDIN>)
{
    if (m/^#/)
  {
        print "$_";
  }
    else
  {
        chomp;
   ($dist, $elev, $east, $north) = split / /;
        print "$east|$north|$elev|#1 \%$dist \@Profile\n";
  }
}
```

BUGS

None known.

AUTHOR

Dave Johnson
DBA Systems, Inc.
10560 Arrowhead Drive
Fairfax, Virginia 22030

Modified by Eric G. Miller for profile output, support of floating point rasters, and optional display raster (02−Dec−2000).

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.rast.arrow − Draws arrows representing cell aspect direction for a raster map layer.
(GRASS Display Program)

SYNOPSIS

d.rast.arrow
d.rast.arrow help
d.rast.arrow [**map**=*name*] [**type**=*name*] [**arrow_color**=*name*] [**grid_color**=*name*] [**x_color**=*name*] [**unknown_color**=*name*]

DESCRIPTION

d.rast.arrow is designed to help users better visualize surface water flow direction, as indicated in an aspect raster map layer. There are two ways to specify the aspect layer the program is to use. The first is to display the aspect map layer on the graphics monitor before running *d.rast.arrow*. The second method involves setting the *map* parameter to the name of the desired aspect map. This allows the arrows to be drawn over any other maps already displayed on the graphics monitor.

d.rast.arrow will draw an arrow over each displayed cell to indicate in which direction the cell slopes. An arrow can point in one of eight directions. If the aspect layer has a category value denoting locations of "unknown" aspect, *d.rast.arrow* draws a question mark over the displayed cells of that category. Cells of category 0 (no data) will be marked with an "X".

Note: Arrows are cell size and cannot be seen if the raster map is relatively close in scale. From a command line mode, the user cannot run *d.rast.arrow* and then run *d.zoom* and still see the arrows. The zooming action erases the arrows and the user must run *d.rast.arrow* again.

OPTIONS

Parameters:

map=*name*
Name of an existing raster map layer to be displayed.
type=*name*
Type of existing raster data to be displayed.
Options: grass, agnps, answers
Default: grass
Using this information, *d.rast.arrow* uses internal information to convert category values into appropriate arrow directions.
arrow_color=*name*

Color in which arrows will be drawn.
Options: white, red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, black
Default: green

grid_color=*name*
Color in which grid outlines will be drawn.
Options: white, red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, black
Default: gray

x_color=*name*
Color in which x's will be drawn.
Options: white, red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, black
Default: white

unknown_color=*name*
Color in which unknown information will be displayed.
Options: white, red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, black
Default: red

SEE ALSO

d.frame
d.rast
d.rast.edit
d.rast.num
d.zoom
g.region
r.mask
r.slope.aspect
parser

AUTHOR

Chris Rewerts, Agricultural Engineering, Purdue University

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.rast.edit − Program allowing users to interactively edit the cell category values of raster map layers displayed on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.rast.edit
d.rast.edit help
d.rast.edit

DESCRIPTION

The *d.rast.edit* program allows users to interactively edit cell category values in a raster map layer displayed to the graphics monitor using a mouse cursor. This program determines the name of the raster map layer currently displayed in the active frame on the selected graphics monitor (if none, program will abort). The user is then prompted for the name of a new raster layer to be created in his mapset. *d.rast.edit* does not modify the user's original raster map layer.

Geographic Region Concerns

d.rast.edit reads the region definition for the raster map layer being edited from its cell header file. The new, edited copy of the raster layer will be created with the same resolution and region dimensions. If the current region resolution does not match the raster map layer's resolution, the program will abort. The north, south, east, and west geographic region settings of the current region can be set to any view, as long as they fall within the boundaries of the raster map layer described in its cell header. This is important, since the current view must be such that individual cell locations are easily visible and sufficiently large on the graphics monitor to be pointed at with the mouse cursor. Any mask in place will be ignored when writing the new raster map layer.

Mouse Menus

Cell editing is done using a mouse cursor to choose menu selections and indicate areas on the displayed raster map that are to be edited.

Selections available via the *Main Menu* and the subsequent functions or sub−menus are described below:

1. Edit
invokes edit mode, during which no graphic menus are used, so that none of the displayed map is hidden. At the start of the edit mode, a grid is drawn over displayed cells to help distinguish cell boundaries. Interaction with the program during edit mode is done by using the three mouse buttons

as follows:

> Left Button
> **what's here**. Identify the category value of the cell under the mouse cursor when the button is pressed. The category value of this cell is printed in the text frame.
>
> Middle Button
> **edit**. The current category value of the cell under the mouse cursor is iterated and a prompt for a new cell value appears on the textual command frame. (Remember to move the mouse to the text frame). Enter a new cell category value which is within the range of current category values for the map layer. Note: the edited cell is displayed in the color of the newly−assigned category value, but is hatched with lines of the grid color to indicate it has been edited, since, if the value of the cell is polled (before leaving edit mode), the old category value will be reported.
>
> Right Button
> **exit edit mode**. Command is returned to the *Main Menu* on the graphics monitor. If cell value changes were made during edit mode, they will be saved upon exit.

2. Redraw
redisplays the raster map layer on the graphics monitor.

3. Zoom
calls the *d.zoom* program, to allow changing the view of the raster layer. **Zoom's** instructions will appear on the text command window. Interaction is conducted using mouse keys.

4. Arrow
This function is available for users editing aspect maps. The *d.rast.arrow* program is called, which draws arrows over the displayed cells to indicate the downslope direction of the cell. After selecting the *arrow* selection from the main menu, sub−menus will appear, allowing input options to be set for the *d.rast.arrow* program. The first sub−menu selects the type of aspect data displayed. (regular "GRASS" format, as produced by *r.slope.aspect* ; a format prepared as input to the "AGNPS" program; and a format prepared for the "ANSWERS" program. The *d.rast.arrow* program can accept the name of a layer not drawn on the display for use as input, and an option is given to input the name of that layer, if desired. Otherwise, the currently displayed map layer will be used. Next, the color options of *d.rast.arrow* may be set via separate menus, or a choice for using default settings may be used.

5. Number
calls the *d.rast.num* program, which will print the cell category values over the displayed cells on the graphic monitor.

6. Exit
quit the *d.rast.edit* program. If edits have been made, the new raster map layer will be created. Support files are constructed. Category labels and color maps (if any) are copied from the original layer. *d.rast.edit* redisplays the original raster map file on the monitor as it exits.

NOTES

d.rast.edit will not create a new raster map layer if the user makes no cell edits while running the program.

Be careful not to cover the graphics monitor window with another frame during the editing process. In some cases, the *redraw* option will be able to remove traces of other frames; otherwise, the graphics monitor will not be refreshed until after *d.rast.edit* exits. Further note, however, that this has only been tested on *SUN* workstations, using a modified *SUNVIEW* graphics monitor driver.

The primary bane of the *d.rast.edit* program involves large map layers (with lots of rows and columns) and/or slow computers, since the program must read and write raster map layers row by row for the full size of the map layer as dictated by its region size and resolution. (The current region settings of north, south, east, and west will not limit the size of the edited copy of the map layer, since by use of the *d.zoom* program, these values may change several times during the editing session). Their effects could be lessened if the program were to create a table of changes that it could incorporate into the new raster file on a forked process started when the user exits, or otherwise allow the user to issue a "save" command when he has made all desired edits to the raster file. Currently, for instance, if the user needs to use *d.zoom* to access a different area of the map, he must wait for a read and write for each entrance and egress of the edit mode.

There is no "undo" command or way to exit edit mode without saving changes.

It would be nice to incorporate a scrollable version of *d.legend* (such that one could see a legend for files with many categories on a standard size sub−frame). It would be even nicer to be able to select the category values from a graphical legend when editing cell values (thereby saving a trip to the text frame to type in the new value).

Perhaps method(s) for multiple or mass cell edits would be useful. This could be done by providing modes in which the user may: 1) edit a block of cells to a given value by drawing a box; 2) be able to choose a given value which is automatically used as the new value on each cell chosen until a different value is desired.

There is no interrupt handling. This could leave files in .tmp or (rarely) result in half−baked raster files. The original file would survive unscathed by an interrupt at most any point in execution, but the graphics monitor may be left in an indeterminate state (try *d.erase* or *d.mon* *select=monitor_name* to bring it back into shape). Beware of exiting the program by means other than using *exit* on the *Main Menu*.

The program has no method to enter new values beyond the current range of categories, but additional programming could make it so.

SEE ALSO

d.erase *d.frame* *d.mon* *d.rast*
d.rast.arrow *d.rast.num* *d.zoom*
g.region *r.mask* *r.slope.aspect*

AUTHOR

Chris Rewerts, Agricultural Engineering, Purdue University

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.rast − Displays and overlays raster map layers in the active display frame on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.rast
d.rast help
d.rast [**−oi**] **map**=*name* [**catlist**=*list*] [**vallist**=*list*] [**bg**=*color*]

DESCRIPTION

d.rast displays the raster map layer *name* in the active display frame on the graphics monitor.

The program can be run either non−interactively or interactively. If the user specifies the name of a raster map layer (**map**=*name*) and (optionally) the **−o** option on the command line, the program will run non−interactively. Alternately, the user can type simply **d.rast** on the command line; in this case, the program will prompt the user for the flag setting and parameter value using the standard GRASS parser interface.

Flags:

−o

Overlay the named raster map layer onto whatever is already displayed in the active graphics frame. Any no data areas in the named raster map will seem transparent, and reveal the underlying image previously displayed in the graphics frame. If the **−o** flag is set, only cells containing non−null values will be displayed from the *overlaid* raster map. All other areas (i.e., the portions of the overlaid map that are null) will leave the underlying display untouched. If the **−o** flag is not set by the user, *d.rast* will (by default) completely overwrite whatever appears in the active graphics display frame.

−i

Invert catlist.

Parameters:

map=*name*

