GRASS Paint Commands

p.chart
p.colors
p.icons
p.labels
p.map
p.map.new
p.ppm
p.select
p.vrml

Other Commands

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

---

NAME

p.chart − Prints the color chart of the currently selected printer.
(GRASS Hardcopy Output Program)

SYNOPSIS

p.chart

DESCRIPTION

This function prints the color chart of the user's hardcopy color printer. The colors in the chart are numbered with the (device−dependent) number associated with each color. This function is useful both for the *p.colors* command as well as for color selection with *p.map*. The user must select a printer using *p.select* before running this command.

SEE ALSO

d.colors
p.colors
p.map
p.select
r.colors

AUTHOR

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

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

NAME

p.colors − Allows the user to develop a color table that associates map categories with user−specified printer colors.
(GRASS Hardcopy Output Program)

SYNOPSIS

p.colors

DESCRIPTION

This function allows the user to modify a color table for a raster map layer, by assigning colors to the categories in the raster map layer based on printer color numbers (instead of red, green, blue percentages).

The user will need to have the printer color chart available to properly select the colors (see manual entry for *p.chart*).

The *p.colors* function allows the user to exercise perfect control over the colors which appear in the hardcopy image.

NOTES

If the user assigns a printer color number outside of the printer's range, *p.colors* will not complain, but will not save the out−of−range printer color number to the raster map layer's color table.

This command modifies the color table associated with a map layer, and will therefore also affect the colors in which a map layer is displayed on the user's graphics monitor. (See map color table files stored under $LOCATION/colr and $LOCATION/colr2.)

SEE ALSO

d.colors
p.chart
p.map
r.colors

AUTHOR

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

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

NAME

p.icons − Creates and modifies icons for map display and output.
(GRASS Hardcopy Output Program)

SYNOPSIS

p.icons

DESCRIPTION

This program allows the user to create and maintain icons which are used by the *p.map* and *d.icons* commands to depict sites.

FILES

Icon files created by the user are stored under $LOCATION/icons.

SEE ALSO

d.icons
d.points
d.sites
p.map
s.menu

AUTHOR

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

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

NAME

p.labels − Create labels for hardcopy maps.
(GRASS Hardcopy Output Program)

SYNOPSIS

p.labels

DESCRIPTION

This module allows the user to create or modify labels files. These labels files, which are stored in the database, define text information for printing with *p.map* and for graphics display with *d.paint.labels*. Each label has components which determine the text, the location of the text on the image, its size, and the background for the text.

The interface is a screen−oriented input/edit layout. Each label is entered with a single screen. After filling in the required information (described below), the user hits <ESC> to accept the label and start a new one. After the last label has been accepted, the user then hists the <ESC> one more time (on an empty label screen) to exit the module and save the labels.

SCREEN LAYOUT

The screen layout for the labels looks like this:

```
_____________________________________________________________________
PAINT LABELS: labelfile              new labels                   [1]

TEXT: ____________________________________________________     SKIP: no__
      ____________________________________________________
      ____________________________________________________
      ____________________________________________________


LOCATION:  EAST: ____________________  OFFSET: _________
          NORTH: ____________________  OFFSET: _________
      PLACEMENT: center_____________

FONT:              standard___________
TEXT SIZE:         500______
TEXT COLOR:        black______________  WIDTH: 1________
HIGHLIGHT COLOR:   none_______________  WIDTH: 0________

BACKGROUND COLOR:  white______________
BORDER COLOR:      black______________
OPAQUE TO VECTORS: yes______
```

```
              AFTER COMPLETING ALL ANSWERS, HIT <ESC> TO CONTINUE
                            (OR <Ctrl-C> TO CANCEL)
-------------------------------------------------------------------------
```

The label information that must be provided is:

TEXT:

Up to four lines of text. Lines in multiple line labels will appear one above the next.

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 on the image. The user specifies the easting and northing, and (optionally) specifies a vertical and horizontal offset (in printer pixels) from the specified easting/northing. (The vertical offset will shift the location to the south if positive, north if negative. The horizontal offset will shift the location east if positive, west if negative.) These offsets are provided to allow finer placement of labels.

PLACEMENT:

Determines which part of the label to which the location refers. If placement is unspecified, the label is centered (*center*), by default. Label placement 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)
```

FONT:

This specifies the font to use. The following fonts are available:
cyrilc gothgbt gothgrt gothitt greekc greekcs greekp greeks italicc italiccs italict romanc romancs romand romans romant scriptc scripts

The word *standard* can be used to specify the default font (which is *romans*).

TEXT SIZE:

This determines the size of the letters. The size specifies the vertical height of the letters in meters on the ground. Thus text will grow or shrink depending on the scale at which the map is drawn. The default text size, if none is specified, is *500*.

TEXT COLOR:

This selects the text color. If unspecified, the label's text is drawn in *black*, by default. The text color can be specified in one of four 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*.)
3. By printer color number to get the exact printer color.
   (This form is not supported by *d.paint.labels*.)
4. Specify *none* to suppress the lettering.

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. *1* is also the default width to which the width is set by *paint*, if none is specified by the user.)

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. No highlight color (*none*) is used, by default, if unspecified by the user. To specify use of no highlight color, specify *none*. (See TEXT COLOR above for a list of permissible color names.)

HIGHLIGHT WIDTH:

Specifies how far from the text lines (in units of pixels) the highlight color should extend. The default highlight width is set to *0* (i.e., no highlight color).

BACKGROUND COLOR:

Text may be boxed in a solid color by specifying a background color. Specify *none* for no background. The default background color setting, if unspecified by the user, is *white*. (See TEXT COLOR above for a list of permissible color names.)

BORDER COLOR:

Select a color for the border around the background. Specify *none* to suppress the border. The default border color used, if unspecified, is *black*. (See TEXT COLOR above for a list of permissible color names.)

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. The default setting, if unspecified by the user, is *yes*.

SEE ALSO

p.map
p.select
p.icons
d.paint.labels

AUTHOR

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

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

NAME

p.map − Hardcopy color map output utility.
(GRASS Hardcopy Output Program)

SYNOPSIS

p.map
p.map help
p.map [**input**=*name*] [**scale**=*mapscale*]

DESCRIPTION

p.map produces hardcopy color map products on your system's color output device. Output can include a raster map, any number of vector overlays, site data, text labels, and other spatial data.

This program has 2 distincts modes of operation. The command−line mode requires the user to prepare a file of mapping instructions describing the various spatial and textual information to be printed prior to running *p.map*. The interactive mode (i.e., no command−line arguments) will prompt the user for items to be mapped and does not require the user to prepare a file of instructions.

The command−line parameters are:

input=*name*
: File containing mapping instructions. (or enter **input**=− to enter from keyboard). These instructions are described in detail below.

scale=*mapscale*
: Scale of the output map, e.g. 1:25000
Default: 1panel
This parameter is provided as a convenience. It is identical to the *scale* mapping instruction described below.

Note: the user must select an output device using *p.select* before running *p.map*. Also, the *preview* device can be selected to view the output from *p.map* on the graphics monitor instead of sending it to a paper printer.

MAPPING INSTRUCTIONS

The mapping instructions allow the user to specify various spatial data to be plotted. These instructions are normally prepared in a regular text file using a system editor. Some instructions are single line instructions while others are multiple line. Multiple line instructions consist of the main instruction followed by a subsection of one or more additional instructions.

colormode

Selects the appropriate method to color raster map layers and images.

```
USAGE:  colormode approx | best
```

There are two methods: *approximate* and *best.* From a user perspective, *approximate* can be used for raster map layers with few categories, such as soils, and *best* should be used for images like LANDSAT images or NHAP photos, or maps with many categories. The *approximate* mode treats each pixel independently, giving it the printer color that best approximates the true color. The *best* mode "blends" colors from pixel to pixel using a dithering technique to simulate more colors than the printer can actually print. The default, if unspecified, is *best.*

This example would select the *approximate* colormode. The assumption is that the raster map layer being printed is has few colors or that the colors would not look good dithered.

