Subject: Re: 8-bit vs. 24-bit color on Windows
Posted by Martin Schultz on Fri, 22 Jan 1999 08:00:00 GMT
View Forum Message <> Reply to Message

William Thompson wrote:
>
> I know this question has been asked many times before, but I'm afraid I don't
> remember what the answer is. Is there any way to convince IDL to use 8-bit
> pseudo-color on a Windows computer with a 16-bit or higher display? I know
> that in other operating systems this is done by using
>
> DEVICE,PSEUDO_COLOR=8
>
> However, this is not supported under Windows. I tried
>
> DEVICE,DECOMPOSED=0
>
> which the documentation claims will make routines work like they did before,
> but this doesn't appear to be the whole story. With DECOMPOSED=0, data will
> come up with the correct color, but only if they are displayed after the color
> table is loaded. This is unacceptable. There must be another step to convince
> IDL to use 8-bit pseudo-color, and if there isn't then RSI must address this.
>
> I've checked David Fanning's Coyote Guide (http://www.dfanning.com/), but the
> only suggestion I could find there that meets my needs is to set Windows to run
> at 256 colors. I'm perfectly happy to do that, but it would be nice to be able
> to use 16-bit or 24-bit for those programs which need it, and 8-bit color for
> IDL. This is possible in other operating systems; can it be done in Windows?
>
> William Thompson

As I just got this new PC with Windows and IDL, I had thought that I woul dhave a very hard time running all my unix based IDL programs on it after I had read so many *color* questions (and David's answers) on this newsgroup. Turns out, it wasn't so bad after all: I made a few fixes to my myct program which I always use to load a colortable and define drawing colors, and voila, I can run all my programs, indexing colors from 0 to 255 as before (and I still have 16M colors available for further use ;-). Besides decomposed=0, the major trick seems to be to limit loadct and/or tvlct to use only 256 colors at maximum. MyCT uses !D.N_Colors to determine the number of colors available in the system and (in it's latest version) truncates the actual maximum number that shall be used to 256 for compatibility with standard unix environments.

Please find the latest source for myct.pro attached - if you don't want to use the program as such, it may give you some idea about the control over colors.

Regards,
Martin.

--
---------- -------
Dr. Martin Schultz
Department for Engineering&Applied Sciences, Harvard University
109 Pierce Hall, 29 Oxford St., Cambridge, MA-02138, USA