Name of the raster map to be displayed. If the active graphics frame already contains text or graphics, and the user does not wish to use the **−o** option, it is wise to first invoke *d.erase* to clear the active graphics frame before running *d.rast.* After running *d.rast* other *d.* programs like *d.vect* and *d.grid* can be used to enhance the plot.

catlist=*list*

List of categories to be displayed (INT maps).

vallist=*list*

List of values to be displayed (FP maps).

bg=*color*

Background color for null.

NOTES

The *d.rast* raster map overlay option (**−o**) only works when the color look−up table for the graphics monitor is set to **d.colormode** **fixed**.

SEE ALSO

d.colormode
d.colors
d.colortable
d.display
d.erase
d.grid
d.rast.labels
d.vect
d.what.rast
r.reclass
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/04/12 01:44:09 $

NAME

d.rast.labels − Labels raster cells using the attribute value or the category value.
(GRASS Display Program)

SYNOPSIS

d.rast.labels
d.rast.labels [−smv] map=*name* [**attr=**string|cat|coords] [**size=***number*] [**color=***colorname*] [**backgr=***colorname*] [**border=***colorname*] [**font=***fontname*]

DESCRIPTION

d.rast.labels performs automated placement of labels for a GRASS *raster map* on the current graphics monitor. Labels can be chosen from the category value (numerical), a string attribute, or the labels positions coordinates. The default uses the first string attribute. The user may select the color and sizing of the labels. Labels are only drawn within the current region.

This program can be run non−interactively or interactively. If the user gives the name of a GRASS *raster map* and (optionally) any other parameters, the command will be run non−interactively. Any parameter left unspecified will be set to its default value (see below). Alternatively, the user can type *d.rast.labels* at the command line and be prompted for the parameter values using the standard GRASS *parser* interface.

Parameters:

−s
: save support sites file

−m
: mark labels on the map

−v
: Verbose output

Parameters:

map=*name*
: Name of a GRASS *raster map* map in the user's current mapset search path.

attr=*string|cat|coords*
: Type of attribute to use for the label.
: Default: *string attr* option is set to *string, cat* or *coords*.
: Default: *string*

size=*number*
: The size of letters in pixels.

Default: *10*

color=*colorname*

The GRASS *colorname* to use for the label text.
Options: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white or black
Default: *white*

backgr=*colorname*

Color to use for text box background.
Options: grey, red, white, magenta, brown, blue, indigo, yellow, black, orange, green, violet, none
Default: none

border=*colorname*

Color to use for text box border.
Options: grey, red, white, magenta, brown, blue, indigo, yellow, black, orange, green, violet, none
Default: none

font=*fontname*

The GRASS *fontname* to use for the label.
Options: cyrilc, gothgbt, gothgrt, gothitt, greekc, greekcs, greekp, greeks, italicc, italiccs, italict, romanc, romancs, romand, romans, romant, scriptc or scripts
Default: *romans*

SEE ALSO

d.sites
d.site.labels
d.vect.labels
d.rast
g.region
parser

AUTHOR

Markus Neteler, based on d.vect.labels and r.poly

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.rast.leg.sh − Displays a raster map and its legend on a graphics window.
(GRASS Shell Script)

SYNOPSIS

d.rast.leg.sh
d.rast.leg.sh help
d.rast.leg.sh *rast_map* [*num_of_lines*]

DESCRIPTION

d.rast.leg.sh is a UNIX Bourne shell macro which clears the entire screen, divides it into a main (left) and a minor (right) frames, and then display a raster map in the main frame and the map legend in the minor frame. The main frame remains active when the program finishs.

OPTIONS

The user can run the program interactively or non−interactively.

Parameters:

rast_map
A raster map to be displayed.

num_of_lines
Number of lines to appear in the legend. If this number is not given, the legend frame will display as many lines as number of categories in the map, otherwise, it will display the first **num_of_lines** minus 1 categories with the rest being truncated.

NOTES

The user may adjust the **num_of_lines** parameter or the size of graphics window to get an appropriate result.

FILES

See the file *d.rast.leg.sh* under **`$GISBASE/scripts`**.

SEE ALSO

d.legend
d.rast

AUTHORS

Jianping Xu,
Scott Madry,
Rutgers University

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.rast.num − Overlays cell category values on a raster map layer displayed to the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.rast.num
d.rast.num help
d.rast.num [**map**=*name*] [**grid_color**=*name*]

DESCRIPTION

d.rast.num overlays cell category values onto a raster map layer displayed to the user's graphics monitor. Category values will be displayed in white and/or black, based on the colors in which underlying cells are displayed. A grid outlining each map cell will also be overlain in a user−specified color.

The user should run *d.rast* to display the desired map layer on the graphics monitor before running this program.

d.rast.num can be run non−interactively or interactively. If the user specifies the map whose category values are to be displayed and/or the grid color on the command line, the program will run non−interactively. Alternately, the user may simply type **d.rast.num** on the command line; in this case, the program will prompt the user for parameter values using the standard GRASS user interface described in the manual entry for *parser* . If no grid color is given the default will be used. If no map layer is specified, the program will use whatever raster map layer is currently displayed in the active frame on the graphics monitor.

OPTIONS

Parameters:

map=*name*
Name of existing raster map layer whose category values will be displayed.
Default: (whatever raster map layer is currently displayed)

grid_color=*name*
Color in which an overlain grid will be displayed.
Options: white, red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, black .
Default: gray

NOTES

The user is advised to set the current region to a relatively small area (i.e., less than 100 rows by 100 columns); otherwise, the individual cells being displayed will be small and the category value associated with each will be difficult to see.

SEE ALSO

d.erase *d.frame* *d.grid* *d.mon*
d.rast *d.rast.arrow* *d.rast.edit* *d.rast.zoom*
d.zoom *g.region* *r.mask* *r.slope.aspect*

AUTHORS

Raghavan Srinivasan, and Chris Rewerts, Agricultural Engineering, Purdue University

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.rast.pg − Generate or display a reclass map based on the unique values in a database column.
(GRASS−RDBMS Interface Display Program)

SYNOPSIS

d.rast.pg
d.rast.pg help
d.rast.pg key=name **tab**=name **col**=name [**lab**=name] [**where**=name] **input**=name [**output**=name]

ALTERNATE

d.rast.pg −s help
d.rast.pg −s sql=*filename* **input**=*name* [**output**=*name*]

DESCRIPTION

d.rast.pg displays a raster image in which each category represents a unique value in a database column. Values are retrieved from the current database based on a user defined SQL SELECT statement. The current database is identified by the GRASS environment variable $PG_DBASE which is set using the *g.select.pg* GRASS−RDBMS interface tool. The output from this program is a set of GRASS reclass rules which are used to create or display a reclassed raster surface based on the attributes returned from the database and the raster data layer specified as the input map.

COMMAND LINE OPTIONS

Parameters:

key=databasecolumnname
Column corresponding to category values in raster map [input]
tab=databasetablename
Table containing [col]
col=databasecolumnname
Column to base reclass on

where=clause
Where clause for query (ie. where col='paved')

lab=name
Column to use as labels (optional)

input=map
Name of existing raster file
output=map
Name of new reclass map

ALTERNATE COMMAND LINE USAGE

The alternate command line structure is selected using the the *[−s]* flag on the command line. When using this flag the user must include the name of a text file on the command line as well. This file must include a complete, well formed SQL SELECT statement.

Flag:

−s
SQL statement is input from a prepared file

Parameters:

sql=filename
SQL statement specifying well formed selection criteria.
input=map
Name of an existing raster map
output=map
Name of a new reclass file

EXAMPLE 1:

d.rast.pg input=kur_rast_id output=soils2 tab=info_kuruma key=rec_id col=soils_id lab=soils_info.name where="soils_info.type_id = soils_id"

RESULT:
This example will create a raster map *soils2* with a category for each unique value taken from the field "soils_id" from table *info_kuruma*, with label of value of the column "name" from the table *soils_info*. The raster map will be a reclassed data layer based on the input raster map *kur_rast_id*.

EXAMPLE 2:

d.rast.pg −s sql=raster.sql input=kur_rast_id output=soils3

raster.sql:

select rec_id, soils_info.type_id from info_kuruma where soils_info.name ~ 'barren' and soils_info.type_id = soils_id

RESULT:
This example will create a raster map *soils3* with a category for each unique value taken from the column "type_id" from the table *soils_info*, with only '*barren*' type. The raster map will be a reclassed data layer based on the input raster map *kur_rast_id*. Note that there must be two fields in the SELECT statement (these for the old and new values).

In this example, the input map *kur_rast_id* was prepared by **v.to.rast** from the vector are map *kuruma_id*, which is approximately 7,000 polygons with unique ids (corresponding to *rec_id* field in the table *info_kuruma*).

BUGS

None known.

NOTE

This program requires the Postgres database software. It uses the GRASS module r.reclass started from inside it.

SEE ALSO

g.column.pg, *g.select.pg*, *g.stats.pg*, *g.table.pg*, *d.site.pg*, *d.vect.pg*, *d.what.r.pg*, *d.what.s.pg*, *d.what.v.pg*, *r.reclass.pg*, *r.rescale.pg*, *v.reclass.pg*

AUTHOR

Original Informix SQL−tools: James A. Farley, Wang Song, and W. Fredrick Limp University of Arkansas, CAST

Postgres modifications: Janne Soimasuo, Faculty of Forestry, University of Joensuu, Finland.

Updated to GRASS 5 by Alex Shevlakov (sixote@yahoo.com)

NAME

d.rgb − Displays three user−specified raster map layers as red, green, and blue overlays in the active graphics frame.
(GRASS Display Program)

SYNOPSIS

d.rgb
d.rgb help
d.rgb [**−o**] **red=***name* **green=***name* **blue=***name*

DESCRIPTION

RGB stands for red, green, and blue. This program visually combines three raster map layers to form a color image. For each layer, the corresponding component from the layer's color table is used (e.g. for the red layer, the red component is used, and so on). In general, the layers should use a grey−scale color table.

OPTIONS

Flags:

−o

Overlay the resulting raster map layer onto whatever is already displayed in the active graphics frame. Any no−data areas in any of the named raster maps will seem transparent, and reveal the underlying image previously displayed in the graphics frame. If the **−o** flag is set, only cells containing non−null values will be displayed from the *overlaid* raster map. All other areas (i.e., the portions of the overlaid map that contain null values) will leave the underlying display untouched.

Parameters:

red=*name*
Name of raster map layer to be used for RED component.
green=*name*
Name of raster map layer to be used for GREEN component.
blue=*name*
Name of raster map layer to be used for BLUE component.

NOTES

This is a new version of **d.rgb**, which sends the data to the graphics monitor in true−color RGB format.

Unlike the previous version, it does not attempt to quantize the combined image into a fixed number of colors. Nor does it have an option to generate a composite layer. The image and raster map layers will not display properly if the graphics device does not have a reasonable sampling of the RGB color−space.

If the color quality of a satellite image color composite appears poor, run *r.colors* on the selected satellite channels.

Example:

