Subject: Working with Specific Colors in IDL
Posted by davidf on Mon, 08 Mar 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Martin Schultz (mgs@io.harvard.edu) writes in an article
on another subject:

> it would certainly be
> helpful to have predefined drawing colors as default (instead of or in
> addition to) the grey scale color table which first has to be
> overwritten (and this seems to be a running themke on this newsgroup
> too). Best would be to have a standard set of named colors so that you
> could write   plot,color=black and it would work no matter whether you
> have < 256 colors or 16M. This shouldn't be hard to implement for true
> color systems

Several weeks ago now Liam Gumley offered a color table in
this newsgroup that represented the 16 colors offered in
the McIDAS color map. I liked those colors so much that I
added them to my GETCOLOR program. At the same time, I
updated GETCOLOR to make it a little easier to use.

Those of you who have used GETCOLOR know that the
purpose of it is to be able to ask for a color by "name".
But I added a second positional parameter to it so that
you can now pass it an index number where the color
should be loaded. For example, suppose you want to
draw a plot in yellow. You can do this:

```
  yellow = GetColor("yellow", 10)
  Plot, Findgen(11), Color=yellow
```

I tend to use it like this. Suppose I want a gray
background, green axes, and yellow data colors:

```
     ; Load the colors.
  yellow = GetColor("yellow", !D.Table_Size-4)
  green = GetColor("green", !D.Table_Size-3)
  gray = GetColor("gray", !D.Table_Size-2)
     ; Draw the plot.
  Plot, Findgen(11), Color=green, Background=gray
  OPlot, Findgen(11), Color=yellow
```

The code above will work on an 8-bit display or on
a 24-bit display with DECOMPOSED color turned OFF.

If you want to work with DECOMPOSED color turned ON,
it is even more straightforward:

```
   Plot, Findgen(11), Color=GetColor("yellow", /True)
```

It is still possible to get the color triple back that
represents a particular color. Just don't pass the index
parameter:

```
   triple = GetColor("yellow")
   Print, triple
```

While this doesn't address all of Martin's issues, it does
make it a little easier to work with 16 pretty nice colors. :-)

You can download GETCOLOR at this URL:

   http://www.dfanning.com/programs/getcolor.pro

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: Working with Specific Colors in IDL
Posted by edors on Mon, 29 Mar 1999 08:00:00 GMT

View Forum Message <> Reply to Message

David,
 I like your getcolor program, nicely implemented. I frequently use
a similar method to define fixed colormap which adds the eight crayola
colors to the bottom of my color table and find that it greatly facilitates
annotation. When I initially started playing with the color tables and
defining annotation colors, I wrote a small test program to try to find
basic colors in a colormap which is already loaded. I thought that you and
the people reading this thread might find it an interesting approach. It
turns out that it is not a very realistic approach for 8-bit color tables
(and I am not sure now to extend it to 24-bit color tables since tvclt
won't help me there). Anyway, here it is.