```
EXAMPLE:
        colormode approx
```

colortable

Includes the color table for the raster map layer in the legend below the map

```
USAGE:  colortable [y|n]
```

The color table will display the colors for each raster map layer category value and the category label. To get a color table, you must have previously requested a raster map layer. Omitting the **colortable** instruction would result in no color table. **Note**: Be careful about asking for color tables for raster map layers which have many categories, such as elevation. This could result in the printing of an extremely long color table!!!!!

This example would print a color table as part of the legend to the map.

```
EXAMPLE:
        colortable y
```

comments

Prints comments beneath the map . You may submit comment text line by line during *p.map* execution or a via a prepared comments file.

```
USAGE:  comments [commentfile]
        comments
        end
```

This example prints the comment "This is a comment" in the legend below the map.

```
EXAMPLE:
```

```
        comments
        This is a comment.
        end
```

This example prints whatever is in the file *veg.comments* in the legend below the map.

```
EXAMPLE:
        raster vegetation
        comments veg.comments
        end
```

Presumably, the file *veg.comments* contain comments pertaining to the raster map layer *vegetation*, such as "This map was created by classifying a LANDSAT TM image".

defpat

Defines area fill patterns to be used in the *setpat* instruction.

```
USAGE:  defpat name
        pattern
        color # color
        end
```

The pattern is given a name on the *defpat* instruction line. The pattern which follows is composed of a sequence of numbers 0−9 (and blanks, which are equivalent to 0). The blanks or zeros indicate holes in the pattern where the normal category color would show thru. The other number 1−9 indicate pattern pixels and can be assigned any color. The default color for all the digits will be black unless specified with the *color* instruction. The color option will begin by entering the word *color* followed by one of the digits (1−9) in the pattern, followed by one of the NAMED COLORS. This should be repeated for each of the digits specified to avoid using black. The instruction *end* terminates the pattern definition. Of course, the user can define more patterns by entering more *defpat* instructions.

NOTE: Do NOT indent the pattern. Leading blank spaces will be interpreted as 0's.

This example creates a black horizontal line pattern.

```
EXAMPLE:
        defpat horiz
        1
        0
        0
        0
        color 1 black
        end
```

This example creates a green vertical line pattern.

```
EXAMPLE:
        defpat vert
        1000
        color 1 green
```

```
        end
```

This example creates a red diagonal line pattern.

```
EXAMPLE:
        defpat diag
        00001
        0001
        001
        01
        1
        color 1 red
        end
```

This example creates a two−toned tree pattern with orange trunks and green leaves.

```
EXAMPLE:
        defpat tree
             2
            222
           22122
          22 1 22
             1



                      2
                     222
                    22122
                   22 1 22
                      1
        color 1 orange
        color 2 black
        end
```

endpanel

Specifies which panel number to end printing. The default is 0, and will print all panels from the startpanel to the last panel.

```
EXAMPLE:
        endpanel 4
```

This example would end output at panel 4.

grid

Overlays a coordinate grid onto the output map.

```
USAGE:  grid spacing
        color color
        numbers # [color]
        end
```

The **spacing** of the grid is given (in the geographic coordinate system units) on the main instruction line. The subsection instructions allow the user to specify the **color** of the grid lines, whether coordinate **numbers** should appear on the grid lines, and if they should appear every grid line (1), every other grid line (2), etc., and what color the numbers should be. The defaults are black grid lines, unnumbered.

This example would overlay a green grid with a spacing of 10000 meters (for a metered database, like UTM) onto the output map. Alternate grid lines would be numbered with red numbers.

```
EXAMPLE:
        grid 10000
        color green
        numbers 2 red
        end
```

labels

Selects a labels file for output (see manual entry for *p.labels*).

```
USAGE:  labels  labelfile|list
```

This example would paint labels from the labels file called *town.names*. Presumably, these labels would indicate the names of towns on the map.

```
EXAMPLE:
        labels town.names
```

line

Draws lines on the output map.

```
USAGE:  line east north east north
        line x% y% x% y%
        color color
        width #
        masked [y|n]
        end
```

The beginning and ending points of the line are entered on the main instruction. These points can be defined either by map coordinates or by using percentages of the geographic region. The user may also specify line **color**, **width** in pixels, and if the line is to be **masked** by the current mask. (See manual entry for *r.mask* for more information on the mask.)

This example would draw a yellow line from the point x=10% y=80% to the point x=30% y=70%. This line would be 2 pixels wide and would appear even if there is a mask.

```
EXAMPLE:
        line 10% 80% 30% 70%
        color yellow
        width 2
        masked n
```

```
        end
```

Of course, multiple lines may be drawn with multiple *line* instructions.

outline

Outlines the areas of a raster map layer with a specified color.

```
USAGE:  outline
        color  color
        end
```

Distinct areas of the raster map will be separated from each other visually by drawing a border (or outline) in the specified **color** (default: black). Note: it is important the user enter the instruction **end** even if a color is not chosen. (It is hoped that in the future the outline of a different raster map layer other than the one currently being painted may be placed on the map.)

This example would outline the category areas of the *soils* raster map layer in grey.

```
EXAMPLE:
        raster soils
        outline
        color grey
        end
```

point

Places additional points or icons on the output map.

```
USAGE:  point east north
        point x% y%
        color color
        icon iconfile|list
        size #
        masked [y|n]
        end
```

The point location is entered in the main instruction line by giving either the map coordinates or by using percentages of the geographic region. The user may also specify the point **color**, the **icon** file to be used to represent the point location (see the the manual entry for *p.icons*), the **size** of the icon in integer multiples of the pattern in the icon file, and whether the point is to be **masked** by the current mask. (See manual entry for *r.mask* for more information on the mask.)

This example would place a purple diamond (from icon file *diamond*) at the point (E456000 N7890000). This diamond would be the same size it is in the diamond icon file and would not be masked by the current mask.

```
EXAMPLE:
        point 456000 7890000
        color purple
        icon diamond
```

```
        size 1
        masked n
        end
```

Of course, multiple points may be drawn with multiple *point* instructions.

raster

Selects a raster map layer for output.

```
USAGE:  raster mapname|list
```

For each *p.map* run, only one raster map layer can be requested. If no raster map layer is requested, a completely white map will be produced. It can be useful to select no raster map layer in order to provide a white background for vector images.

This example would paint a map of the raster map layer *soils*.

```
EXAMPLE:
        raster soils
```

read

Provides *p.map* with a previously prepared input stream.

```
USAGE:  read previously prepared UNIX file
```

Mapping instructions can be placed into a file and read into *p.map.*

Note:*p.map* will not search for this file. The user must be in the correct directory or specify the full path on the **read** instruction. (Note to /bin/csh users: ~ won't work with this instruction).

This example reads the UNIX file *pmap.roads* into *p.map*. This file may contain all the *p.map* instructions for placing the vector map layer *roads* onto the output map.

```
EXAMPLE:
        read pmap.roads
```

The user may have created this file because this vector map layer is particularly useful for many *p.map* outputs. By using the **read** option, the user need not enter all the input for the **vector** instruction, but simply **read** the previously prepared file with the correct instructions.

region

Places the outline of a smaller geographic region on the output.

```
USAGE:  region regionfile | list
        color color
        width #
        end
```

Geographic region settings are created and saved using *g.region*. The *p.map region* option can be used to

show an outline of a smaller region which was printed on a separate run of *p.map* on other user−created maps.

The user can specify the **color** and the **width** (in pixel units) of the outline. The default is a black border of one pixel width.

This example would place a white outline, 2 pixels wide, of the geographic region called *fire.zones* onto the output map. This geographic region would have been created and saved using *g.region*.

```
EXAMPLE:
        region fire.zones
        color white
        width 2
        end
```

scale

Selects a scale for the output map.

USAGE: **scale** *scale*

The scale can be selected either as:

a relative ratio, e.g. 1:25000;
an absolute width of the printed map, e.g. 10 inches;
the number of printed paper panels, e.g. 3 panels;
the number of miles per inch, e.g. 1 inch equals 4 miles.

This example would set the scale of the map to 1 unit = 25000 units.