phone: (617)-496-8318
fax : (617)-495-4551

e-mail: mgs@io.harvard.edu
Internet-homepage: http://www-as.harvard.edu/people/staff/mgs/
---------- -------
```
; $Id: myct.pro,v 1.10 1999/01/22 20:12:17 mgs Stab $
;---------- --
;+
; NAME:
;       MYCT
;
; PURPOSE:
;       Define a set of standard drawing colors and load a
;       colortable on top of these. The color table can be manipulated
;       in various ways (see KEYWORD PARAMETERS).
;       Standard drawing colors are:
;          0 : white          11 : black
;          1 : black          12 : 85% grey
;          2 : red            13 : 67% grey
;          3 : green          14 : 50% grey
;          4 : blue           15 : 33% grey
;          5 : yellow         16 : 15% grey
;          6 : magenta        17 : white
;          7 : cyan
;          8 : lightred
;          9 : lightgreen
;         10 : lightblue
;
; CATEGORY:
;       color table manipulation
;
; CALLING SEQUENCE:
;       MYCT,table [,keywords]
;
; INPUTS:
;       TABLE --> [optional] number of the color table to be used
;             If no number is provided, the routine will only define
;             the standard drawing colors (unless NO_STD is set).
```

```
;             MYCT will always load a dummy colortable, therefore
;             it ensures that the system variable !D.N_COLORS is set
;             correctly afterwards. If you want to define the number
;             of available colors by using a dummy WINDOW command
;             (see IDL help), you must issue the WINDOW command *before*
;             the call to myct.
;
; KEYWORD PARAMETERS:
;       BOTTOM --> specify where to start color table (see BOTTOM keyword
;             in loadct). Default is number of standard drawing colors+1
;             or 0 (if NO_STD is set). If BOTTOM is less than the number
;             of standard drawing colors (17), no standard colors will be
;             defined (equivalent to setting NO_STD).
;
;       NCOLORS --> number of color indices to be used by the color table.
;             Default is !D.N_COLORS-BOTTOM
;
;       RANGE --> a two element vector which specifies the range of colors
;             from the color table to be used (fraction 0-1). The colortable
;             is first loaded into the complete available space, then
;             the selected portion is interpolated in order to achieve the
;             desired number of colors.
;             RANGE is only effective when a TABLE parameter is given.
;
;       REVERSE --> reverse color table
;
;       SATURATION --> factor to scale saturation values of the extra
;             color table. Saturation ranges from 0..1 (but the factor
;             is free choice as long as positive)
;
;       VALUE --> factor to scale the "value" of the added colortable.
;             Value ranges from 0..1; 0 = black, 1 = white.
;
;       NO_STD --> prevents definition of standard drawing colors.
;
; OUTPUTS:
;
; SUBROUTINES:
;
; REQUIREMENTS:
;
; NOTES:
;       It is recommended to use the COLOR= keyword in all PLOT commands.
;       This will ensure correct colors on (hopefully) all devices.
;       In order to get 256 colors on a postcript printer use
;       DEVICE,/COLOR,BITS_PER_PIXEL=8
;
; EXAMPLE:
```

```
;       .comp myct    ; needed to invoke testmyct program
;       testmyct      ; draw a sine wave with all available colors
;       myct,8,/no_std      ; load colortable green-white
;                     ; identical result as loadct,3
;       wait,4
;       myct,27,NCOLORS=20  ; change first 16 colors to standard drawing
;                     ; colors and add EOS-B color table as color
;                     ; index 17 to 36
;       wait,4
;       myct,0,bottom=37,ncol=20,/reverse,/no_std
;                     ; add reversed grey scale table on top
;       wait,4
;       myct,27,bottom=57,ncol=40,/no_std,range=[0.1,0.7],sat=0.7
;                     ; add a less saturated version of a fraction
;                     ; of the EOS-B color table in the next 40 indices
;       ; NOTE that color indices above 97 will still contain the upper
;       ; portion of the green-white color table.
;
;       On b/w terminals MYCT can be used to revert black and white:
;          myct,0,/rev     (may need further testing and development)
;
; MODIFICATION HISTORY:
;       mgs, 06 Feb 1997: VERSION 1.00
;       mgs, 03 Aug 1997: added input parameter and template
;       mgs, 26 Mar 1998: added NCOLORS keyword
;       mgs, 06 Apr 1998: added BOTTOM, RANGE, and RGB keywords
;       mgs, 04 May 1998: added test for null device
;       mgs, 03 Jun 1998: return if !D.N_COLORS is less than 3 (b/w)
;       mgs, 16 Jun 1998: bug fix: range check now after tvlct
;       mgs, 18 Jul 1998: bug re-visited, added HLS keyword and changed
;             default to HSV. Also added Saturation and Value keywords.
;       mgs, 12 Aug 1998: re-written with bug fixes and more concise.
;             removed RGB and HLS keywords, added REVERSE and NO_STD
;             keywords.
;       mgs, 14 Jan 1999: limit oldcolors and ncolors to MaxColors (256)
;             on PC with TrueColor Graphics to ensure compatibility
;             with Unix.
;
;-
; Copyright (C) 1997-1999, Martin Schultz, Harvard University
; This software is provided as is without any warranty
; whatsoever. It may be freely used, copied or distributed
; for non-commercial purposes. This copyright notice must be
; kept with any copy of this software. If this software shall
; be used commercially or sold as part of a larger package,
; please contact the author to arrange payment.
; Bugs and comments should be directed to mgs@io.harvard.edu
; with subject "IDL routine myct"
```

```
;---------------------------------------------------------- --


pro myct,table,BOTTOM=bottom,NCOLORS=ncolors,RANGE=range,   $
        REVERSE=REVERSE_COLORS,SATURATION=saturation,VALUE=value, $
        NO_STD=NO_STD


   ; check for NULL device. Can be used to identify remote IDL sessions
   if (!d.name eq 'NULL') then return


   ;  ============================================================
   ; set defaults
   ;  ============================================================

   ; color vectors for standard drawing colors