```
r.info −r image.1
min=0
max=255
r.colors map=image.1 color=rules << EOF
0 black
255 white
EOF
r.colors map=image.2 rast=image.1
r.colors map=image.3 rast=image.1
```

Note: Future GRASS versions may do this for you automatically.

To write out the color composite to a R/G/B raster maps, use *r.composite* (three maps) or *i.composite* (single composite map).

SEE ALSO

blend.sh
d.colormode
d.colors
d.colortable
d.his
hsv.rgb.sh
r.mapcalc
rgb.hsv.sh
i.composite
r.colors
r.composite

AUTHOR

Glynn Clements

Last changed: $Date: 2003/08/25 07:49:30 $

NAME

d.rhumbline − Displays the rhumbline joining two user−specified points, in the active frame on the user's graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.rhumbline
d.rhumbline help
d.rhumbline coor=*lon1,lat1,lon2,lat2* [**lcolor=***name*]

DESCRIPTION

A rhumbline is a line following a constant angle of the compass (i.e., a line of constant direction). There are 32 points on the compass (points are roughly 11 degrees 15 minutes apart). *d.rhumbline* displays the rhumbline joining any two user−specified points in the active frame on the user's graphics monitor. The named coordinate locations must fall within the boundaries of the user's current geographic region.

The user can run this program either interactively or non−interactively. If the user simply types **d.rhumbline** on the command line without specifying parameter values, the mouse will be activated and the user will be asked to use the mouse to indicate the two endpoints of the rhumbline. The rhumbline is then drawn in the default color (white). The program also outputs the coordinate locations of the two endpoints on the user's terminal and the number associated with the mouse button depressed by the user in a format useful for input to other programs.

Alternately, the user can specify the starting and ending longitude/latitude coordinates of the rhumbline and (optionally) the color in which the rhumbline will be displayed; in this case, the program will run non−interactively.

OPTIONS

Parameters:

coor=*lon1,lat1,lon2,lat2*
Starting and ending coordinates of the rhumblines to be displayed, given as longitudes and latitudes.

lcolor=*name*
Rhumbline color.
Options: red, orange, yellow, green, blue, magenta, indigo, violet, gray, white, black
Default: white

NOTES

This program works only with longitude−latitude data bases.

SEE ALSO

d.geodesic

AUTHOR

Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.save − Creates a list of commands for recreating screen graphics.
(GRASS Display Program)

SYNOPSIS

d.save
d.save help
d.save [**−ca**] [**frame**=*name*[,*name*,...]] [**remove**=*val*[,*val*,...]] [**move**=*from*,*to*[,*from*,*to*,...]]

DESCRIPTION

When GRASS display (*d.*) commands are used to generate graphical images on the graphics monitor, some commands are noted in lists that the current graphics driver (see manual entry for *d.mon*) maintains. This command, *d.save*, uses this information to create a shell script that can be used to recreate graphics in another frame at another time. The shell script is sent to standard output (the user's terminal). If you wish to save the shell script created by *d.save*, its output should be redirected to a file; for example:

> **d.save > script.out**

The user can run the program non−interactively by specifying flag settings and parameter values on the command line. If the user types **d.save** without specifying these program arguments, the user will be prompted for inputs through the standard user interface described in the manual entry for *parser*.

OPTIONS

Flags:

−c

Shell script will contain only information for regenerating graphics from the active (current) frame. Use of this flag overrides the *frame* option (below).

−a

Save information needed to regenerate all the frames on the graphics monitor. The shell script can be used to completely reconstruct the contents of the graphics screen.

Parameter:

frame=*name[,name,...]*

Name(s) of those display frame(s) to be saved.
Options: All available frames
Default: Active frame

remove=*val[,val,...]*
List no's to be removed.

move=*from,to[,from,to,...]*
List no's to be moved, "from" to "to". remove option will be done first, if any.

NOTES

List no's are displayed in the right−most comments.

Display commands used interactively, and those that take their graphic instructions from standard input, will not be saved by *d.save*.

SEE ALSO

d.frame
d.mon
g.region
parser

AUTHOR

David Satnik, Central Washington University

Last changed: $Date: 2003/08/20 08:10:12 $

NAME

d.scale − Overlays a bar scale and north arrow for the current geographic region at a user−defined location in the active display frame.
(GRASS Display Program)

SYNOPSIS

d.scale
d.scale help
d.scale [**−m**] [**bcolor=***name*] [**tcolor=***name*] [**at=***x,y*]

DESCRIPTION

d.scale overlays a bar scale and north arrow for the current geographic region at a user−defined location in the active display frame. The scale and north arrow are proportioned to fit in the active frame when placed along the frame's left edge.

The user can specify the location of the scale and north arrow interactively (using a mouse), or provide the geographic coordinates at which they will be placed. The scale and north arrow will only overwrite those portions of the graphics display that lie directly beneath them.

The user can also specify the colors in which the scale and north arrow will be drawn. By default, if unspecified by the user, a white scale and north arrow will be displayed on a black background.

OPTIONS

The user can run this program either non−interactively or interactively. The program will be run non−interactively if the user specifies program parameter values and desired flag settings on the command line, using the form:

> **d.scale** [**−m**] [**bcolor=***name*] [**tcolor=***name*] [**at=***x,y*]

If at least one parameter value is given on the command line and the −m flag is not set, the program will be run non−interactively; any other parameter values left unspecified on the command line will be set equal to their default values (see below). Alternately, the user can simply type **d.scale** on the command line, without program arguments. In this case, the user will be prompted for needed parameter values and the flag setting using the standard GRASS parser interface. If the user sets the −m flag, *d.scale* will expect the user to designate the scale's location interactively, using the mouse.

Flag:

−m
Use the mouse to interactively place the location of the scale and north arrow.

Parameters:

bcolor=*name*
Set the background color underlying the scale and north arrow to *name*.
Options: *red*, *orange*, *yellow*, *green*, *blue*, *indigo*, *violet*, *gray*, *brown*, *magenta*, *white*, and *black*
Default: *black*

tcolor=*name*
Set the foreground color in which the text, scale, and north arrow are displayed to *name*.
Options: Same as *bcolor* colors.
Default: *white*

at=*x,y*
The map easting (x) and northing (y) geographic coordinates designating the location at which the upper left corner of the scale is to be placed. The user should choose geographic coordinates located within the current geographic region (see *d.where* and *g.region* for these coordinates).
Default: 0.0,0.0

NOTES

d.scale assumes that layer units are in meters.

The scale that *d.scale* generates is probably not suitable for very tiny maps (in small display frames).

SEE ALSO

d.barscale
d.legend
d.measure
d.what.rast
d.what.sites
d.what.vect
d.where
g.region
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.site.edit − Allows the user to interactively query and update/delete the category contents of single site map layer at user− specified locations within the current geographic region. (GRASS Display Program)

SYNOPSIS

d.site.edit
d.site.edit help
d.site.edit [**input**=old_site_name] [**output**=output_site] [**dcolor**=color]

DESCRIPTION

d.site.edit outputs the east, north coordinate values and labels associated with site(s) nearest to the user−specified location(s) on user−named site map layer.

The program will query the contents of site map layer named by the user on the command line and allow to either update the description of a site or remove a site. The input site layer must exist in the user's current mapset search path.

The program activates the mouse, and expects the user to indicate the site location(s) to be queried by depressing a mouse button over desired location(s) within the current geographic region in the active display frame on the graphic monitor.

Parameters:

input=old_site_name Name of existing site map layer.

output=new_site_name Name of a new site map layer.

dcolor=name Sets the display color to name.

> Options: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white, and black
> Default: red

d.site.edit allows to edit or display the contents of the geographic coordinates of a location pointed to, and, for the site map layer, and category label nearest site in the named site map layers at this location.

EXAMPLE

It is helpful, but not necessary, to first display a map to be used for reference in the active display frame before running d.site.edit. For example, the user might type the following series of commands and receive the output below.