```
EXAMPLE:
        scale 1:25000
```

setcolor

Overrides the color assigned to one or more categories of the raster map layer.

USAGE: **setcolor** cat(s) color

This example would set the color for categories 2,5 and 8 of the raster map layer *watersheds* to white and category 10 to green. (**NOTE**: no spaces are inserted between the category values.)

```
EXAMPLE:
        raster watersheds
        setcolor 2,5,8 white
        setcolor 10 green
```

Of course, *setcolor* can be requested more than once to override the default color for additional categories. More than one category can be changed for each request by listing all the category values separated by commas (but with no spaces).

setpat

Assigns a (previously defined) pattern on a raster map layer category.

```
USAGE:  setpat cat name
  or    setpat builtin
  or    setpat all
```

The user can choose to use: the name of a specific pattern created using **defpat** (see above); the patterns built into *p.map*; or all the patterns the user may have created.

This example assigns the vertical pattern created using **defpat** (see example in **defpat** above) to category 3 of the raster map layer *vegetation* and the tree pattern (see example in **defpat** above) to category 10.

```
EXAMPLE:
        raster veg
        setpat 3 vert
        setpat 10 tree
```

This example reads a previously prepared UNIX file *horiz.pat* with the correct **defpat** instructions for creating a black horizontal pattern (see example in **defpat** above) and assigns that pattern to category 5 of the raster map layer *soils* via the **setpat** instruction.

```
EXAMPLE:
        raster soils
        read horiz.pat
        setpat 5 horiz
```

To select the builtin patterns:

```
EXAMPLE:
        raster soils
        setpat builtin
```

To select individual builtin patterns:

```
EXAMPLE:
        raster soils
        setpat 5 #1
        setpat 10 #2
```

startpanel

Specifies at which panel number to begin printing. Default is 0 and would start printing from the first panel.

```
EXAMPLE:
        startpanel 2
```

This example would begin printing at panel 2.

sites

Selects sites data to be placed on the output map (see manual entry for *s.menu*).

```
USAGE:  sites sitemap |list
        color color
        icon iconfile |list
        size #
        desc [y|n]
        end
```

The user may specify the the **color** of the sites (see section on NAMED COLORS below); the **icon** to be used to represent the presence of a site (see the manual entry for *p.icons*); the **size** of the icon (number of times larger than the size it is in the icon file); and whether or not the **desc**ription associated with each site is also to be printed.

This example would paint a sites map with blue windmills (from an icon file created by the user using the *p.icons* GRASS command) placed at all windmill locations (from a sites list). These windmills would be two times larger than the size of the icon in the icon file and have descriptions from the sites list file printed beside them.

```
EXAMPLE:
        sites windmills
        color blue
        icon windmill
        size 2
        desc y
        end
```

text

Places text on the map.

```
USAGE:  text  east north text
        text  x% y% text
        font fontname
        color color|none
        width #
        hcolor color|none
        hwidth #
        background color|none
        border color|none
        size #
        ref reference point
        xoffset #
        yoffset #
        opaque [y|n]
        end
```

The user specifies where the text will be placed by providing map coordinates or percentages of the geographic region map. The text follows these coordinates on the same instruction line. More than one line of text can be specified by notating the end of a line with **\en** (e.g. USA\|\enCERL).

The user can then specify various text features:

font: cyrilc gothgbt gothgrt gothitt greekc greekcs greekp greeks italicc italiccs italict romanc romancs romand romans romant scriptc scripts (The default font is romans);

color (see NAMED COLORS);

width of the lines used to draw the text (to make thicker letters);

size as the vertical height of the letters in meters on the ground (text size will grow or shrink depending on the scale at which the map is painted);

the highlight color (**hcolor**) and the width of the highlight color (**hwidth**);

the text−enclosing−box **background** color; the text box **border** color;

ref. This reference point specifies the text handle − what part of the text should be placed on the location specified by the map coordinates. Reference points can refer to: [lower|upper|center]\ [left|right|center] of the text to be printed;

yoffset, which provides finer placement of text by shifting the text a vertical distance in pixels from the specified north. The vertical offset will shift the location to the south if positive, north if negative;

xoffset, which shifts the text a horizontal distance in pixels from the specified east The horizontal offset will shift the location east if positive, west if negative;

whether or not the text should be **opaque** to vectors. Entering **no** to the opaque option will allow the user to see any vectors which go through the text's background box. Otherwise, they will end at the box's edge.

This example would place the text *SPEARFISH LAND COVER* at the coordinates E650000 N7365000. The text would be a total of 3 pixels wide (2 pixels of red text and 1 pixel black highlight), have a white background enclosed in a red box, and be 500 meters in size. The lower right corner of the text would be centered over the coordinates provided. All vectors on the map would stop at the border of this text.

```
EXAMPLE:
        text 650000 7365000 SPEARFISH LAND COVER
        font romand
        color red
        width 2
        hcolor black
        hwidth 1
        background white
        border red
        size 500
        ref lower left
        opaque y
        end
```

vector

Selects a vector map layer for output.

```
USAGE:  vector vectormap|list
        color [#] color
        width #
        hcolor color
        hwidth #
        masked [y|n]
        style  0−9
        end
```

The user can specify the **color** of the vectors; the **width** of the vectors lines in pixels; the highlight color (**hcolor**) for the vector lines; the width of the highlight color (**hwidth**) in pixels; whether or not the raster map layer is to be **masked** by the current mask (see manual entry *r.mask* for more information on the mask); and the line **style**. The line style allows the vectors to be dashed in different patterns and colors. This is done by typing a series of numbers (0−9) in a desired sequence or pattern. Colors for the numbers (1−9) can be assigned using the **color** instruction. Blanks and non−digit characters are recognized as 0's. Using 0 would allow the colors of the raster map layer (or the background color if no raster map layer was selected) to show through.

This example would paint a map of the vector map layer named *streams*. These streams would be a total of 3 pixels wide (the inner two pixels blue and the outer highlight pixel white). The map would not show streams outside of the current mask.

```
EXAMPLE:
        vector streams
        color blue
        width 2
        hcolor white
        hwidth 1
        masked y
        end
```

This example would paint a map of the vector map layer *roads*. These roads would be 2 pixels wide and would be dashed blank−black−red (the blank areas would show what lies under the roads). This map would show roads inside and outside of the current mask.

```
EXAMPLE:
        vector roads
        width 2
        style 001122
        color 1 black
        color 2 red
        masked n
        end
```

verbose

Changes the amount of talking *p.map* will do.

```
USAGE: verbose [0|1|2]
```

A higher value implies more chatter. The default is 2. This example sets the amount of chatter to a minimum.

```
EXAMPLE:
        verbose 0
```

end

Terminates input and begin painting the map.

```
USAGE:  end
```

NAMED COLORS

The following are the colors that are accepted by *p.map*:

```
aqua
black
blue
brown
cyan
gray
green
grey
indigo
magenta
orange
purple
red
violet
white
yellow
```

ICONS VS. PATTERNS

Icons and patterns as used in *p.map* are not the same thing. Patterns are defined and are normally used to cover those extended areas covered by a raster map layer category. A pattern will repeat above, below and adjacent to itself. Icons are used to represent a single point.

Patterns are supported directly within *p.map* using the *defpat* instruction, while icons must be created using the *p.icons* program.

EXAMPLE p.map INPUT FILE

The following is an example of a *p.map* script file. The file has been name *spear.soils*. For the purposes of illustration, the file is in two columns. This script file can be entered at the command line:

p.map input=*spear.soils*

```
        (cont.)
raster soils
vector streams
   color blue
   width 2
   hcolor white
   hwidth 1
   masked y
   end
vector roads
   width 2
   style 001122
   color 1 black
   color 2 red
   masked n
   end
labels town.names
region subregion
   color white
```

```
defpat diag
000001
00001
0001
001
01
1
   color 1 red
   end
setpat 4 diag
text 608000.00 3476004.73 SPEARFISH SOILS MAP
   color red
   width 2
   hcolor black
   hwidth 1
   background white
   border red
   size 500
```