red  =[  255,  0,255,  0,  0,255,255,  0,255,127,127,0,90,150,188,220,240,255]
green=[  255,  0,  0,255,  0,255,  0,255,127,255,127,0,90,150,188,220,240,255]
blue =[  255,  0,  0,  0,255,  0,255,255,127,127,255,0,90,150,188,220,240,255]

   no_std = keyword_set(no_std)
   if (n_elements(bottom) eq 0) then begin
      if (no_std) then bottom = 0 $
      else bottom = n_elements(red)
   endif


   ; limit NCOLORS to 256 on TrueColor PC to ensure compatibility
   if (n_elements(ncolors) eq 0) then ncolors = ( !d.n_colors-bottom > 2 ) < 256
   if (n_elements(saturation) eq 0) then saturation = 1.  ; leave unchanged
   if (n_elements(value) eq 0) then value = 1.  ; leave unchanged

   if (bottom lt n_elements(red)) then no_std = 1

   ;  ============================================================
   ; Set maximum allowed color number
   ; For now, restrict to 256 to ensure compatibility between
   ; Workstations and PC
   ;  ============================================================

   MaxColors = 256

   ;  ============================================================
   ; Get current entries in colortable
   ; NOTE: Upon startup,
   ; * the colortable results in 256 grey scale entries
   ; * !D.N_Colors is not defined correctly
```

```
;  ============================================================

tvlct,rback,gback,bback,/get

; remember number of colors available
; may change after !D.N_Colors is determined correctly
oldcolors = !D.N_COLORS

;  ============================================================
; load temporary new colortable in position starting from 0
;  ============================================================

; if no table is specified, load only two colors:
; this should be save also on b/w terminals
if (n_elements(table) eq 0) then begin
   loadct,0,bottom=0,ncolors=2
endif else $

; if RANGE is given, use all available color indices
if (n_elements(RANGE) eq 2) then begin
   loadct, table, bottom=0
endif else $

; otherwise load number of colors desired
begin
   loadct, table, bottom=0, ncolors=ncolors
endelse

; get entries of new color table
tvlct, r,g,b, /get

; At this point, the correct value of !D.N_Colors has been
; determined. Adjust MaxColors if necessary
MaxColors = ( MaxColors < !D.N_Colors )


; On b/w terminals: does the following help ??  ###
; if (!D.N_COLORS lt 3) then return

; if number of available colors has changed due to loadct
; (i.e. during IDL startup) then create dummy backup color
; arrays with the correct number of colors.
; Limit array size to MaxColors
; (Don't change top value: this indicates the byte entry 'white'
; and is not a color index!)
if (!D.N_COLORS ne oldcolors) then begin
   rback = bytscl(findgen(MaxColors),top=255)
   gback = rback
```

```
    bback = rback
  endif

  ;  ============================================================
  ; Now handle requested settings for new colortable
  ;  ============================================================
  ; adjust value of NCOLORS, BOTTOM and NO_STD if necessary
  if (bottom ge MaxColors) then begin
    bottom = 0
    no_std = 1
  endif
  if (bottom+ncolors gt MaxColors) then $
      ncolors = MaxColors-bottom > 2


  ;  ============================================================
  ; if no color table is requested, only set drawing colors
  ;  ============================================================

  if (n_elements(table) eq 0) then goto,ONLY_SET_DRAWINGCOLS


  ;  ============================================================
  ; filter colors that are actually used and insert them
  ; into backup table
  ;  ============================================================

  ; if RANGE is provided, extract subset and interpolate colortable
  if (n_elements(RANGE) eq 2) then begin
    ; convert percentage to index
    irange = fix( (range < 1.0) * MaxColors )
    if (irange(0) eq irange(1)) then $
       irange(1) = irange(0)+1
    irange = ( (irange(sort(irange)) > 0) < (MaxColors-1) )

; print,'##IRANGE:',IRANGE
    ; extract portion of colortable that shall be used
    r = r(irange[0]:irange[1])
    g = g(irange[0]:irange[1])
    b = b(irange[0]:irange[1])

    ; and interpolate
HSV_EXPAND:
    COLOR_CONVERT,r,g,b,h,s,v,/RGB_HSV
    ; expand and interpolate color values
    h = congrid(h,fix(NCOLORS),/interp)
    s = congrid(s,fix(NCOLORS),/interp)
    v = congrid(v,fix(NCOLORS),/interp)
```

```
; help,r,h

   endif else $

   ; if no range given, extract number of colors requested and
   ; convert them to HSV
   begin
      COLOR_CONVERT,r(0:ncolors-1),g(0:ncolors-1),b(0:ncolors-1),h ,s,v,/RGB_HSV
   endelse

   ; revert colortable if wished

   if (keyword_set(REVERSE_COLORS)) then begin
      h = reverse(h)
      s = reverse(s)
      v = reverse(v)
   endif

; color_convert,h,s,v,rr,gg,bb,/hsv_rgb & print,rr,gg,bb

   ;  ============================================================
   ; adapt saturation and "lightness" for color palette
   ; (requires color values in HSV system)
   ;  ============================================================

   s = ((s * saturation) > 0) < 1
   v = ((v * value) > 0) < 1

   ;  ============================================================
   ; overwrite portion of original colortable and store it
   ;  ============================================================

   ; restore original first, then overwrite.
   ; (if too flickering, think new)

   tvlct,rback,gback,bback
   tvlct,h,s,v,bottom,/HSV

   ;  ============================================================
   ; set standard drawing colors
   ;  ============================================================
ONLY_SET_DRAWINGCOLS:

   if (n_elements(red) le !D.N_COLORS AND no_std eq 0 ) then  $
      TVLCT, red, green, blue

end
```

```
; ********************************************************

pro testmyct

   ; test program for myct
   ; call testmyct, then invoke myct with various options
   !p.position=[0.04,0.04,0.97,0.97]

   myct,0,/no_std

   x=findgen(!D.n_colors)/!D.N_COLORS*!PI*4
   y = sin(x)
   c=bytscl(x,top=!D.n_colors)

   plot,x,y,/nodata,xstyle=5,ystyle=5
   plots,x,y,psym=sym(1),symsize=1.3,col=c

   !p.position=0
end
```

File Attachments

1) myct.pro, downloaded 116 times