```
d.sites map=wells
```

To first display the soils map in the active frame.

```
d.site.edit input=wells output=new_wells
```

User then moves the mouse to desired location on the displayed well map layer, and presses the middle mouse button to query the category contents of the nearest well location. The program then outputs the below information to the user's terminal.

```
−613988.990000 east, 804343.040000 north, 08376300 1968 1980 1 3
```

The query may be repeated as often as desired using the left mouse button. The right button on the mouse is used to quit the d.site.edit session. The left button will allow get into update menu where using left click the category information can be updated, using middle mouse option a site could be deleted and the right mouse option will bring back to main menu.

NOTES

d.site.edit will always print its output to the terminal screen. d.site.edit output can be redirected into a file; however, if it is, the output will go both to the screen and to the file. For example: .nf d.site.edit input=wells output=new_wells > what.out .fi will both send d.site.edit output to the screen and capture its output in the file named what.out.

SEE ALSO

d.sites
d.what.site
d.what.rast
d.what.vect
d.where
g.region
parser

AUTHOR

Raghavan Srinivasan, Blackland Research Center, TAES, Temple, TX 76502

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.site.labels − Interactively label sites using a string or double attribute value or the category value on the current graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.site.labels
d.site.labels −m file=*name* [**attr=**string|cat|double|coords|dim] [**index=***number*] [**xref=**left|center|right] [**yref=**top|center|bottom] [**size=***number*] [**color=***colorname*] [**backgr=***colorname*] [**border=***colorname*] [**font=***fontname*]

DESCRIPTION

d.site.labels performs interactive placement of labels for a GRASS *site_lists* on the current graphics monitor. Labels can be chosen from the category value, a double attribute or a string attribute, the coordinates, or a dimension. The default uses the first string attribute. The *index* is a number greater than zero for the attribute to use as a label. The user may select the position, color and sizing of the labels. Labels are only drawn within the current region.

This program can be run non−interactively or interactively. If the user gives the name of a GRASS *site_lists* and (optionally) any other parameters, the command will be run non−interactively. Any parameter left unspecified will be set to its default value (see below). Alternatively, the user can type *d.site.labels* at the command line and be prompted for the parameter values using the standard GRASS *parser* interface.

Parameters:

−m
Use the mouse to place labels interactively. If the user chooses no, this program behaves similarly to d.sites.label.
Default: *No*

file=*name*
Name of a GRASS *site_lists* map in the user's current mapset search path.

attr=*string|cat|double|coords|dim*
Type of attribute to use for the label.
Default: *string*

index=*number*
Index number to the attribute. For instance, *3* would request values from the third attribute column. This option is ignored if the *attr* option is set to *cat* or *coords*.
Default: *1*

xref=*left|center|right*

Horizontal label placement in relation to the site point.
Default: *center*

yref=*top|center|bottom*
Vertical label placement in relation to the site point.

size=*number*
The size of letters in pixels.
Default: *10*

color=*colorname*
The GRASS *colorname* to use for the label text.
Options: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white or black
Default: *white*

backgr=*colorname*
Color to use for text box background.
Options: grey, red, white, magenta, brown, blue, indigo, yellow, black, orange, green, violet, none
Default: none

border=*colorname*
Color to use for text box border.
Options: grey, red, white, magenta, brown, blue, indigo, yellow, black, orange, green, violet, none
Default: none

font=*fontname*
The GRASS *fontname* to use for the label.
Options: cyrilc, gothgbt, gothgrt, gothitt, greekc, greekcs, greekp, greeks, italicc, italiccs, italict, romanc, romancs, romand, romans, romant, scriptc or scripts
Default: *romans*

SEE ALSO

d.display
d.icons
d.sites
d.sites.label
d.points
d.rast
d.vect
g.region
parser

AUTHOR

Eric G. Miller, *Unknown others*

$Id: d.site.labels.html,v 1.6.2.1 2003/08/20 08:10:12 markus Exp $

Last changed: $Date: 2003/08/20 08:10:12 $

NAME

d.site.pg − Display select sites from a database query. (GRASS−RDBMS Interface Display Program)

SYNOPSIS

d.site.pg
d.site.pg help
d.site.pg [**sql**=file] **tab**=name **coordx**=name **coordy**=name [**cats**=name] [**where**=name] [**map**=name] [**plot**=color,icon,size]

ALTERNATE

d.vect.pg −s help
d.vect.pg −s sql=filename [**map**=name] [**plot=**color,icon,size]

DESCRIPTION

Query sites in a PostgreSQL database using an SQL file or by specifying conditions directly. Optionally, plot display or send output to a site_lists "map".

d.site.pg displays select point location information returned from a database query. The site locations returned from the database query are displayed as icons in the active frame. The user controls the color, size and icon used in the graphic output. As option, a site list of the database output is generated and placed in the current mapset/location.

COMMAND LINE OPTIONS

Parameters:

tab=databasetablename
: Table containing X, Y coordinate values.

coordx=database_x_columnname
: Column containing x coordinate values.

coordy=database_y_columnname
: Column containing y coordinate values.

where=SQLwhereclause
: SQL "where" clause which specifies the query criteria to be used in subsetting the database. The information specified in the where option must indicate the column(s) to be used, the operators to be used in the evaluation and the values which the data in the column will be evaluated against.

For example, if you want to select only those records from the table well where the value for depth is either 58 or 75 the following could be entered:
well.depth = 58 or well.depth = 75

To select all wells in an area where the value for well is between 50 and 120:

well.depth > 50 and well.depth < 120

To select all wells of depth greater than 75 where the value for owner is not equal to SMITH:

well.depth > 75 and well.owner != 'SMITH'

If the database column used as the selection criteria is a character field then the associated value must be placed in quotes. To determine the data types associated with columns in the currently selected database use the *g.column.pg* command with the *−v* flag.

In addition to the operators presented in the examples above a range of relational operators including and, or, etc. are supported.

Queries more complex are best implemented using a prepared SQL file.

map=list

Name of sites list to output

cats=name

Name of the category field to be assigned in the new site list

plot=color,icon,size

Colors: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white, black
Icon: diamond, box, plus, x
Size: 1−9
Default: gray, x, 3

sql=filename

SQL statements specifying well formed selection criteria

ALTERNATE COMMAND LINE USAGE

The alternate command line usage is provided to simplify the process of retrieving information from more than one table in the query criteria. The alternate command line structure is selected using the

the *[−s]* flag on the command line. When using this flag the user must include the name of a text file on the command line as well. This file must include a complete, well formed SQL SELECT statement specifying the query criteria to retrieve values which correspond to category values in a GRASS data layer.

Flag:

−s

SQL select statements are input from a prepared file

Parameters:

map=list

Name of sites list to output

plot=color,icon,size

Colors: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white, black
Icon: diamond, box, plus, x
Size: 1−9
Default: gray, x, 3

sql=filename

SQL statements specifying well formed selection criteria

EXAMPLE:

1. *d.site.pg tab=izb coordx=xcoor coordy=ycoor where="how_big~'small'" plot=white,diamond,9*

Result: displays only 'small' huts in white color diamonds size 9

2. *d.site.pg −s sql=izb3.sql map=huts_good*

izb3.sql is:
select xcoor,ycoor,owner from izb where how_good ~ 'good' and izb_info.name=name and izb_info.rec_id=num

Result: displays only 'good' huts (and only huts which have 'name' coinciding with that taken from a table *izb_info*, linked on the hut number) and creates site map called *huts_good* with categories taken from the database field 'owner'

BUGS

None known.

NOTE

This program requires the Postgres database software.

SEE ALSO

g.column.pg, *g.select.pg*, *g.stats.pg*, *g.table.pg*, *d.rast.pg*, *d.vect.pg*, *d.what.r.pg*, *d.what.s.pg*, *d.what.v.pg*, *r.reclass.pg*, *r.rescale.pg*, *v.reclass.pg*

AUTHOR

Original Informix SQL−tools: James A. Farley, Wang Song, and W. Fredrick Limp University of Arkansas, CAST

Postgres modifications: Janne Soimasuo, Faculty of Forestry, University of Joensuu, Finland.

Updated to GRASS 5 by Alex Shevlakov (sixote@yahoo.com)

NAME

d.siter − Interactively displays subsets of a sites list based on site attributes (*GRASS Display Program*)

SYNOPSIS

d.siter
d.siter help
d.siter *name*

DESCRIPTION

d.siter is a script which calls s.info to determine the number, types and ranges of site attributes, then calls a tcl script which presents a user interface. On the interface there are two sliders for each dimension, category, or decimal attribute in the sites file. If a header with labels is in the file, the sliders will be labeled. The value range for each slider pair is determined by the range for the corresponding attribute found in the sites file for sites within the current region. At startup one slider in each pair is set at the minimum value and the other slider is set at the maximum value. The user adjusts sliders as needed to set ranges of desired values for each attribute, dimension, or category. The interface also has controls for setting the size, color, and type of marker, and buttons for **display**, **erase**, **reset scales**, and **quit**.

- When the **display** button is chosen, the GRASS program d.sites.qual is called with rules derived from the current state of the interface and the specified subset is drawn in the active GRASS monitor. By adjusting the size, color and type of marker and drawing multiple subsets, the user may more easily visualize spatial relationships of various subsets or determine inconsistant attribute values.
- Subsets may be saved by entering a filename in the text entry box of the interface and choosing **save as:**.
- Due to the limited resolution of sliders and considering the wide range and distribution often found in site attribute data, a **rescale** button is provided. Choosing **rescale** results in the currently defined ranges becoming the new min and max and the sliders being reset to their left and right extremes, allowing further refinement of the range.
- To undo the **rescale** operation, the user may select **reset scales** which resets all scales to their startup values (loosing any ranges already specified).

OPTIONS

This program can be run non−interactively or interactively. If the user gives the name of a GRASS site_lists map on the command line the program will be run non−interactively. Alternately, the user can type simply d.siter on the command line; in this case, the program will prompt the user for a site_lists map.

BUGS

Due to pixel rounding, startup ranges for sliders may be slightly off for floating point attributes. The slider min value will be slightly lower than the min value found in the file and the slider max value will be slightly higher.

SEE ALSO

d.mon, d.sites, d.sites.qual, s.info

AUTHOR

Bill Brown, UI GMS Laboratory

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.sites − Displays site markers in the active display frame on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.sites
d.sites help
d.sites sitefile=*name* [**color=***name*] [**size=***value*] [**type=***name*]

DESCRIPTION

d.sites displays a GRASS *site_lists* map using site markers of a color, size, and type specified by the user. Output is displayed in the active frame on the graphics monitor.

This program can be run non−interactively or interactively. If the user gives the name of a GRASS *site_lists* map and (optionally) specifies other parameter values on the command line the program will be run non−interactively; any parameter values left unspecified are set to their default values (see below). Alternately, the user can type simply **d.sites** on the command line; in this case, the program will prompt the user for parameter values using the standard GRASS *parser* interface.

Parameters:

sitefile*=name*
Name of a GRASS *site_lists* map in the user's current mapset search path.

color*=name*
Sets the current color to *name*.
Options: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white, and black
Default: *gray*

size*=value*
Size, in pixels, in which the site icons are to be drawn.
Options: 0−1000
Default: 5

type*=name*
The type of icon to be displayed at site locations.
Options: x, diamond, box, +
Default: x

SEE ALSO

d.display
d.icons
d.points
d.rast
d.vect
g.region
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:33 $

This module is deprecated. See d.site.labels

NAME

d.sites.label − Labels sites using a string or double attribute value or the category value on the current graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.sites.label
d.sites.label sitefile=*name* [**attr=**string|cat|double|coords|dim] [**index=***number*] [**anchor=**c|n|s|e|w|ne|nw|se|sw] [**color=***colorname*] [**font=***fontname*] [**width=***number*] [**height=***number*]

DESCRIPTION

d.sites.label displays labels for a GRASS *site_lists* on the current graphics monitor. Labels can be chosen from the category value, a double attribute or a string attribute, the coordinates, or a dimension. The default uses the first string attribute. The *index* is a number greater than zero for the attribute to use as a label. The user may select the position, color and sizing of the labels. Labels are only drawn within the current region.

This program can be run non−interactively or interactively. If the user gives the name of a GRASS *site_lists* and (optionally) any other parameters, the command will be run non−interactively. Any parameter left unspecified will be set to its default value (see below). Alternatively, the user can type *d.sites.label* at the command line and be prompted for the parameter values using the standard GRASS *parser* interface.

Parameters:

sitefile=*name*
Name of a GRASS *site_lists* map in the user's current mapset search path.

attr=*string|cat|double|coords|dim*
Type of attribute to use for the label.
Default: *string*

index=*number*
Index number to the attribute. For instance, *3* would request values from the third attribute column. This option is ignored if the *attr* option is set to *cat* or *coords*.
Default: *1*

anchor=*c|n|s|e|w|ne|nw|se|sw*
Where to place the label in relation to the site point.
Default: *c* (Center)

color=*colorname*

The GRASS *colorname* to use for the label text.
Options: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white or black
Default: *green*

font=*fontname*
The GRASS *fontname* to use for the label.
Options: cyrilc, gothgbt, gothgrt, gothitt, greekc, greekcs, greekp, greeks, italicc, italiccs, italict, romanc, romancs, romand, romans, romant, scriptc or scripts
Default: *romanp*

width=*number*
The width of letters in pixels.
Default: *10*

height=*number*
The height of letters in pixels.
Default: *10*

SEE ALSO

d.display
d.icons
d.sites
d.site.labels
d.points
d.rast
d.vect
g.region
parser

AUTHOR

Eric G. Miller, *Unknown others*

$Id: d.sites.label.html,v 1.5.2.1 2003/08/20 08:10:13 markus Exp $

Last changed: $Date: 2003/08/20 08:10:13 $

NAME

d.sites.qual − Displays a subset of a sites list based on site attributes (*GRASS Display Program*)

SYNOPSIS

d.sites.qual
d.sites.qual help
d.sites.qual [−n] sitefile=*name* [**color**=*name*] [**size**=*value*] [**type**=*name*] [**rules**=*name*[,*name*,...]] [**output**=*name*]

DESCRIPTION

d.sites.qual displays sites from a sites file only if they match criteria established by **rules**.

Syntax of rule string is:

```
[D|C|A]#.RANGE
```

where D, C, or A specify Dimension, Category, or Attribute and RANGE is one of:

```
#−#, ge#, le#, eq#
```

examples:

```
D1.le−88.125     easting <= −88.125
D3.100−200       dimension 3 in range 100 to 200
C.eq7.2          category equal 7.2 (note no number after C)
A2.ge5000        2nd double attribute >= 5000
A1.−2.0−−1.0     1st double attribute in range −2.0 to −1.0
```

If an output filename is provided, a new sites file containing the subset defined by the command will be written in the user's mapset. Header information will be copied from the original sites file and the d.sites.qual command and current window will be written as comments to the new file. Note that only those qualifying sites within the current window are written to the new file. The **−n** flag is useful when calling d.sites.qual from a script, as the number of qualifying sites may be needed to further process the file or report to the user in an interface, as in d.siter.

OPTIONS

This program can be run non−interactively or interactively. If the user gives the name of a GRASS site_lists map and (optionally) specifies other parameter values on the command line the program will be run non−interactively; any parameter values left unspecified are set to their default values (see below). Alternately, the user can type simply **d.sites.qual** on the command line; in this case, the program will prompt the user for parameter values using the standard GRASS parser interface described in the manual entry for parser.

Flags

−n
Output number of sites displayed

Parameters

sitefile
Name of a site file **color**
Sets the current color to that stated:
options: red, orange, yellow, green, blue, indigo, white, black, brown, magenta, aqua, gray, grey
default: gray

size
Size, in pixels, in which the icon is to be drawn
options: 0−1000
default: 5

type
Specify the type of the icon
options: x, diamond, box, +
default: x

rules
qualifying rules for display: [D|C|A]#.RANGE
where RANGE is one of #−#, ge#, le#, eq#

output
Name of a site file to output drawn sites

SEE ALSO

d.mon, d.sites, d.siter

AUTHOR

Bill Brown, UI GMS Laboratory

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.text.freetype − Draws text in the graphics monitor's active display frame using TrueType fonts
(GRASS Display Program)

SYNOPSIS

d.text.freetype
d.text.freetype help
d.text.freetype [**−rpsc**] [**text=**"*string*"] [**east_north=***easting,northing*] [**font=***name*] [**path=***name*] [**charset=***name*] [**color=***name*] [**size=***value*] [**align=***name*] [**rotation=***value*]

DESCRIPTION

d.text.freetype draws text in the graphics monitor's active display frame using TrueType fonts already installed on the user's system.

Flags:

−r
Use radians instead of degrees for rotation

−p
Use pixel coordinates ([0,0] is top left)

−s
Font size is height in pixels

−c
Command mode (Compatibility with d.text)

Parameters:

text=*string*
Text to be displayed. Enclose multiple words "in quotes".

east_north=*easting,northing*
Locates the text at these map coordinates. If this option is omitted, the user will be prompted to use the mouse to specify the location.

font=*name*
Specifies a font listed in the GRASS 'freetypecap' definition file.

path=*/path/filename.ttf*
Specifies a TrueType font with full pathname.

charset=*name*

Sets character encoding for input **text**. (e.g. `ISO−8859−1`)

color=*name*

Sets text color to *name*. It can also be a hexadecimal value of `0xRRGGBB`.
Options: red, orange, yellow, green, blue, indigo, violet, gray, white, and black
Default: *gray*

size=*value*

Sets text height in percent of available frame height.
Default: 10

align=*name*

Aligns text in this way.
Options: ll, lc, lr, cl, cc, cr, ul, uc, ur
l?: Lower, c?: Vertically Centered, u?: Upper,
?l: Left, ?c: Horizontally Centered, ?r: Right
Default: *ll* (Lower Left)

rotation=*value*

Rotates text counterclockwise. The default unit is degrees.
Default: 0

Command mode:

Key

{REQUIRED}
[OPTIONAL]
THIS|orTHAT
+, −

Relative value

r, %, p

Radian, Percentage, and Pixels respectively

Commands

.F {font|path}[:charset] *font*
.C {color_name|0xRRGGBB} *color*
.S [+|−]size[p] *size*
.A {ll|lc|lr|cl|cc|cr|ul|uc|ur} *align*
.R [+|−]rotation[r] *rotation*
.X [+|−]x[%|p] *x−coord*
.Y [+|−]y[%|p] *y−coord*
.L {0|1} *linefeed*
.E [+|−]east[%|p] *x−origin*
.N [+|−]north[%|p] *y−origin*
.. *dot*

To end input, press Ctrl−d on a blank line.

EXAMPLES

Standard mode:

```
d.text.freetype text="GRASS GIS" east_north=2107055,5603133 \
size=7 path=/usr/X11R6/lib/X11/fonts/TTF/luximb.ttf
```

Command mode:

The following example prints three lines: "`GRASS d.text.freetype output`", "`See test.input file`", and "`for more details.`".