```
   width 2
   end
grid 10000
   color green
   numbers 2 red
   end
outline
   color black
   end
colortable y
comments
   This is a comment
   end
scale 1:25000
setcolor 6,8,9 white
setcolor 10 green
   ref lower left
   opaque y
   end
line 606969.73 3423092.91 616969.73 3423092.91
   color yellow
   width 2
   opaque yes
   end
point 40% 60%
   color purple
   icon diamond
   size 2
   masked n
   end
end
```

INTERACTIVE MODE

If the user simply enter *p.map* without arguments, then a simple prompting session occurs. Some, but not all of the non−interactive requests are available at this level.

SEE ALSO

p.chart
p.icons
p.labels
p.select

AUTHOR

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

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

NAME

p.map.new − Color map output utility.
(GRASS Hardcopy Output Program)

SYNOPSIS

p.map.new
p.map.new [**input**= *name*] [**scale**= *mapscale*]

DESCRIPTION

The *p.map.new* command produces color maps for output on a color hardcopy device or a graphics monitor. Output can include a raster map, any number of vector overlays, site data, text labels, and other map elements.

This command has three modes of operation. The command−line mode requires a previously prepared file of mapping instructions describing the map elements to be printed. The interactive mode (i.e., no command−line arguments) will prompt the user for items to be mapped and does not require the user to prepare a file of instructions. The keyboard mode is started by entering a hyphen (−) for the **input** parameter. The *p.map.new* instructions would then be entered via the keyboard.

The command−line parameters are:

input=*name*
: File containing mapping instructions (or enter **input**=− to enter instructions from the keyboard). These instructions are described in detail below.

scale=*mapscale*
: Scale of the output map, e.g. 1:25000
Default: 1 panel
The options for this parameter are identical to the *scale* mapping instruction described below. If a *scale* instruction is present in an input file, it is superseded by the command−line **scale** parameter.

An output device can be selected using *p.select* before running *p.map.new*. Valid devices include on−line hardcopy devices, plus *preview*, *preview2*, and *ppm*. See manual entry for *p.select*.

The current geographic region determines the area that is mapped using *p.map.new*

NON−INTERACTIVE MAPPING INSTRUCTIONS

Mapping instructions allow the user to specify various map elements to be plotted. These instructions are normally prepared in an ASCII text file using a system editor. All of the listed mapping instructions are usable in a prepared file in the command−line mode. Not all of them are available in the interactive and keyboard

modes.

Some instructions are single line instructions while others are multiple line. Multiple−line instructions consist of the main instruction followed by a subsection of one or more additional instructions. All multiple−line instructions must be completed by the **end** terminator.

Some instructions, such as those using data layers, icons, or labels, access files via the current mapset search path.

barscale

Places a barscale on the output map.

```
USAGE:  barscale  east north
        barscale  x% y%
        unit  ft| mi| m| km
        length  #
        interval  #
        style  dash| tick
        width  #
        color  color
        textsize  #
        textcolor  color| none
        textfont  font
        background  color| none
        border  color| none
        end
```

The location of the zero point of the scale bar is entered on the first instruction line. The location can be defined either by map coordinates or by percentages of the map area, where 0% 0% is the lower left corner of the map.

The user specifies the barscale **unit** of measurement, the total **length** using that unit, and the length of one **interval** (a smaller length evenly divisible into the total length).

The **style** of the scale bar can be specified. The *dash* style has solid lines representing each interval, separated by gaps. The *tick* style has a solid total length with vertical ticks marking each interval.

The user can also specify the **width** of the bar in pixels, its **color** (see VALID COLORS NAMES), the **textcolor**, **textsize** in geographic units, **textfont** (see VALID FONT NAMES), **background** color, and **border** color.

The **barscale** instruction set must be completed with the **end** terminator.

This example would result in a scale bar representing two kilometers. Vertical ticks would be placed at the scale origin, the mid−point, and at the end. The black bar and its accompanying black text would overlay a white box trimmed be a red border.

```
EXAMPLE:
        barscale  605000 4915000
        unit  km
        length  2
        interval  1
```

```
        style  tick
        width  2
        color  black
        textcolor  black
        textsize  150
        background  white
        border  red
        end
```

colormode

Selects the method to portray the colors of the raster map layer or image.

USAGE: **colormode** *approx*| *best*

There are two options for **colormode**: *approx* and *best*. The *approx* option should be used for raster map layers with few categories, and *best* should be used for images like LANDSAT images or NHAP photos, or maps with very many categories. The *approx* mode treats each pixel independently, giving it the printer color that best approximates the true color. The *best* mode "blends" colors from pixel to pixel using a dithering technique to simulate more colors than the printer can actually print. If unspecified, the default is *best.*

This example would select the *approx* colormode. The assumption is that the raster map layer being printed has few colors or that the colors would not look good dithered.

EXAMPLE:

```
        colormode  approx
```

colortable

Includes the color table for the raster map layer in the area below the map on hardcopy output.

USAGE: **colortable** [*y*| *n*]

The color table will display the colors for each raster map layer category and the category value. The **colortable** instruction can not precede the **raster** instruction in the *p.map.new* input. The color table is not shown when the output device is the color monitor.

The user should be careful about asking for color tables for raster map layers that have very many categories, such as an elevation layer. This could result in the printing of an extremely long and generally useless color table!

This example would print a color table below the data area of the map.

EXAMPLE:

```
        colortable  y
```

comments

Prints comments beneath the map on hardcopy output.

```
USAGE:  comments  [commentfile]
        comments
        end
```

Comment text can be entered in the *p.map.new* file or from a separate, previously prepared file. Comments are not shown when the output device is the color monitor. The **comments** instruction set must be completed by the **end** terminator.

This example prints the comment "This is a comment" below the data area on the the map.

```
EXAMPLE:
        comments
        This is a comment.
        end
```

This example prints the text in a file called "veg.comments" in the current directory.

```
EXAMPLE:
        raster  vegetation
        comments  veg.comments
        end
```

defpat

Defines an area fill pattern to be used in **setpat** instructions.

```
USAGE:  defpat  name
        pattern
        color  #  color
        end
```

An area fill pattern is given a name on the **defpat** instruction line. This name can then be used in subsequent *setpat* instructions. The **defpat** instruction can be used more than once to specify multiple patterns.

The specified pattern is composed of a sequence of numbers (0−9, and blanks, which are equivalent to 0) on one or more lines. The zeros and blanks indicate areas in the pattern where the normal category colors are visible. The other digits, 1−9, indicate pattern pixels and can be assigned any valid color.

The **color** option specifies a non−zero digit in the pattern, followed by a valid color name. It can be repeated for each of the non−zero digits in the pattern. The default color for all non−zero digits is black unless specified with the **color** option.

The **defpat** instruction set must be completed by the **end** terminator.

In the *p.map.new* input, the **defpat** instruction must precede any **setpat** instruction using the specified pattern.

Note: Indented pattern specifications will be interpreted as having leading blanks.

This example creates a black horizontal line pattern called "horiz". Each black line in the pattern would be one pixel wide and would be three pixels from neighboring lines.

```
EXAMPLE:
        defpat  horiz
        1
        0
        0
        0
        color  1 black
```

```
        end
```

This example creates a green vertical line pattern.

```
EXAMPLE:
        defpat  vert
        1000
        color  1 green
        end
```

The following example creates a red diagonal line pattern.

```
EXAMPLE:
        defpat  diag
        00001
        0001
        001
        01
        1
        color  1 red
        end
```

This example creates a two−toned tree pattern with orange "trunks" and green "leaves".

```
EXAMPLE:
        defpat  tree
             2
            222
           22122
          22 1 22
              1


                       2
                      222
                     22122
                    22 1 22
                        1
        color  1 orange
        color  2 black
        end
```

end

Terminates input and begins the painting of the map to the output device.

```
USAGE:  end
```

An **end** instruction completes the entire input to *p.map.new*. It is normally the last line in an input file, but it can be moved forward to eliminate any instructions following its position.

The **end** instruction for the entire input should not be confused with **end** terminators that are required with all multiple−line instruction sets.

endpanel

Specifies which panel number to end printing. The default is 0, and will print all panels from the startpanel to the last panel.