  Cheers,

   Eric

```
FUNCTION setcolors
;+
; NAME:
;       setcolors
;
; PURPOSE:
;       This procedure tries to find the 9 nine basic colors in the
;       current colormap.
;
; CATEGORY:
;       Utilitiy
;
; CALLING SEQUENCE:
;
;       cs = setcolors()
;
; OUTPUTS:
;
;       cs:    A structure containing the color indices of black, red,
;              green, blue, cyan, magenta, yellow, grey, and white; if
;              they could be found in the current color table.
;
; PROCEDURE:
;       The RGB values of the color table are searched for 9 basic
;       colors.  The color table indices of these colors are returned,
;       or -1, if the color wasn't found.
;
;       This code was built for experimentation, it was written in an
;       rather brute force fasion.  It may be more sucessful in
;       another color space, I hope to try this in the future.
;
; EXAMPLE:
;
;               cs = setcolors()
;
; Written by:   Eric E. Dors, 29 March 1999.
;
; MODIFICATION HISTORY:
;
;-

  cs={ white:0, black:0, yellow:0, magenta:0, cyan:0, $
       red:0, green:0, blue:0, grey:0, colortable:intarr(2) }

  red_arr=bytarr(!d.table_size)
  green_arr=bytarr(!d.table_size)
  blue_arr=bytarr(!d.table_size)
```

```
tvlct, /get, r, g, b

cs.black  =(where((r EQ   0) AND (b EQ   0) AND (g EQ   0)))(0)
cs.red    =(where((r EQ 255) AND (b EQ   0) AND (g EQ   0)))(0)
cs.green  =(where((r EQ   0) AND (b EQ   0) AND (g EQ 255)))(0)
cs.blue   =(where((r EQ   0) AND (b EQ 255) AND (g EQ   0)))(0)
cs.cyan   =(where((r EQ   0) AND (b EQ 255) AND (g EQ 255)))(0)
cs.magenta=(where((r EQ 255) AND (b EQ 255) AND (g EQ   0)))(0)
cs.yellow =(where((r EQ 255) AND (b EQ   0) AND (g EQ 255)))(0)
cs.grey   =(where((r EQ 200) AND (b EQ 200) AND (g EQ 200)))(0)
cs.white  =(where((r EQ 255) AND (b EQ 255) AND (g EQ 255)))(0)

deltafuzz = 10
botfuzz = 50
topfuzz = 200
IF cs.black EQ -1 THEN BEGIN
   tmpcolor = where( (r le botfuzz) AND (b LE botfuzz) AND (g LE botfuzz), $
                     n_hits)
   cs.black = tmpcolor(n_hits/2)
ENDIF
IF cs.red EQ -1 THEN BEGIN
   tmpcolor = where((r GE topfuzz) AND (b LE botfuzz) AND (g LE botfuzz), $
                    n_hits)
   cs.red = tmpcolor(n_hits/2)
ENDIF

IF cs.green EQ -1 THEN BEGIN
   tmpcolor = where((r LE botfuzz) AND (b LE botfuzz) AND (g GE topfuzz), $
                    n_hits)
   cs.green = tmpcolor(n_hits/2)
ENDIF
IF cs.blue EQ -1 THEN BEGIN
   tmpcolor = where((r LE botfuzz) AND (b GE topfuzz) AND (g LE botfuzz), $
                    n_hits)
   cs.blue  = tmpcolor(n_hits/2)
ENDIF
IF cs.cyan EQ -1 THEN BEGIN
   tmpcolor = where((r LE botfuzz) AND (b GE topfuzz) AND (g GE topfuzz), $
                    n_hits)
   cs.cyan = tmpcolor(n_hits/2)
ENDIF
IF cs.magenta EQ -1 THEN BEGIN
   tmpcolor = where((r GE topfuzz) AND (b GE topfuzz) AND (g LE botfuzz), $
                    n_hits)
   cs.magenta = tmpcolor(n_hits/2)
ENDIF
IF cs.yellow EQ -1 THEN BEGIN
```

```
     tmpcolor = where((r GE topfuzz) AND (b LE botfuzz) AND (g GE topfuzz), $
                     n_hits)
     cs.yellow = tmpcolor(n_hits/2)
  ENDIF
  IF cs.grey EQ -1 THEN BEGIN
     tmpcolor = where( ((r GT 200-deltafuzz) AND (r LT deltafuzz*200+deltafuzz)) AND $
                      ((b GT 200-deltafuzz) AND (b LT deltafuzz*200+deltafuzz)) AND $
                       ((g GT 200-deltafuzz) AND (g LT deltafuzz*200+deltafuzz)), $
                      n_hits)
     cs.grey  = tmpcolor(n_hits/2)
  ENDIF
  IF cs.white EQ -1 THEN BEGIN
     tmpcolor = where((r GE topfuzz) AND (b GE topfuzz) AND (g GE topfuzz), $
                     n_hits)
     cs.white = tmpcolor(n_hits/2)
  ENDIF

  return, cs
end
```

davidf@dfanning.com (David Fanning) writes:

> Martin Schultz (mgs@io.harvard.edu) writes in an article
> on another subject:
>
>> it would certainly be
>> helpful to have predefined drawing colors as default (instead of or in
>> addition to) the grey scale color table which first has to be
>> overwritten (and this seems to be a running themke on this newsgroup
>> too). Best would be to have a standard set of named colors so that you
>> could write   plot,color=black and it would work no matter whether you
>> have < 256 colors or 16M. This shouldn't be hard to implement for true
>> color systems
>
> Several weeks ago now Liam Gumley offered a color table in
> this newsgroup that represented the 16 colors offered in
> the McIDAS color map. I liked those colors so much that I
> added them to my GETCOLOR program. At the same time, I
> updated GETCOLOR to make it a little easier to use.
>
> Those of you who have used GETCOLOR know that the
> purpose of it is to be able to ask for a color by "name".
> But I added a second positional parameter to it so that
> you can now pass it an index number where the color
> should be loaded. For example, suppose you want to

> draw a plot in yellow. You can do this:
>
>   yellow = GetColor("yellow", 10)
>   Plot, Findgen(11), Color=yellow
>
> I tend to use it like this. Suppose I want a gray
> background, green axes, and yellow data colors:
>
>   ; Load the colors.
>   yellow = GetColor("yellow", !D.Table_Size-4)
>   green = GetColor("green", !D.Table_Size-3)
>   gray = GetColor("gray", !D.Table_Size-2)
>   ; Draw the plot.
>   Plot, Findgen(11), Color=green, Background=gray
>   OPlot, Findgen(11), Color=yellow
>
> The code above will work on an 8-bit display or on
> a 24-bit display with DECOMPOSED color turned OFF.
>
> If you want to work with DECOMPOSED color turned ON,
> it is even more straightforward:
>
>   Plot, Findgen(11), Color=GetColor("yellow", /True)
>
> It is still possible to get the color triple back that
> represents a particular color. Just don't pass the index
> parameter:
>
>   triple = GetColor("yellow")
>   Print, triple
>
> While this doesn't address all of Martin's issues, it does
> make it a little easier to work with 16 pretty nice colors. :-)
>
> You can download GETCOLOR at this URL:
>
>   http://www.dfanning.com/programs/getcolor.pro
>
> Cheers,
>
> David
>
> --
> David Fanning, Ph.D.
> Fanning Software Consulting
> Phone: 970-221-0438 E-Mail: davidf@dfanning.com
> Coyote's Guide to IDL Programming: http://www.dfanning.com/
> Toll-Free IDL Book Orders: 1-888-461-0155

--

===
| Eric E. Dors | SMTP: edors@lanl.gov |
| Los Alamos National Laboratory | WWW: http://nis-www.lanl.gov |
===