```
d.text.freetype -c
        .F /usr/X11R6/lib/X11/fonts/TTF/luximb.ttf:ISO−8859−1
        .S 20p
        .L 0
        GRASS
        .C red
        d.text.freetype
        .C gray
        output
        .L 1
        See test.input file
        for more details.
```

Command mode instructions may also be read directly from a text file:

```
d.text.freetype −c < test.input
```

DEFINITION FILE

To help users select fonts conveniently, the definition file, `$(GISBASE)/etc/freetypecap`, is provided. Its format is as follows:

```
        font:fontpath:charset:color:size:description
        *default_font:fontpath:charset:color:size:description
```

See the file for examples.

SEE ALSO

d.text
d.font
d.title
show.fonts.sh
d.where
parser

AUTHOR

Huidae Cho <grass@geni.cemtlo.com>

Last changed: $Date: 2003/10/12 04:53:31 $

NAME

d.text − Draws text in the active display frame on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.text
d.text help
d.text [**size=***value*] [**color=***name*] [**line=***value*]

DESCRIPTION

d.text draws text in the active display frame on the graphics monitor. Text can be provided through standard input or redirected from a file (using the UNIX redirection mechanism).

Parameters:

size=*value*

Height of letters, stated as a percent of the available display frame height.
Options: 0 − 100
Default: 5

color=*name*

Sets display text color to *name*.
Options: red, orange, yellow, green, blue, indigo, violet, gray, white, and black
Default: *gray*

line=*value*

The screen line number on which the first line of text will be drawn.
(Line 1 is at the top of the active display frame.)
Options: 1 − 1000
Default: 1

In addition to the options provided on the command line, colors, text size, font type, and boldness, can be adjusted with commands in the standard input (i.e., if the user invokes *d.text* without options on the command line, and then assigns values to these options on lines within the standard input). In this case, the user also sees the standard GRASS parser interface.

Commands:

.C color

(where *color* is one of the available colors) causes text appearing on subsequent lines to be drawn in that color.

.S *size*

(where *size* is a percentage within the range 0 to 100) adjusts text size. Note that a size of 10 would allow 10 lines to be drawn in the active display frame, 5 would allow the drawing of 20 lines, and 50 would allow the drawing of 2 lines.

.F *font*

(where *font* is one of the fonts known by the GRASS program *d.font*) manipulates the font type. Available fonts are listed in the GRASS manual entry for *d.font*. The default font type used (if unspecified by the user) is *romans*. Run the GRASS macro *show.fonts.sh* to see what these fonts look like.

.B 1

stipulates that following text be printed in **bold**. This command means *bold on*.

.B 0

turns *bold off* of all text appearing on lines beneath it. (*Bold off* is used by default, if unspecified by the user.)

EXAMPLE

The following command will print the short phrase "This is a test of d.text" in the active display frame using the color yellow, in bold, and using 4/100'ths (4%) of the active frame's vertical space per line:

d.text

```
.C yellow
.S 4
.B 1
This is a test of d.text
```

The user presses *control−d* (the "ctrl" and "d" keys) to end input to *d.text* (equal to EOF).

NOTES

Note that the GRASS command *d.title* creates map TITLEs in a format suitable for input to *d.text*.

d.text needs escape sequences that can be used within lines to change colors, boldness, and perhaps size.

SEE ALSO

d.font
d.title
show.fonts.sh
d.text.freetype
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/05/13 07:41:59 $

NAME

d.title − Outputs a TITLE for a raster map layer in a form suitable for display by *d.text*.
(GRASS Display Program)

SYNOPSIS

d.title
d.title help
d.title [**−f**] **map**=*name* [**color**=*name*] [**size**=*value*]

DESCRIPTION

d.title generates to standard output a string which can be used by *d.text* to draw a TITLE for the raster map layer *name* in the active display frame on the graphics monitor. Output created by *d.title* can be redirected into a file, or piped directly into *d.text* to display the map TITLE created by *d.title*. The map TITLE created will include the map layer's name, TITLE, MAPSET, LOCATION_NAME, geographic region boundary coordinates, and cell resolution.

The user can state program arguments on the command line, or type simply **d.title** on the command line. In the latter case, the program will prompt the user for the parameter values and flag setting using the standard GRASS parser interface.

Flag:

−f
Displays a fancier TITLE.

Parameters:

map=*name*
Name of an existing raster map layer in the user's mapset search path.
color=*name*
Sets the current color to the *name* stated.
Options: red, orange, green, blue, indigo, violet, black, white, gray, yellow, brown, and magenta
Default: *white*
size=*value*
Sets the text size as a percentage of the active display frame's height, to *value*. Floating point values can be used.
Options: 0 − 100
Default: 4.0

EXAMPLE

For example, a user wishing to create a suitable TITLE for the Spearfish, SD *soils* map layer and to display this TITLE in the active display frame on the graphics monitor might type the following:

> **d.title map=***soils* **color=***red* **size=***5* **> TITLE.file**
> **d.text < TITLE.file**

Alternately, the user might pipe *d.title* output directly into *d.text:*

> **d.title map=***soils* **color=***red* **size=***5* | d.text

A file created by *d.title* can be displayed with *d.text*. Information contained in this file takes precedence over the *color* and *size* parameters for *d.text*.

NOTES

The text created with *d.text* will not necessarily fit within the active display frame on the graphics monitor; the user should choose a text size appropriate to this frame.

SEE ALSO

d.font
d.text
parser

AUTHOR

James Westervelt, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2003/10/12 04:55:26 $

NAME

d.area − Draws vector areas to the GRASS display
(GRASS Display/Vector Program)

SYNOPSIS

d.vect.area [**−r**] **map**=*name* [**fillcolor**=*name|R:G:B*] [**linecolor**=*name|R:G:B*] [**catnum**=*value[,value,...]*] [**legend**=*filename|"−"*]

Flags:

−r

Fill all areas with a randomly selected color from a small color palette, and draw all boundaries in black. The palette is similar to the GRASS named colors, but replaces white and black with other colors.

Parameters:

map
Name of existing vector map to be displayed

fillcolor
Color desired for filling polygons. May be specified as a standard grass color name, or as an RGB triplet (e.g. 80:198:255). Fill can be disabled by specifying a color of "none".
Default: white

linecolor
Color desired for drawing area boundaries. May be specified as a standard grass color name, or as an RGB triplet (e.g. 127:127:127). Line drawing can be disabled by specifying a color of "none".
Default: black

catnum
List of category numbers to restrict which areas are drawn. Ignored if either the *−r* flag or the *legend* options are used.

legend
Name of a legend file specifying what color to use for each category (see below). If "−" is specified, the legend will be read from standard input.
Default: none

DESCRIPTION

Use *d.vect.area* to draw labelled areas from a vector map to the GRASS display. Using, the "*−r*" flag, the program will cycle through a small color palette, overriding any *fillcolor* or *linecolor* argument. If a *legend* file is specified, it will be used. Otherwise, the color used for filling areas is as specified.

The *fillcolor* argument may be any of the standard named GRASS colors, or and RGB (red, green, blue) triplet. An RGB triplet must have the form *<num>:<num>:<num>* where *num* is between 0 and 255. If "none" is specified, no fill will be done. Specifying "none" does not make sense for the random color option, since it always draws lines in black.

The *linecolor* argument specifies the color to use for drawing area boundary lines. It has the same format as *fillcolor*, and also accepts the "none" color to disable line drawing.

An optional list of category numbers may be used to restrict which areas are drawn (only valid with the *fillcolor* and *linecolor* options).

More control can be had by creating a *legend* file. At the time of this writing, GRASS does not have any management functions for specifying how vectors should be drawn, so this file must be created by "hand". The easiest way to create such a file, is to copy the category file for the vector map, and edit it to contain color specifications. The format of the *legend* file is *category red green blue [red green blue]*, with one entry per line. The first RGB color specifies the fill color, while the optional second one specifies the boundary line color. If a category for a map is not included in this file, it will not be drawn unless there is a special "default" entry with a category of −1. Boundary lines are not drawn unless either a boundary color is specified for the category, or there is a default boundary color. Using the keyword "none" for either *fillcolor* or *linecolor* in conjunction with a legend will override drawing fill or boundary lines.

Example Legend File