```
EXAMPLE:
        endpanel 4
```

This example would end output at panel 4.

grid

Overlays a coordinate grid on the output map.

```
USAGE:  grid  spacing
        pattern  notick| tick # #
        masked  [data| nodata| all]
        style  sequence
        width  #
        color  color
        numbers  #  [color] [in| out]
        numbersbg  color| none
        numbersize  #
        end
```

The spacing of the grid in geographic coordinate system units must be be specified on the first instruction line. The user can specify the overall look of the grid using the **pattern** parameter. The *notick* option is for a complete net of intersecting lines. The *tick* option is for smaller tick marks where grid intervals intersect. The horizontal and vertical lengths, in pixels, must be specified with the *tick* option.

The user can control the areas covered by the grid by using the **masked** parameter. With the *data* option of **masked**, the grid will be seen over all areas of the map's raster layer except the no−data (category 0) areas. With the *nodata* option, the grid will be seen only over the no−data areas. The entire grid is seen with the *all* option to **masked**.

The grid line **style** can be specified using a series of 1's and 0's. The 1's represent the visible dashes and the 0's represent gaps between the dashes. The default is solid lines. The **width** (in pixels) and **color** of the grid lines can also be specified.

The user can control the placement and look of grid label numbers using the **numbers**, **numbersbg**, and **numbersize** parameters. The **numbers** parameter is used to include grid labels, to specify which labels should be shown (where 1 is every grid label, 2 is every other grid label, etc.), and to specify the label color. It is also used to place the labels inside or outside the current region. The background color behind each label is specified by the **numbersbg** parameter. The user controls the grid label size using the **numbersize** parameter, in geographic units.

The **grid** instruction set must be concluded by the **end** parameter.

When used in a metric location, this example would produce grid ticks every 5000 meters. The purple ticks would have "arms" ten pixels long and would be visible over the entire map area. The purple grid numbers would be 350 meters high (to scale), inside the current region map area, and have no background color. A grid label would appear every 5000 meters.

```
EXAMPLE:
        grid  5000
        pattern  tick 10 10
        masked  all
        width  1
        color  purple
        numbers  1 purple in
        numbersbg  none
        numbersize  350
        end
```

The following example would produce black grid lines every 1000 meters. The lines would be visible only in the areas of category 0, and they would be dashed, with one long dash for every short gap. Every other grid label would be shown, each with a white background.

```
EXAMPLE:
        grid  1000
        pattern  notick
        masked  nodata
        style  11111100
        width  1
        color  black
        numbers  2 black in
        numbersbg  white
        numbersize  200
        end
```

labels

Selects a labels file for output.

```
USAGE:  labels  labelfile| list
```

The **labels** instruction includes previously prepared label specifications. See manual entry for *p.labels* for correct format of the labels file. The labels file must be accessible via the current mapset search path. The *list* option is available in keyboard mode.

This example would paint labels from a labels file called *town.names*.

```
EXAMPLE:
        labels  town.names
```

legend

Places a user−designed map legend on the output.

```
USAGE:  legend  east north
        legend  x% y%
        height  #
        width  #
        vlen  #
        textcolor  color
        textsize  #
        textwidth  #
        xspace  #
        yspace  #
        background  color
```

```
border  color
beginrast
ramp  value| label vertical| horizontal
catnum  cat description
end
beginvect
vectname  vectormap description
vectTITLE  vectormap
end
beginsite
sitename  sitemap description
end
end
```

The location of the upper left corner of the legend must be entered on the first instruction line. The location can be defined either by map coordinates or by percentages of the map area, where 0% 0% is the lower left corner of the map.

The user specifies the **height** and **width** of the boxes that will show raster category colors and/or patterns. The length of line segments that will show vector line colors and patterns is specified with the **vlen** parameter. The user controls horizontal spacing between legend symbols and legend text using **xspace**. The **yspace** parameter is used to control vertical spacing between legend symbols. All of these measurements are in pixels.

The user designs the legend text using the **textcolor**, **textsize**, and **textwidth** parameters. Colors are listed in the VALID COLOR NAMES section of this manual entry. The **textsize** is specified in pixels. The **textwidth** is also specified in pixels.

The user can specify the color for the **background** box containing the entire legend. If a color is chosen, underlying map elements are opaqued. The user can also specify a **border** color for the legend box.

The user specifies the symbols to be included in the legend using the **beginrast**, **beginvect**, and **beginsite** parameters. Each of these parameters starts a subsection of the **legend** instruction that must be completed by an **end** terminator. These should not be confused with the **end** terminator for the entire **legend** instruction set.

If the user simply uses the **beginrast** parameter followed by **end**, all categories of the map's raster layer will be shown in individual boxes, and the legend labels will be the corresponding category names in the layer's *cats* file. The user can include specific categories and optional labels by using one or more **catnum** lines, each including a category number and the accompanying legend text. If the map's raster layer portrays a continuous range of data, a **ramp** in the legend might be appropriate. The **ramp** can be vertical or horizontal, and it's accompanying text can be either the smallest and largest category values, or the *cats* labels associated with the smallest and largest categories. The **beginrast** subsection must be completed with an **end** terminator.

Symbols for vector data on the map can be included in the legend by using the **beginvect** parameter. If the user simply follows **beginvect** with **end**, all vector layers in the map will be included in the legend. The user can include specific vector layers in the legend by using the **vectname** line one or more times, each including a vector layer name and an accompanying description. The vector layer TITLEs as written in *dig_cats* files can be included as the legend text by using the **vectTITLE** line one or more times. The **beginvect** subsection must be completed with an **end** terminator.

Site symbols are included in the legend by using one or more **sitename** lines in the **beginsite** subsection. Each line includes the name of the site list and an accompanying description. The **beginsite** subsection must be completed with an **end** terminator.

The entire **legend** instruction set must be completed by an **end** terminator.

In the *ps.map.new* input, the **legend** instruction can not precede the instructions for any of the map elements that are to be shown in the legend.

This example would produce a legend with five symbols: a point symbol, the colors and patterns for three raster categories, and a line representing one vector layer, in that order. The background of the legend would be white and surrounded by a red border. All text in the legend would be black.

```
EXAMPLE:
        legend  589000 4921200
        height  10
        width  20
        vlen  20
        xspace  10
        yspace  7
        textcolor  black
        textsize  250
        textwidth  1
        background  white
        border  red
        beginsite
        sitename  archsites Arch. site
        end
        beginrast
        catnum  4 Sandstone
        catnum  5 Limestone
        catnum  6 Shale
        end
        beginvect
        vectname  roads Road
        end
        end
```

The following example would produce a legend with a vertical ramp showing all the colors in the map's raster layer. The labels of the first and last categories would be included.

```
EXAMPLE:
        legend  589000 4921200
        height  10
        width  20
        xspace  10
        textcolor  black
        textsize  250
        textwidth  1
        background  gray
        border  black
        beginrast
        ramp  label vertical
        end
        end
```

line

Draws a line that is independent of any vector map layer on the output map.

```
USAGE:  line  east north east north
        line  x% y% x% y%
```

```
        style  sequence
        color  [# ] color
        width  #
        hcolor  color
        hwidth  #
        masked  [y| n]
        end
```

The beginning and ending points of the line are entered on the main instruction line. These points can be defined either by map coordinates or by using percentages of the geographic region, where 0% 0% is the lower left corner of the map.

The default line style is a continuous, solid line, but the user can specify a dashed line using the **style** parameter. The **style** parameter can contain a sequence of digits (0−9) that represent a colored pattern on the desired line. Colors can be assigned to each non−zero digit by using the **color** parameter multiple times. If the **color** parameter is used without a specified digit, the named color will be assigned to the entire line. Colors are listed in the VALID COLOR NAMES section in this manual entry.

The user can specify line **width** in pixels. A highlight color can be assigned with **hcolor**, and the highlight's width in pixels can be assigned with **hwidth**. The user can also specify if the line is to be **masked** by the current mask. (See manual entry for *r.mask*for more information on the mask.)

The **line** instruction set must be completed by an **end** terminator.

The **line** instruction can be used more than once to create multiple lines.

This example would draw a blue line from the point x= 10% y= 80% to the point x= 30% y= 70%. The line would be two pixels wide and would appear even if there is a mask.