```
−1 255 255 255 127 127 127
1 210 198 168
5 168 198 220
```

In the above, all labelled areas will be drawn in white (255, 255, 255) with dark gray boundaries (127, 127, 127) except those labelled as categories 1 or 5. If the "default" entry had been omitted, then only areas labelled 1 or 5 would have been drawn. Since the entries for 1 and 5 don't specify a color for boundary lines, no boundary lines are drawn.

One may place comments on any line in the legend file by starting it with a non−numeric character ('#' is a good choice). Empty lines are ignored.

NOTES

It is expected that the legend specification aspect of this module will change when an infrastructure for managing vector drawing styles is implemented.

AUTHOR

Eric G. Miller

Last changed: $Date: 2002/02/25 04:18:52 $

NAME

d.vect − Displays GRASS vector data in the active frame on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.vect
d.vect help
d.vect [−fv] map=*name* [**color=***name*] **cat**=*value[,value,...]*

DESCRIPTION

d.vect displays the user−named binary vector file in the active display frame on the graphics monitor, in the user−named color. The vector file must already exist under a mapset listed in the user's current mapset search path.

This program can be run either non−interactively or interactively. If the user gives the name of a binary vector map to be displayed and (optionally) specifies a color for vector display, the program will run non−interactively. (If the user gives a vector map name but no vector color on the command line, the named vector map layer will be displayed in the default color *white*.) Alternately, the user can simply type **d.vect** on the command line; in this case, the program will prompt the user for parameter values.

Flags:

−f
Fill areas of selected polygons

−v
Run verbosely

Parameters:

map=*name*
Vector map layer to be displayed in the active display frame.
Must be binary vector file in the user's current mapset search path.

color=*name*
Sets vector display color to *name*.
Options: white, red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, and black
Default: *white*

catnum=value[,value,...]
Category vnumbers to be displayed

NOTES

To display text labels, the d.vect.labels module may be used.

Selected vector polygons can be optionally filled with color. To fill all vector polygons with colors, d.area can be used.

This program needs to be upgraded with features that would:

1. Allow different line colors, weights, and textures to be assigned to vectors with different category values.
2. Allow different fill patterns and colors to be assigned to areas.
3. Be linked with an associated program identifying these options.

SEE ALSO

d.area, *d.rast*, *d.sites*, *d.vect.labels*, *d.what.vect*, *g.mapsets*, *parser*

AUTHOR

Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

Merge with d.vect.cats: Markus Neteler

Last changed: $Date: 2002/03/01 00:08:31 $

NAME

d.vect.labels − Labels vectors using the attribute value or the category value.
(GRASS Display Program)

SYNOPSIS

d.vect.labels
d.vect.labels [−smv] map=*name* [**attr=**string|cat|coords] [**size=***number*] [**color=***colorname*] [**backgr=***colorname*] [**border=***colorname*] [**font=***fontname*]

DESCRIPTION

d.vect.labels performs automated placement of labels for a GRASS *vector map* on the current graphics monitor. Labels can be chosen from the category value (numerical), a string attribute, or the labels positions coordinates. The default uses the first string attribute. The user may select the color and sizing of the labels. Labels are only drawn within the current region.

This program can be run non−interactively or interactively. If the user gives the name of a GRASS *vector map* and (optionally) any other parameters, the command will be run non−interactively. Any parameter left unspecified will be set to its default value (see below). Alternatively, the user can type *d.vect.labels* at the command line and be prompted for the parameter values using the standard GRASS *parser* interface.

Parameters:

−s
save support sites file
−m
mark labels on the map
−v
Verbose output

Parameters:

map=*name*
Name of a GRASS *vector map* map in the user's current mapset search path.
attr=*string|cat|double|coords|dim*
Type of attribute to use for the label.
Default: *string attr* option is set to *string, cat* or *coords*.
Default: *string*
size=*number*
The size of letters in pixels.

Default: *10*

color=*colorname*

The GRASS *colorname* to use for the label text.
Options: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white or black
Default: *white*

backgr=*colorname*

Color to use for text box background.
Options: grey, red, white, magenta, brown, blue, indigo, yellow, black, orange, green, violet, none
Default: none

border=*colorname*

Color to use for text box border.
Options: grey, red, white, magenta, brown, blue, indigo, yellow, black, orange, green, violet, none
Default: none

font=*fontname*

The GRASS *fontname* to use for the label.
Options: cyrilc, gothgbt, gothgrt, gothitt, greekc, greekcs, greekp, greeks, italicc, italiccs, italict, romanc, romancs, romand, romans, romant, scriptc or scripts
Default: *romans*

SEE ALSO

d.sites
d.rast.labels
d.site.labels
d.vect
g.region
parser

AUTHOR

Stefano Merler, ITC−Irst, based on d.site.labels

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.area − Draws labelled vector lines to the GRASS display
(GRASS Display/Vector Program)

SYNOPSIS

d.vect.line **map**=*name* [**color**=*name*|*R:G:B*] [**catnum**=*i[,j,k,...,n]*] [**legend**=*filename*|*"−"*]

Parameters:

map
: Name of existing vector map to be displayed

color
: Color desired for drawing lines. May be specified as a standard grass color name, or as an RGB triplet (e.g. 80:198:255).
Default: white

catnum
: List of category numbers to restrict which lines are drawn. Ignored if the *legend* option is used.

legend
: Name of a legend file specifying what color to use for each category (see below). If "−" is specified, the legend will be read from standard input.
Default: none

DESCRIPTION

Use *d.vect.line* to draw labelled lines from a vector map to the GRASS display. By default, *d.vect.line* will draw all labelled lines in the vector map with white. The lines drawn may be restricted via the *catnum* argument or by using a *legend* file.

The *color* argument may be any of the standard named GRASS colors, or and RGB (red, green, blue) triplet. An RGB triplet must have the form *<num>:<num>:<num>* where *num* is between 0 and 255.

An optional list of category numbers may be used to restrict which lines are drawn (option ignored if a *legend* option is given).

More control can be had by creating a *legend* file. At the time of this writing, GRASS does not have any management functions for specifying how vectors should be drawn, so this file must be created by "hand". The easiest way to create such a file, is to copy the category file for the vector map, and edit it to contain color specifications. The format of the *legend* file is *category red green blue*, with one entry per line. If a category for a map is not included in this file, it will not be drawn unless there is a special "default" entry with a category of −1.

Example Legend File

```
−1 255 255 255
1 210 198 168
5 168 198 220
```

In the above, all labelled lines will be drawn in white (255, 255, 255) except those labelled as categories 1 or 5. If the "default" entry had been omitted, then only lines labelled 1 or 5 would have been drawn.

One may place comments on any line in the legend file by starting it with a non−numeric character ('#' is a good choice). Empty lines are ignored.

NOTES

It is expected that the legend specification aspect of this module will change when an infrastructure for managing vector drawing styles is implemented.

AUTHOR

Eric G. Miller

Last changed: $Date: 2002/02/06 06:19:26 $

NAME

d.vect.pg − Display select vectors from an existing vector map. (GRASS−RDBMS Interface Display Program)

SYNOPSIS

d.vect.pg
d.vect.pg help
d.vect.pg [−f] key=*name* **tab**=*name* [**where**=*name*] **map**=*name* [**color**=*name*]

ALTERNATE

d.vect.pg −s help
d.vect.pg −s sql=*filename* **map**=*name* [**color**=*name*]

DESCRIPTION

d.vect.pg displays select vectors from an existing vector map based on the unique values in a database column. Each row returned by a user constructed database query will be associated with a vector feature which is subsequently drawn on the graphics display in the active frame. The user can control the color of the vector draw by specifying a color on the command line.

COMMAND LINE OPTIONS

Flag:

−f
: Fill polygons selected on the query criteria

Parameters:

key=databasecolumnname
: Column in table "tab" of the currently selected database containing values corresponding to the vector maps category values. Table is designated on the command line by tab=tablename and vector is designated on the command line by map=mapname.

tab=databasetablename
: Table in the currently selected database containing a column which has values corresponding to vector category values in the map designated by map=map.

where=SQLwhereclause
: SQL "where" clause which specifies the query criteria to be used in subsetting the database. The field

names specified in the where option must indicate the column(s) to be used, the operators to be used in the evaluation and the values which the data in the column will be evaluated against.

If the database column used as the selection criteria is a character field then the associated value must be placed in quotes. To determine the data types associated with columns in the currently selected database use the *g.column.pg* command with the *−v* flag.

Queries which are more complex are best implemented using the *−s* flag and a prepared SQL file.

map=map
Name of an existing vector map with category values which correspond to values in a specified column in the currently selected database.

color=name
Color to draw vectors in
Colors: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white, black

ALTERNATE COMMAND LINE USAGE

The alternate command line usage is provided to simplify the process of retrieving information from more than one table in the query criteria. The alternate command line structure is selected using the the *[−s]* flag on the command line. When using this flag the user must include the name of a text file on the command line as well. This file must include a complete, well formed SQL SELECT statement specifying the query criteria to retrieve values which correspond to category values in a GRASS data layer.

Flag:

−s
SQL select statements are input from a prepared file

−f
Fill polygons selected on the query criteria

Parameters:

sql=filename
SQL statements specifying well formed selection criteria.

map=name
Name of an existing vector map

color=name
Color to draw vectors in
Colors: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white, black

EXAMPLE:

1. *d.vect.pg −f key=rec_id map=kuruma_id tab=info_kuruma where='type_id >32 and type_id < 38' color=red*

Result: only polygons with type in 33−37 range would be displayed in red color..

2. *d.vect.pg −f −s sql=oak.sql map=kuruma_id*

oak.sql is:

select rec_id from info_kuruma where type_id > 32 and type_id <38;

Result: this would be the same as in the first example.

BUGS

none

NOTE

This program requires the Postgres database software.

SEE ALSO

g.column.pg, g.select.pg, g.stats.pg, g.table.pg, d.rast.pg, d.site.pg, d.what.r.pg, d.what.s.pg, d.what.v.pg, r.reclass.pg, r.rescale.pg, v.reclass.pg

AUTHOR

Original Informix SQL−tools: James A. Farley, Wang Song, and W. Fredrick Limp University of Arkansas, CAST

Postgres modifications: Janne Soimasuo, Faculty of Forestry, University of Joensuu, Finland.

Updated to GRASS 5 by Alex Shevlakov (sixote@yahoo.com)

NAME

d.what.db − View/edit attributes of selected DB site.
(GRASS Display Program)

SYNOPSIS

d.what.db
d.what.db help
d.what.db table=*table_name* **key=***colun* **x=***column_name* **y=***column_name* [**where=***SQL_where_clause*]

DESCRIPTION

d.what.db opens window containing attributes of site in selected location. User can change attributes and update row in database.

Parameters:

table=*name*
    Name of a *table* in the database.
key=*column_name*
    Name of a *column* update will be based on.
x=*column_name*
    Name of a *column* containing x coordinates.
y=*column_name*
    Name of a *column* containing y coordinates.

SEE ALSO

d.connect

AUTHOR

Radim Blazek, Radim.Blazek@dhv.cz

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.what.rast − Allows the user to interactively query the category contents of multiple raster map layers at user−specified locations within the current geographic region.
(GRASS Display Program)

SYNOPSIS

d.what.rast
d.what.rast help
d.what.rast [**−1tc**] [**map=***name*[,*name*,...]] [**fs=***name*]

DESCRIPTION

d.what.rast outputs the category values and labels associated with cell(s) at user−specified location(s) on user−named raster map layer(s).

The program will query the contents of raster map layer(s) named by the user on the command line. These map layers must exist in the user's current mapset search path. If the user does not name any raster map layers on the command line, *d.what.rast* will query the category contents of whatever raster map layer is already displayed in the active frame on the graphics monitor.

The program activates the mouse, and expects the user to indicate the cell location(s) to be queried by depressing a mouse button over desired location(s) within the current geographic region in the active display frame on the graphic monitor.

OPTIONS

Flags:

−1
: Identify and query just one point location. Only one mouse click is executed. This option is provided for shell scripts and programs that want to obtain only one point from the user.

−t
: Provide only terse output. This option is provided to simplify the parsing of output by other programs.

−c
: Print out column and row for entire map region.

Parameters:

map=*name*

Name of existing raster map layer(s). Limit: 15 maps
Default: Query map currently displayed in the active graphics frame.

fs=*name*

Output field separator to be used (in terse mode only).
Default: **:**

d.what.rast output consists of the geographic coordinates of the location pointed to, and, for each map layer, the map layer name, the category value, and category label in the named raster map layers at this location.

EXAMPLE

It is helpful, but not necessary, to first display a map to be used for reference in the active display frame before running *d.what.rast*. For example, the user might type the following series of commands and receive the output below.

d.rast map=*soils*
To first display the *soils* map in the active frame.

d.what.rast map=*soils,aspect*
User then moves the mouse to desired location on the displayed *soils* map layer, and presses the left mouse button to query the category contents of the *soils* and *aspect* maps at this geographic location. The program then outputs the below information to the user's terminal.