```
EXAMPLE:
        line  10% 80% 30% 70%
        color  blue
        width  2
        masked  n
        end
```

The following example would draw a line with yellow dashes on a black background.

```
EXAMPLE:
        line  605000 4915000 595300 4918200
        style  1111100
        color  1 yellow
        width  1
        hcolor  black
        hwidth  1
        end
```

outline

Outlines areas of a raster map layer with a specified color.

```
USAGE:  outline
        color  color
        end
```

The **outline** instruction can be used to place a border around all contiguous groups of same−value cells in a raster map layer. A valid color name can be specified with the optional **color** parameter. The default color is black.

The **outline** instruction set must be completed by the **end** terminator, even if the **color** parameter is not used.

The **outline** instruction can not precede a **raster** instruction in a *p.map.new* input file.

The instruction sequence in this example would outline in grey the category areas of a raster map layer called "soils".

```
EXAMPLE:
        raster  soils
        outline
        color  grey
        end
```

point

Places a point symbol on the output map.

```
USAGE:  point  east north
        point  x% y%
        icon  iconfile| list
        color  color
        size  #
        masked  [y| n]
        end
```

The user enters a point symbol location on the main instruction line. The location can be defined either by map coordinates or by percentages of the map area, where 0% 0% is the lower left corner of the map.

The icon to be used can be specified with the **icon** parameter. The user can use any icon in an *icons* directory within the current mapset search path. Icons can be created using *p.icons* or by simply using a system editor. The default icon is a diamond. The *list* option for **icon** is available in keyboard mode.

The user can specify the symbol **color**. Colors are listed in the VALID COLOR NAMES section of this manual entry.

The icon **size** is a positive, floating−point scaling factor of the pattern in the icon file. A size of 1 produces an icon with the same number of pixels (at the output device's resolution) as ASCII characters in the icon file.

The user can also specify whether the point symbol is to be **masked** by the current mask. (See manual entry for *r.mask* for more information on the mask.)

The **point** instruction set must be completed be an **end** terminator. Multiple points may be drawn with multiple **point** instructions.

This example would access an icon file called "box" within the current mapset search path. The red box symbol would be placed at the point E603000, N4921750. The box would have the same number of pixels as characters in the icon file. It would not be masked by the current mask.

```
EXAMPLE:
```

```
        point  603000 4921750
        icon  box
        color  red
        size  1
        masked  n
        end
```

raster

Selects a raster map layer for output.

```
USAGE:  raster  rastermap| list
```

Only one GRASS raster map layer can be specified in a *p.map.new* input file. If no raster map layer is requested, a white background will be produced. The *list* option is available in keyboard mode.

The raster layer must be accessible within the current mapset search path. In a *p.map.new* input file, the **raster** instruction must precede these instructions: **colortable**, **outline**, **setcolor**, and **setpat**. It also must precede any **legend** instruction set that applies to the raster map layer.

This example would paint a map of the raster map layer *soils*.

```
EXAMPLE:
        raster  soils
```

read

Provides input to *p.map.new* from a previously prepared instruction file.

```
USAGE:  read  filename
```

Mapping instructions can be placed in a file and read as input to *p.map.new*. If a certain set of mapping instructions are used in many different maps, they can be placed in one separate file and efficiently accessed by each map's instructions using the **read** instruction.

Note: *p.map.new* will not search for the file to be read. The file must be in the current directory or a full path needs to be specified on the **read** instruction line. (Note to /bin/csh users: the tilde [~] path alias will not work with this instruction).

This example reads the ASCII file "pmap.roads" into *p.map.new*.

```
EXAMPLE:
        read  pmap.roads
```

region

Places the outline of a geographic region on the output map.

```
USAGE:  region  regionfile| list
        style  sequence
        color  [# ] color
        width  #
        hcolor  color
```

```
        hwidth  #
        masked  [y| n]
        end
```

The user can place the outline of a saved geographic region on the map using **region**. The named region file must be in a *windows* directory within the current mapset search path. Geographic region settings can be created and saved using *g.region*. The *list* option is available in keyboard mode.

The default region outline style is a continuous, solid line, but the user can specify a dashed line using the **style** parameter. The **style** parameter can contain a sequence of digits (0−9) that represent a colored pattern on the desired line. Colors can be assigned to each non−zero digit by using the **color** parameter multiple times. If the **color** parameter is used without a specified digit, the named color will be assigned to the entire region outline. Colors are listed in the VALID COLOR NAMES section in this manual entry.

The user can specify the region outline **width** in pixels. A highlight color can be assigned with **hcolor**, and the highlight's width can be assigned with **hwidth**. The user can also specify if the outline is to be **masked** by the current mask. (See manual entry for *r.mask*for more information on the mask.)

The **region** instruction set must be completed by an **end** terminator.

The **region** instruction can be used more than once to show multiple regions.

This example would produce a white outline, two pixels wide, showing the geographic region called "fire.zones".

```
EXAMPLE:
        region  fire.zones
        color  white
        width  2
        end
```

scale

Specifies the scale of the hardcopy output map.

```
USAGE:  scale  scale
```

The scale of the output map can be specified in one of several different forms:

relative ratio e.g. 1:25000
number of geographic units per map unit e.g. 1 inch equals 4 miles
absolute width of the printed map e.g. 10 inches
width in number of printed panels e.g. 3 panels

Map inches can be equated with these geographic units: miles, kilometers, and meters. Valid width units are inches, centimeters, and panels. One panel is the single−sheet maximum width available on the hardcopy medium.

The final size of the hardcopy map output is determined by the combination of the specified scale and the current geographic region.

The **scale** instruction does not affect output to the *preview* device. If used, the command−line *scale* parameter overrides the **scale** instruction.

This example would set the scale of the map to one map unit represents 25,000 geographic units.

```
EXAMPLE:
        scale  1:25000
```

The following example would specify an output map that would be fifteen centimeters wide.

```
EXAMPLE:
        scale  15 centimeters
```

setcolor

Overrides the color assigned to one or more categories of the raster map layer.

```
USAGE:  setcolor  cat(s) color
```

The user can assign a desired color to categories in a raster map layer by using **setcolor**. Categories are specified on the parameter line before a valid color name. One or more category numbers can appear on the parameter line, separated by commas (with no spaces), or in ranges using hyphens.

The **setcolor** instruction can be used more than once for assignment of multiple colors.

In the input *p.map.new* file, the **setcolor** instruction must follow the **raster** instruction.

Colors are listed in the VALID COLOR NAMES section of this manual entry.

In this example, the color for raster map categories 1 through 3, plus category 5, would be set to green, categories 4, 6, and 8 would be set to blue, and category 7 would be set to red, regardless of their assigned colors in the database.

```
EXAMPLE:
        raster  watersheds
        setcolor  1−3,5 green
        setcolor  4,6,8 blue
        setcolor  7 red
```

setpat

Assigns a previously defined pattern to one or more raster map layer categories.

```
USAGE:  setpat  cat(s) name
        setpat  cat(s) #number
        setpat  all| builtin
```

The user can assign a pattern to categories in a raster map layer by using **setpat**. Categories are specified on the parameter line before the name of a pattern defined earlier using **defpat**, or before the number signifying a built−in *p.map.new* pattern. One or more category numbers can appear on the parameter line, separated by commas (with no spaces), or in ranges using hyphens.

The built−in patterns are defined in etc/paint/patterns in the compiled GRASS code directory. Each built−in pattern has an assigned number. These numbers can be used following a pound sign (#) on a **setpat** instruction line. By using the *builtin* option, each category in a raster map layer can be assigned the correspondingly numbered builtin pattern.

All raster map categories can be assigned the same defined pattern if the *all* option is used. In this case, only one pattern should be defined within the *p.map.new* mapping instruction file.

The **setpat** instruction can be used more than once for assignment of multiple patterns.

In the input *p.map.new* file, the **setpat** instruction must follow the **raster** instruction, as well as the **defpat** instruction defining the pattern that is used.

This example assigns a pattern called "vert" to categories 3 and 4 of the raster map layer "vegetation" and a pattern called "tree" to category 10.