```
617112(E) 3732014(N)
soils in PERMANENT (44)Nunn clay loam, NdC
aspect in PERMANENT (20)15 degrees north of west
```

The first line of output gives the easting (E) and northing (N) coordinates of the geographic location at which the user clicked the mouse. The subsequent two lines give the map name and mapset, map category value (within parentheses), and map category label corresponding to this user−selected map location, for each of the maps queried by the user.

The query may be repeated as often as desired using the left mouse button. The right button on the mouse is used to quit the *d.what.rast* session.

Users can set the **−t** flag to obtain a terse output from *d.what.rast*. This is useful when the user wishes output to be parsed by another program (like *awk*). If the **−t** flag is set, users can also select the field separator used (with the **fs**=*name* option), or elect to use the default **:** field separator. In this case, the command

d.what.rast −t map=*soils,aspect*

produces output in the form shown below. The first line of output gives the easting, northing, and the mouse button return value (see NOTES below). Each subsequent line gives the map name and its mapset, the category value, and category label of the point specified on the user−named raster map layers. The default output field separator, a colon, is used since none was specified on the command line.

```
        617112:3732014:1
        soils@PERMANENT:44:Nunn clay loam, NdC
        aspect@PERMANENT:20:15 degrees north of west
```

Users can also use this program inside of shell scripts that require as input a map category value and a mouse button depressed. Users can choose an option to run *d.what.rast* only once, and return only the map category value found and the number of the mouse button depressed.

NOTES

Mouse button return values are as follows: 0 indicates no button was pressed, 1 indicates that the left mouse button was pressed, 2 indicates the middle button was pressed, and 3 indicates that the right mouse button was pressed.

d.what.rast will always print its output to the terminal screen. *d.what.rast* output can be redirected into a file; however, if it is, the output will go both to the screen and to the file. For example:

> **d.what.rast map=***soils,aspect* > **what.out**

will both send *d.what.rast* output to the screen and capture its output in the file named *what.out*.

The maximum number of raster map layers that can be queried at one time is 15.

d.what.vect allows the user to interactively query the category contents of multiple vector map layers at user−specified locations.

SEE ALSO

d.rast
d.what.vect
d.where
g.region
parser

AUTHOR

Michael Shapiro, U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/03/19 09:12:58 $

NAME

d.what.r.pg − Report database attributes associated with a raster category value at a specified location. (GRASS−RDBMS Attribute Interface Program)

SYNOPSIS

d.what.r.pg
d.what.r.pg help
d.what.r.pg map=*name* **tab**=*name* **col**=*name* [**hv**=name]

ALTERNATE

d.what.r.pg −s help
d.what.r.pg −s sql=*filename* **map**=*name*

DESCRIPTION

d.what.r.pg reports database attributes from the currently selected database associated with a raster category at a specific location on a raster map. The currently selected database is identified by the GRASS environment variable $PG_DBASE which is set using the *g.select.pg* GRASS−RDBMS interface tool. If this environment variable is not set the program terminates with a message to the user. The current raster location is selected by the user with the mouse. If the alternate form of this command is selected with the *−s* flag the user has greater control over the manner in which the attribute information is displayed. Using the *−s* option the attributes from more than one table can be returned and displayed.

COMMAND LINE OPTIONS

Parameters:

map=map
: Name of an existing raster map with category values linked to the currently selected database.

tab=databasetablename
: Table in the currently selected database containing a column associated with raster category values.

col=databasecolumnname
: Column associated with raster category values.

hv
Type of database output − [h/v]
default: h

ALTERNATE COMMAND LINE USAGE

The alternate command line usage is provided to simplify the process of retrieving information from more than one table in the query criteria. The alternate command line structure is selected using the the *[−s]* flag on the command line. When using this flag the user must include the name of a text file on the command line as well. This file must include a complete, well formed SQL SELECT statement specifying the query criteria needed to retrieve values which correspond to category values in a GRASS data layer.

The user must also specify the name of the GRASS data layer containing these category values on the command line. The following examples illustrate the syntax which must be used in constructing a SELECT statement for use with the *−s* flag.

EXAMPLE:

1. *d.what.r.pg −s sql=raster.sql map=blag_forest_kur*

and raster.sql is:

select rec_id, type_id from info_kuruma where type_id = ?

Result:
Reports two fields from the table info_kuruma associated with the raster category value at the current mouse location the raster map.

2. *d.what.r.pg −s sql=raster2.sql map=blag_forest_kur*

and raster2.sql is:

select rec_id, type_id, census_info.type_good from info_kuruma where type_id = ? and rec_id = census_info.rec_id

Result:
Reports fields from two tables info_kuruma and census_info associated with the raster category value at the current mouse location the raster map.

3. *d.what.r.pg −s kur_rast_id sql=process.sql*

process.sql is:

update census_info set type_good = '1' where rec_id = ? and info_kuruma.type_id = 35 and info_kuruma.rec_id = rec_id;

Result: this would only change field of the census_info table for the clicked polygon or line,
if the corresponding type taken from *info_kuruma* table is equal to 35.

Flag:

−s

SQL select statements are input from a prepared file.

Parameters:

sql=filename

SQL statements specifying well formed selection criteria.

map=map

Name of an existing raster map with category values linked to a database.

hv

Type of database output − [h/v]
default: h

BUGS

None known.

NOTE

This program requires the Postgres database software.

SEE ALSO

g.column.pg, *g.select.pg*, *g.stats.pg*, *g.table.pg*, *d.rast.pg*, *d.site.pg*, *d.vect.pg*, *d.what.s.pg*, *d.what.v.pg*, *r.reclass.pg*, *r.rescale.pg*, *v.reclass.pg*

AUTHOR

Original Informix SQL−tools: James A. Farley, Wang Song, and W. Fredrick Limp University of Arkansas, CAST

Postgres modifications: Janne Soimasuo, Faculty of Forestry, University of Joensuu, Finland.

Updated to GRASS 5 by Alex Shevlakov (sixote@yahoo.com)

NAME

d.what.sites − interactively query a single site list descriptions
(GRASS Display Program)

SYNOPSIS

d.what.sites
d.what.sites help
d.what.sites [**−1tq**] [**sitefile=***name*] [**dcolor=***color*] [**recolor=***color*]

DESCRIPTION

d.what.sites outputs the east, north coordinate values and descriptions with site(s) nearest to the user−specified location(s) on user−named site list.

The program will query the contents of site list named by the user on the command line. These site lists must exist in the user's current mapset search path.

The program activates the mouse, and expects the user to indicate the site location(s) to be queried by depressing a mouse button over desired location(s) within the current geographic region in the active display frame on the graphic monitor.

Flag:

−1
Identify and query just one point location. Only one mouse click is executed. This option is provided for shell scripts and programs that want to obtain only one point from the user.

−t
Provide only terse output. This option is provided to simplify the parsing of output by other programs.

−q
Load quietly. Do not show progress in loading the sites file into memory.

Parameters:

sitefile=*name*
Name of existing site list.

dcolor=*name*
Sets the display color to name. Options: red, orange, yellow, green, blue, indigo, violet, magenta, brown, gray, white, and black Default: red

recolor=*name*
Sets the redraw color to name. Options: red, orange, yellow, green, blue, indigo, violet, magenta,

brown, gray, white, and black Default: yellow

size=*value*

Size, in pixels, in which the site icons are to be drawn.
Options: 0−1000
Default: 5

type=*name*

The type of icon to be displayed at site locations.
Options: x, diamond, box, +
Default: x

EXAMPLE

It is helpful, but not necessary, to first display a site list to be used for reference in the active display frame before running *d.what.sites*. For example, the user might type the following series of commands and receive the output below.

d.sites sitefile=*wells*

To first display the well site list in the active frame.

d.what.sites sitefile=*wells*

User then moves the mouse to desired location on the displayed well site list, and presses the left mouse button to query the category contents of the nearest well location. The program then outputs the below information to the user's terminal.