```
EXAMPLE:
        raster  veg
        setpat  3−4 vert
        setpat  10 tree
```

This example reads a previously prepared ASCII file called *horiz.pat* containing **defpat** instructions for creating a black, horizontal pattern called "horiz", and assigns that pattern to category 5 of the raster map layer "soils".

```
EXAMPLE:
        raster  soils
        read  horiz.pat
        setpat  5 horiz
```

This example assigns built−in pattern 1 to category 1 of the "soils" raster layer, pattern 2 to category 2, and so on.

```
EXAMPLE:
        raster  soils
        setpat  builtin
```

This example assigns built−in pattern 1 to categories 5 through 7 in the "soils" raster map layer, and built−in pattern 2 to categories 10 and 12.

```
EXAMPLE:
        raster  soils
        setpat  5−7 # 1
        setpat  10,12 # 2
```

sites

Selects sites data to be placed on the output map.

```
USAGE:  sites  sitemap| list
        icon  iconfile| list
        color color
        size  #
        desc  [y| n]
        textcolor  color
```

```
        textsize  #
        end
```

GRASS sites data can be portrayed on the map using the **sites** instruction. The user can specify the point symbol to be used, and whether labels are to appear next to the symbols. The sites data must be accessible via the current mapset search path. The *list* option is available in keyboard mode.

An icon can be specified with the **icon** parameter. The user can use any icon in an *icons* directory within the current mapset search path. Icons can be created using *p.icons* or by simply using a system editor. The default icon is a diamond. The *list* option for **icon** is available in keyboard mode.

The user can specify the symbol **color**. Colors are listed in the VALID COLOR NAMES section of this manual entry.

The icon **size** is a positive, floating−point scaling factor of the pattern in the icon file. A size of 1 produces an icon with the same number of pixels (at the output device's resolution) as ASCII characters in the icon file.

The **desc** parameter is used to specify whether or not the description of each site in the *site_lists* file is also to be printed. These labels will appear directly to the right of each site symbol. The user controls the color of the labels using the **textcolor** parameters. Valid colors are listed in the named colors section of this manual entry. The label size is specified in geographic units using **textsize**.

The **sites** instruction set must be completed by the **end** terminator.

This instruction can be used more than once to portray multiple site lists.

This example would produce point symbols representing the data in a *site_lists* file called "windmills". An icon called "windmill" would be placed at each site location. These symbols would be two times larger than the size of the icon in the icon file (twice as many pixels as there are characters in the icon file). Descriptions from the sites list file would not be produced in this example.

```
EXAMPLE:
        sites  windmills
        icon  windmill
        color  blue
        size  2
        desc  n
        end
```

startpanel

Specifies at which panel number to begin printing. Default is 0 and would start printing from the first panel.

```
EXAMPLE:
        startpanel 2
```

This example would begin printing at panel 2.

text

Places text at a user−specified location on the map.

```
USAGE:  text  east north text
        text  x% y%  text
        textfont  font
        size  #
        color  color| none
        width  #
        hcolor  color| none
        hwidth  #
        ref  reference_point
        rotation  #
        xoffset  #
        yoffset  #
        background  color| none
        opaque  [y| n]
        border  color| none
        end
```

The user specifies where text will be placed by providing map coordinates or percentages of the map area, where 0% 0% is the lower left corner of the map. The text follows the locational information on the same instruction line. Multiple lines of text can be specified by notating the end of a line with **\en** (e.g., USA\|\enCERL). Leading blanks an be inserted by preceding the text string with a backslash and the blanks (e.g., text 600000 4920500\|\e See\|\enWall Drug).

The user can control the appearance of the text, its location, and the appearance of its background box.

The user can specify **textfont** (see VALID FONT NAMES in this manual entry), **size** in geographic units, **color** (see VALID COLOR NAMES), and **width** in pixels. The user can further control the text appearance by specifying a highlight color (**hcolor**) and the width of the highlight color (**hwidth**).

The text is located at the specified coordinate or percentage pair in relation to a reference point on the text string. This point, specified with the **ref** parameter, has two parts. The first part refers to a vertical location on the text string. Valid choices are *lower*, *center*, and *upper*. The second part refers to a horizontal location: *left*, *center*, and *right*.

The text string can be rotated at the reference point by using the **rotation** parameter. The value specified will be the counter−clockwise rotation in degrees from the horizontal.

The **xoffset** parameter provides finer placement of text by shifting the text a horizontal distance in pixels from the specified easting. The **xoffset** will shift the text location east if positive and west if negative. The **yoffset** parameter shifts the text a vertical distance in pixels from the specified northing. The **yoffset** will shift the location to the south if positive, north if negative.

The user can specify if a **background** box is present, and what color it should be. The user can also specify whether or not the background box is **opaque** to other map elements. The color of the **border** of this box can be specified.

This example would place the text "SPEARFISH LAND COVER" at the coordinates E650000, N7365000. The text would be a total of three pixels wide (one pixel of red text and one pixel of black on each side), have a white background enclosed in a red box, and be 500 meters in size (to scale). The lower left corner of the text would be placed at the coordinates provided. All other map elements would not be seen under the text.

```
EXAMPLE:
        text  650000 7365000 SPEARFISH LAND COVER
        textfont  romand
```

```
        color  red
        width  1
        size  500
        ref  lower left
        hcolor  black
        hwidth  1
        background  white
        border  red
        opaque  y
        end
```

vector

Selects a vector map layer for output.

```
USAGE:  vector  vectormap| list
        style  sequence
        color  [# ] color
        width  #
        hcolor  color
        hwidth  #
        masked  [y| n]
        end
```

GRASS vector data can be portrayed on the map using the **vector** instruction. The name of the vector layer is specified on the first instruction line. The named vector layer must be accessible via the current mapset search path. The *list* option is available in keyboard mode.

The default vector line style is a continuous, solid line, but the user can specify a dashed line using the **style** parameter. The **style** parameter can contain a sequence of digits (0−9) that represent a colored pattern on the vectors. Colors can be assigned to each non−zero digit by using the **color** parameter multiple times. If the **color** parameter is used without a specified digit, the named color will be assigned to the entire lengths of the vectors. Colors are listed in the VALID COLOR NAMES section in this manual entry.

The user can specify the vector line **width** in pixels. A highlight color can be assigned with **hcolor**, and the highlight's width in pixels can be assigned with **hwidth**. The user can also specify if the vectors are to be **masked** by the current mask. (See manual entry for *r.mask* for more information on the mask.)

The **vector** instruction set must be completed by an **end** terminator.

The **vector** instruction can be used more than once to portray multiple vector data layers.

This example would include a vector map layer named "streams" in the output map. These streams would be a total of four pixels wide (two blue pixels with a white outer highlight one pixel wide on each side). The map would not show streams outside of the current mask.

```
EXAMPLE:
        vector  streams
        color  blue
        width  2
        hcolor  white
        hwidth  1
        masked  y
        end
```

The following example would portray a vector map layer named "roads". These roads would be two pixels wide and would be dashed blank−black−red (the blank areas would show other map elements under the roads). The roads would be visible inside and outside of the current mask.

```
EXAMPLE:
        vector  roads
        width  2
        style  001122
        color  1 black
        color  2 red
        masked  n
        end
```

verbose

Sets the amount of feedback sent out by *p.map.new*.

USAGE: **verbose** *0| 1| 2*

A higher value set using **verbose** results in more feedback. The default is 2.

This example sets the amount of feedback to a minimum.