```
the closest site is: −613988.990000 east, 804343.040000 north
08376300 1968 1980 13
```

The query may be repeated as often as desired using the left mouse button. The right button on the mouse is used to quit the *d.what.sites* session. The middle button will allow to redraw the site list.

NOTES

d.what.sites will always print its output to the terminal screen. *d.what.sites* output can be redirected into a file; however, if it is, the output will go both to the screen and to the file. For example:

d.what.sites sitefile=*wells* > **what.out**

will both send d.what.sites output to the screen and capture its output in the file named *what.out*.

SEE ALSO

d.sites
d.what.rast
d.what.vect
d.where
g.region
parser

AUTHOR

Raghavan Srinivasan, Blackland Research Center, TAES, Temple, TX 76502

Last changed: $Date: 2002/01/25 05:45:33 $

NAME

d.what.s.pg − Report database attributes associated with a specific point location. (GRASS−RDBMS Attribute Interface Program)

SYNOPSIS

d.what.s.pg
d.what.s.pg help
d.what.s.pg tab=name **xcol**=name **ycol**=name [**xpos**=value] [**ypos**=value] **distance**=value [**hv**=name]

ALTERNATE

d.what.s.pg −s help
d.what.s.pg [−s] [**sql**=file] **distance**=value [**hv**=name]

DESCRIPTION

d.what.s.pg reports database attributes in the currently selected database which are associated with a specific point location. The current database is identified by the GRASS environment variable $PG_DBASE which is set using the *g.select.pg* GRASS−RDBMS interface tool. The location to query is selected by the user with the mouse. The radius around this point to search is specified by the distance argument. Sites are selected by comparing the values in the database columns containing X and Y coordinate data against the coordinate data at the current mouse location.The columns in the table containing the coordinate values are returned in the report. If information from more than one table is required use the *−s* flag and the alternate command line format (see below). The user can also provide the center of the search point and the distance around it to find sites within the circle around the search center non−interactively.

COMMAND LINE OPTIONS

Parameters:

tab=databasetablename
Table containing X, Y coordinate values
xcol=databasecolumnname
Column containing the X coordinate (E/W) value.
ycol=databasecolumnname
Column containing the Y coordinate (N/S) value.
distance=value
Radius distance from the current mouse location to conduct database search.

xpos=value
X coord. (E/W) of search.

ypos=value
Y coord. (N/S) of search.

hv=name
(h)Horizontal/(v)vertical output (default is v)

ALTERNATE COMMAND LINE USAGE

The alternate command line structure is selected using the the *[−s]* flag on the command line. When using this flag the user must include the name of a text file on the command line as well. This file must include a complete, well formed SQL SELECT statement specifying the query criteria.

The user must also specify the name of the GRASS data layer containing these category values on the command line. The following example illustrates the syntax which must be used in constructing a SELECT statement for use with the *−s* flag.

EXAMPLE 1. (single table)

d.what.s.pg −s sql=izb.sql distance=100

here *izb.sql*:

SELECT * from izb where point(xcoor,ycoor)

EXAMPLE 2. (two tables)

d.what.s.pg −s sql=izb2.sql distance=100

here *izb2.sql*:

SELECT *, huts_info.description from izb where huts_info.num=num and point(xcoor,ycoor)

Flag:

−s
SQL select statements are input from a prepared file.

Parameters:

sql=filename
SQL statements specifying well formed selection criteria. These criteria must include the database table name, the Xcol and Ycol names as well as the search distance. Additional criteria may be placed in the "where" clause to further subset information returned from the database.

distance=value
Radius distance from the current mouse location to conduct database search.

hv=name

(h)Horizontal/(v)vertical output (default is v)

BUGS

None known.

NOTE

This program requires the Postgres database software.

SEE ALSO

g.column.pg, *g.select.pg*, *g.stats.pg*, *g.table.pg*, *d.rast.pg*, *d.site.pg*, *d.vect.pg*, *d.what.r.pg*, *d.what.v.pg*, *r.reclass.pg*, *r.rescale.pg*, *v.reclass.pg*

AUTHOR

Original Informix SQL−tools: James A. Farley, Wang Song, and W. Fredrick Limp University of Arkansas, CAST

Postgres modifications: Janne Soimasuo, Faculty of Forestry, University of Joensuu, Finland.

Updated to GRASS 5 by Alex Shevlakov (sixote@yahoo.com)

NAME

d.what.vect − Allows the user to interactively query the category contents of a (binary) vector map layer at user−selected locations within the current geographic region.
(GRASS Display Program)

SYNOPSIS

d.what.vect
d.what.vect help
d.what.vect [**−1t**] **map**=*name* [**table**=*name*] [**key**=*name*]

DESCRIPTION

d.what.vect outputs the category value(s) associated with user−specified location(s) in a user−specified vector map layer. This program currently returns only category values for line types in the user−specified vector file.

The program will query the contents of the vector map layer named by the user on the command line. This map layer must exist in the user's current mapset search path.

The program activates the mouse, and expects the user to indicate the location(s) to be queried by depressing a mouse button over desired location(s) within the current geographic region in the active display frame on the graphic monitor.

Alternatively user can specify **table=** and **key=** and window with database attributes will be opened for each selected object with assigned category.

Flag:

−1

Identify and query just one point location. Only one mouse click is executed. This option is provided for shell scripts and programs that want to obtain only one point from the user.

−t

Provide only terse output. This option is provided to simplify the parsing of output by other programs. Database connection must be set up by db.connect first.

Parameter:

map=*name*

Name of an existing binary vector map in the user's mapset search path.

table=*name*

Name of database table containing attributes.

key=*name*

Name of database table key (id) column.

EXAMPLE

A sample *d.what.vect* session is given below. Although it is not necessary that the user first display a vector map to be queried in the active display frame, it is helpful to have a map displayed there for reference.
d.vect map=*roads.24000*
Displays the 1:24,000 scale *roads* vector map layer on the graphics monitor.

d.what.vect map=*roads.24000*
After typing this, the user moves the mouse to a desired location on the displayed *roads* map layer, and presses the left mouse button to query the category value of the *roads* vector map at this location. The program then outputs the category value of a line type corresponding to this user−selected map location, for the vector map queried by the user.
The query may be repeated as often as desired using the left mouse button. The right button on the mouse is used to quit the *d.what.vect* session.

Users can also use this program inside of shell scripts that require as input a map category value and a mouse button depressed. Users can set the −1 flag to run *d.what.vect* only once, and return only the map category value found and the number of the mouse button depressed. (Mouse button return values are as follows: 0 indicates no button was pressed, 1 indicates that the left mouse button was pressed, 2 indicates the middle button was pressed, and 3 indicates that the right mouse button was pressed.)

NOTES

Currently, *d.what.vect* only outputs category values for lines. It does not output category labels for lines, nor output category values or category labels for areas in a vector file.

d.what.vect will always print its output to the user's terminal screen. *d.what.vect* output can be redirected into a file; however, if it is, the output will go both to the screen and to the file. For example:

> **d.what.vect map**=*roads* > **what.out**

will both send *d.what.vect* output to the screen and capture its output in the file named *what.out*.

d.what.rast can be used to interactively query the map category contents of multiple raster map layers at user−selected locations.

SEE ALSO

d.rast
d.vect
d.what.rast
g.region
db.connect
parser

AUTHOR

Jim Hinthorne, Central Washington University
Upgrades: Dennis Finch, National Park Service

NAME

d.what.v.pg − Report database attributes associated with vector features.

SYNOPSIS:

d.what.v.pg [−f] map=name **tab=**name **col=**name **[color=**name**]**
[fillcolor=name**] [hv=**name**]**

d.what.v.pg [−sf] sql=file **map=**name **[color=**name**] [fillcolor=**name**]**
[hv=name]

DESCRIPTION

Flags:
−s Use [−s] for SQL command file input.
−f Fill polygons?

Parameters:

sql

Name of the SQL command file

map
Vector map to run query on

tab
Postgres table with categories

col
Column with categories from this table

color
Selected lines color
default: yellow

fillcolor
Selected areas color (for fill)
options: red,orange,yellow,green,blue,indigo,white,black,brown, magenta,aqua,gray,grey

default: gray

hv
Type of database output − [h/v]
default: v

tab is Postgres table containing column linked to vector attribute values, **col** − column associated with vector attribute values, **map** is name of an existing vector map with attribute values linked to the currently selected database.

Reports database attributes from the currently selected database which are associated with specific vector features identified using the mouse. The currently selected database is identified by the GRASS environment variable $PG_DBASE which is set using the *g.select.pg* GRASS−RDBMS interface tool. If this environment variable is not set the program terminates with a message to the user. If the alternate form of this command is selected with the *−s* flag the user has greater control over the manner in which the attribute information is displayed.

d.what.v.pg is used to query lines/areas with mouse. Like *d.what.vect*, it lists attributes and optionally fills queried areas (like v.area). It may be used for the UPDATE command in Postgres through the input SQL file (see Example 2). The idea is to prepare an arbitrary algorithm, code it in SQL, and then update DB with clicking on chosen polys/lines. You may consider this as piping through various filters anything you select on screen.

The database information can be optionally output as comma−separated list (horizontal) which is convenient for feeding it to other programs like table editors.

EXAMPLE:

1. *d.what.v.pg −s sql=census.sql map=census*

and census.sql is:

select slope, type_id from info_kuruma where rec_id = ?

Result: only two fields information is displayed on screen when the mouse is clicked upon the vector object.

2. *d.what.v.pg −s sql=process.sql map=census*

process.sql is:

update census_info set type_good = '1' where rec_id = ? and info_kuruma.type_id = 35 and info_kuruma.rec_id = rec_id;

Result: this would only change field of the *census_info* table for the clicked polygon or line,
if the corresponding type taken from *info_kuruma* table is equal to 35 (the last limitation is a filter for any polygon/line picked from the map with mouse) .

Flags: SQL select statements are input from a prepared file.

Parameters: SQL statements specifying well formed selection criteria. Name of an existing vector map.

BUGS

1. Tcl−Tk modules require input "where" clause restricted to one rule typed without "whitespaces", (unlike terminal input where number of subclause is not limited).

This program requires the Postgres database software.

SEE ALSO

g.column.pg, *g.select.pg*, *g.stats.pg*, *g.table.pg*, *d.rast.pg*, *d.site.pg*, *d.vect.pg*, *d.what.r.pg*, *d.what.s.pg*, *r.reclass.pg*, *r.rescale.pg*, *v.reclass.pg*

AUTHOR

Original Informix SQL−tools: James A. Farley, Wang Song, and W. Fredrick Limp University of Arkansas, CAST

Postgres modifications: Janne Soimasuo, Faculty of Forestry, University of Joensuu, Finland.

Updated to GRASS 5 by Alex Shevlakov (sixote@yahoo.com)

NAME

d.where − Identifies the geographic coordinates associated with point locations in the active frame on the graphics monitor.
(GRASS Display Program)

SYNOPSIS

d.where
d.where help
d.where [**−1d**] [**spheroid=***name*]

DESCRIPTION

d.where is an *interactive* program that allows the user, using the pointing device (mouse), to identify the geographic coordinates associated with point locations within the current geographic region in the active display frame on the graphics monitor.

If the user runs *d.where* without specifying the name of a spheroid on the command line, each mouse click will output the easting and northing of the point currently located beneath the mouse pointer. A mouse−button menu is presented so the user knows which mouse buttons to use. The output is always printed to the terminal screen; if the output is redirected into a file, it will be written to the file as well.

Mouse buttons:

```
     Left:   where am i
     Middle: draw to/from here
     Right:  quit this
```

The left mouse button prints the coordinates at the selected point, the middle mouse button allows to query two points (they are connected by a line for convenience). By using the right mouse button the module is left.

Flag:

−1

Only one mouse click is executed.
This option is provided for shell scripts and programs which want to obtain only one point from the user. The output is only written to stdout, unless redirected into a file. The geographic location and mouse button pressed are output.

−d

Output lat/long in decimal degree

Parameter:

spheroid=*name*

Name of a spheroid (for latitude/longitude coordinate conversion).
Options: australian, bessel, clark66, clark80, everest, international, wgs72, wgs84

NOTES

This program uses the current geographic region setting and active frame. It is not necessary, although useful, to have displayed a map in the current frame before running *d.where*. The *−d* flag allows to optionally output latitude/longitude coordinates pair(s) in decimal degree rather than DD:MM:SS format. In other projections the flag is silently ignored.

SEE ALSO

d.what.rast
d.what.vect
g.region

AUTHORS

James Westervelt,
Michael Shapiro,
U.S. Army Construction Engineering Research Laboratory

Last changed: $Date: 2003/08/25 07:48:11 $

NAME

d.zoom − Allows the user to change the current geographic region settings interactively, with a mouse. *(GRASS Display Program)*

SYNOPSIS

d.zoom
d.zoom help
d.zoom [**−fhj**] [**rast=***name*[,*name*,...]] [**vector=***name*[,*name*,...]] [**site=***name*[,*name*,...]] [**zoom=***value*]

DESCRIPTION

d.zoom allows the user to interactively adjust the settings of the current geographic region using a pointing device such as a mouse. Like *g.region*, *d.zoom* re−defines the settings of the geographic region. However, *d.zoom* allows the user to change the current region settings interactively, by either outlining the new region perimeter with a mouse, unzoom or pan the region. The graphics display is updated by the module.

OPTIONS

Flag:

−f
Full menu with pan and zoom options.

−h
Handheld mode, zoom by window with left button only.

−j
Just redraw given maps using default colors.

Parameter:

rast*=name[,name,...]*
Name of raster map(s)

vector*=name[,name,...]*
Name of vector map(s)

site*=name[,name,...]*
Name of site map(s)
Options: zoom

zoom*=value*
magnification: >1.0 zooms in, <1.0 zooms out
options: 0.001−1000.0
default: 1.0

After the user types the command **d.zoom** a mouse−button menu will appear directing the user. Additionally the region settings can be modified by running *g.region*. The user may also alter the current geographic region by running the "region" option of the *d.display* program.

NOTES

Although it is not necessary that the user display a map in the active display frame before running *d.zoom*, it is helpful to do this for reference.

SEE ALSO

d.display
d.erase
d.pan
d.rast
d.sites
d.vect
g.region

AUTHOR

Michael Shapiro, U.S. Army Construction Engineering Research Laboratory
Improvements by Huidae Cho and Markus Neteler

Last changed: $Date: 2002/04/29 14:16:06 $

NAME

d.zoom.last.sh − Allows the user to change the current geographic region settings interactively and tracks the last region settings. (An addition to the GRASS display program *d.zoom*).
(GRASS Shell Script)

SYNOPSIS

d.zoom.last.sh

DESCRIPTION

d.zoom.last.sh is a UNIX Bourne shell macro which allows the user to interactively adjust the settings of the current geographic region using a pointing device such as a mouse, and automatically saves the last region settings. The last region can be restored by a GRASS command:

    g.region **region**=last

or

    g.region last

This script and the above restoring command function as an 'undo' in editing.

NOTES

Only the last region is saved. When zooming multiple times, regions before the last are NOT available. Region−file 'last' is reserved for *d.zoom.last.sh*.

FILES

See the file **`$GISBASE/scripts/d.zoom.last.sh`**.

SEE ALSO

d.zoom
g.region.

AUTHOR

Jianping Xu,
John Bognar,
Rutgers University

Last changed: $Date: 2002/01/25 05:45:33 $