```
EXAMPLE:
        verbose  0
```

VALID COLOR NAMES

The following are the valid color names in *p.map.new*:

```
aqua    cyan    indigo  red
black   gray    magenta violet
blue    green   orange  white
brown   grey    purple  yellow
```

any integer from 0 through 124, representing printer color numbers (see *p.colors*manual entry)

VALID FONT NAMES

The following are the valid font names in *p.map.new*:

```
cyrilc  greekcs italict romant
gothgbt greekp  romanc  scriptc
gothgrt greeks  romancs scripts
gothitt italicc romand
greekc  italiccs        romans (default)
```

ICONS VS. PATTERNS

Icons and patterns as used in *p.map.new* are not the same things. Patterns can only be used to cover the extended areas of a raster map layer category. A pattern will repeat above, below and adjacent to itself. Icons are used to represent single points.

Patterns can be defined directly within *p.map.new* using the **defpat** instruction, while icons are created outside of *p.map.new* using the *p.icons*command or a system editor.

EXAMPLE p.map.new INPUT FILE

The following is an example of a *p.map.new* script file. The file has been named "spear.soils". For the purposes of illustration only, the file is shown in two columns. This script file can be entered at the command line.

p.map.new input=*spear.soils*

```
                (cont.)
raster soils              defpat diag
vector streams            000001
   color blue             00001
   width 2                0001
   hcolor white           001
   hwidth 1               01
   masked y               1
   end                       color 1 red
vector roads                 end
   width 2                setpat 4 diag
   style 001122           text 608000 3476004 SPEARFISH SOILS MAP
   color 1 black             color red
   color 2 red               width 2
   masked n                  hcolor black
   end                       hwidth 1
labels town.names            background white
region subregion             border red
   color white               size 500
   width 2                   ref lower left
   end                       opaque y
grid 10000                   end
   color green            line 606969 3423092 616969 3423092
   numbers 2 red            color yellow
   end                       width 2
outline                      opaque yes
   color black               end
   end                    point 40% 60%
colortable\ y                color purple
comments                     icon diamond
   This is a comment         size 2
   end                       masked n
scale 1:25000                end
setcolor 6,8,9 white      end
setcolor 10 green
```

INTERACTIVE MODE

If the user enters *p.map.new* on the command−line without arguments, a prompting session occurs. Some, but not all, of the non−interactive requests are available in this mode.

SEE ALSO

g.mapsets
g.region

p.chart
p.colors
p.icons
p.labels
p.ppm
p.select
r.mask

AUTHOR

Michael Shapiro, U.S.Army Construction Engineering Research Laboratory
Joo Joo Chia, U.S. Army Construction Engineering Research Laboratories

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

NAME

p.ppm − Reads portable pixmap (ppm) files created by PPM utilities.
(GRASS Paint/Print Program)

SYNOPSIS

p.ppm
p.ppm help
p.ppm [**−f**] [**input=***name*]

DESCRIPTION

This program, *p.ppm*, reads a user−specified portable pixmap (ppm) file and outputs it to the currently selected printer (see *p.select*). The *input* ppm file should be one that has been created using the PPM utilities developed by Jeff Poskanzer. These utilities can import various image formats (including Sun raster, X Windows pixmaps, and others) into the PPM formats *ppm* (color pixmaps), *pgm* (grey scale maps), and *pbm* (black and white bit−maps).

If the image doesn't fit the output device, it won't get printed. If you want the image printed (but clipped), use the **p.ppm −f** option, or scale the input using *ppmscale*, or rotate the image using *ppmrotate* (if it will fit that way −− otherwise you might have to scale it as well). If the image doesn't fit, *p.ppm* will tell you what scaling value to enter to *ppmscale* that will make it fit.

EXAMPLES

If the user is running GRASS under SunOS, the following command could be used to send a monitor screen image to the printer:

screendump | rasttoppm | p.ppm

If you are running suntools, the user might type:

sleep 10; screendump | rasttoppm | ppmrotate 90 | p.ppm

The UNIX *sleep* command allows you time to arrange the frames before the screen dump starts. The *ppmrotate* is usually needed because the Sun screens are wider than they are long (and wider than 1024 pixels − which is the width of most of our printers).

If you are running X, the user might type:

xwd | xwdtoppm | ppmrotate 90 | p.ppm

NOTES

This program only supports the ppm binary format (P6).

Maximum color level is 255. If the ppm file has more color levels, use *ppmcscale* to reduce the number of colors.

No scaling is done. Use *ppmscale* to change image size.

No rotation is done. Use *ppmrotate* to rotate the image.

SEE ALSO

See also:

The PPM utilities *ppmrotate* (rotates ppm images), *ppmscale* (scales ppm images for printing), *rasttoppm* (converts a Sun raster file to ppm format), and *xwdtoppm* (converts an X Windows dump file to ppm format).

The SunOS program *screendump* (dumps the image on the color graphics monitor into a file in Sun raster file format).

The X program *xwd* (dumps the image in an X window into a file in X window dump [xwd] format).

The GRASS programs
d.save
p.map
p.select
parser

AUTHOR

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

This program uses the PPM utilities, developed by Jeff Poskanzer.

Last changed: $Date: 2002/01/25 05:45:34 $

NAME

p.select − Selects a device (printer) for GRASS hardcopy output.
(GRASS Hardcopy Output Program)

SYNOPSIS

p.select
p.select help
p.select [**−lpq**] [**painter=***name*]

DESCRIPTION

p.select allows the user to select the device for GRASS hardcopy output. The user must select a device *before* running the other GRASS hardcopy output functions (e.g., *p.map*, *p.labels*, *p.chart*).

OPTIONS

Flags:

−l
: List all available painters.

−p
: Print name of currently selected painter.

−q
: Quietly select painter.

Parameter:

painter=*name*
: Name of painter to select.
Options: (this will be a list of available hardcopy output devices, and also *preview*, the user's graphics monitor)

INTERACTIVE MODE

If the user runs *p.select* without specifying program arguments on the command line, the program will prompt the user for the name of a hardcopy output device to select.

NOTE

The options available with *p.select* vary from system to system and depend upon which drivers have been compiled.

SEE ALSO

p.chart
p.labels
p.map

AUTHOR

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

Last changed: $Date: 2002/01/25 05:45:34 $

NAME

p.vrml − output of VRML code from GRASS raster maps

This version only outputs raster maps in VRML 1.0 format. The newer VRML 2.0 format will be more efficient for geographic applications, as it introduces an "ElevationGrid" node so that only the elevation points will have to be written instead of the whole geometry. The vast majority of VRML viewers currently only support VRML 1.0.

SYNOPSIS

p.vrml elev=name [**color=**name] **output=**name

Parameters:

elev
: Name of elevation raster file to use for surface topography.

color
: Name of raster file to use for surface color.

exag
: Exaggeration factor for the vertical dimension.
Default: 1.0

output
: Name of new VRML file. If the extension "wrl" (world) is not present in the name, it will be added.

WARNING:

VRML is not well suited for large geometrys which can result from even a small geographic region. Most viewers seem to bog down with more than 12,000 polygons, depending on your hardware & specific viewer. Each grid cell results in two polygons, so a reasonable size region would be something less than about 75x75. For improved performance and smaller file size, leave off a color map. Since VRML is ascii text, gzip works very well to significantly compress file size.

BUGS:

Currently the region is transformed to a unit size, so real geographic location is lost. Side effects when working in a lat−lon location are that besides general distortion due to projection, a very small exaggeration factor (on order of .001) must be used to compensate for vertical units expected to be the same as map units.

NOTE

This is a preliminary release of "p.vrml", a GRASS program to output GRASS data in the format of Virtual Reality Modeling Language (VRML).

For further information about VRML and available viewers for various platforms, see:

http://vag.vrml.org/
http://www.sdsc.edu/vrml/

TODO

Future plans for this module are to allow draping of sites objects and vector files and using the new sites format available in floating point GRASS to embed WWW links into site objects. It will also be upgraded to support VRML 2.0 and will allow entering multiple preset "views" using the existing GRASS 3d_view file format.

Other possible additions:

- Allow animation of elevation, color, or sites based on user interaction.
- Degradation of the raster to produce TINs for improved performance.

AUTHOR

Bill Brown, US Army Construction Engineering Research Laboratory

Last changed: $Date: 2002/01/25 05:45:34 $