Subject: Re: Wanted: colour table good for high contrast grayscale output
Posted by Martin Schultz on Fri, 06 Mar 1998 08:00:00 GMT
View Forum Message <> Reply to Message

oops! I just noticed that you need a few extra routines from my library
in order to run the demo I just sent. Here they are (and I hope David
won't mind that I attach his TVIMAGE for completeness):

Martin.

```
 ;------------------------------------------------------------ --
;+
; NAME:
;       MYCT
;
; PURPOSE:
;       load a color table and define the first 16 colors for
;       drawing colors (white,black,red,green,blue,yellow,magenta,
;       lightblue,lightred,lightgreen,purple,black,85%grey,67%grey,
;       50%grey,33%grey,15%grey).
;
; CATEGORY:
;       color table manipulation
;
; CALLING SEQUENCE:
;       MYCT
;
; INPUTS:
;       TABLE --> [optional] number of the color table to be used
;            default is EOS-B (number 27)
;
; KEYWORD PARAMETERS:
;
; OUTPUTS:
;
; SUBROUTINES:
;
; REQUIREMENTS:
;
; NOTES:
;       It is recommended to use the COLOR= keyword whenever possible
;       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:
;
; MODIFICATION HISTORY:
```

```
;       mgs, 06 Feb 1997: VERSION 1.00
;       mgs, 03 Aug 1997: added input parameter and template
;
;-
; Copyright (C) 1997, 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

; loads colortable (default EOS-B) and modifies first entries:
;   color 0 becomes whitefor background
;   colors 1..10 become brilliant plot colors
;   colors 11..16 become grey shadings, beginning with black
;   the rest is unaltered

if (n_params() le 0) then table = 27

loadct, table

red  =[  255,  0,255,  0,  0,255,255,  0,255,127,127,0,62,98,172,200,232,255]
green=[  255,  0,  0,255,  0,255,  0,255,127,255,127,0,62,98,172,200,232,255]
blue =[  255,  0,  0,  0,255,  0,255,255,127,127,255,0,62,98,172,200,232,255]
; red  =[  255,  0,255,  0,  0,255,255,  0,255,127,127,0,42,85,128,170,212,255]
; green=[  255,  0,  0,255,  0,255,  0,255,127,255,127,0,42,85,128,170,212,255]
; blue =[  255,  0,  0,  0,255,  0,255,255,127,127,255,0,42,85,128,170,212,255]

TVLCT, red, green, blue
end

 ;------------------------------------------------------------ --
;+
; NAME:
;       OPEN_DEVICE
;
; PURPOSE:
;       If hard copy is to be generated, OPEN_DEVICE opens the
;       PostScript device.  Otherwise OPEN_DEVICE opens an Xwindow.
;
; CATEGORY:
```

```
;       Input/Output
;
; CALLING SEQUENCE:
;       OPEN_DEVICE, OLD_DEVICE, [,keywords]
;
; INPUTS:
;
; KEYWORD PARAMETERS:
;       PS       (int)  -> will send PostScript file to printer
;
;       COLOR    (int)  -> will enable PostScript color mode
;
;       LANDSCAPE (int)  -> will enable PostScript landscape mode
;
;       PORTRAIT  (int)  -> will enable PostScript portrait mode
;
;       FILENAME  (str)  -> The name to be given the PostScript file.
;               Default: idl.ps
;
;       WINPARAM  (int)  -> An integer vector with 3 elements:
;                       WINPARAM(0) = window number
;                       WINPARAM(1) = X dimension of window in pixels
;                       WINPARAM(2) = Y dimension of window in pixels
;
;       _EXTRA -> additional keywords that are passed to the call to
;               the DEVICE routine
;
; OUTPUTS:
;       OLD_DEVICE (str) -> stores the previous value of !D.NAME
;
; SUBROUTINES:
;
; REQUIREMENTS:
;
; NOTES:
;       If PS=0 then
;          Open Xwindow WINPARAM(0), which is WINPARAM(1) pixels wide
;          in the X-direction, and WINPARAM(2) pixels wide in the
;          Y-direction.
;
;       If PS=1 then
;         depending on /PORTRAIT or /LANDSCAPE and /COLOR
;         postscript is enabled in either portrait or landscape
;         mode as color or b/w plot
;
;       The key parameter which determines whether to open a postscript
;       file or a screen window is PS. Therefore, e.g. in a widget
;       application, you can pass a standard set of parameters for both,
```

```
;       postscript and screen, to this routine, and determine the device
;       to be chosen by a button state or checkbox which is passed into PS.
;
;
; EXAMPLE:
;       OPEN_DEVICE, WINPARAM=[0,800,800]
;          opens a screen window of size 800x800 pixels
;
;       OPEN_DEVICE, OLD_DEVICE, /LANDSCAPE, FILENAME='myplot.ps'
;          opens a postscript file named myplot.ps in b/w and landscape
;          orientation
;
;       OPEN_DEVICE, OLDDEV, PS=PS, /PORTRAIT, /COLOR, WIN=2
;          depending on the value of PS either a color postscript file
;          named idl.ps is opened or screen window number 2 in default
;          size.
;
;
; MODIFICATION HISTORY:
;       bmy  15 Aug 1997: VERSION 1.00
;       bmy, 19 Aug 1997: VERSION 1.01
;       mgs, 20 Aug 1997: VERSION 1.02
;
;-
; Copyright (C) 1997, Bob Yantosca, 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 bmy@io.harvard.edu
; with subject "IDL routine open_device"
 ;-------------------------------------------------------------- --


pro open_device, OLD_DEVICE,                          $
              PS=PS, COLOR=COLOR, FILENAME=FILENAME,   $
              LANDSCAPE=LANDSCAPE, PORTRAIT=PORTRAIT,  $
              WINPARAM=WINPARAM, _EXTRA=E

     on_error, 2   ; return to caller

     OLD_DEVICE = !D.NAME    ; retrieve current device


     if (not keyword_set(FILENAME)) then FILENAME = 'idl.ps'
     if (not keyword_set(COLOR))    then COLOR    = 0
```

```
    if (not keyword_set(PORTRAIT)) then LANDSCAPE = 1 ; default

    if (keyword_set(PS)) then begin
       set_plot, 'PS'

       if (keyword_set(LANDSCAPE)) then begin       ;Landscape mode
          device, /landscape, color=COLOR, $
             bits=8, filename=FILENAME, _EXTRA=e

       endif else begin    ; Portrait mode
          device, color=COLOR, bits=8, /portrait,   $
             /inches, xoffset=0.25, yoffset=0.25,   $
             xsize=8.0, ysize=10, filename=FILENAME, _EXTRA=e
       endelse


    endif else begin      ; no postscript desired
                     ; only action if winparam given

       if (n_elements(WINPARAM) gt 0) then begin      ;Open Xwindow
; if winparam is 3 element vector then open window of desired size
;  else open window with standard size
          if(n_elements(winparam) eq 3) then $
             window, WINPARAM(0), xsize=WINPARAM(1), ysize=WINPARAM(2) $
          else  $
             window, winparam(0)
       endif

    endelse


return
end


 ;------------------------------------------------------------ --
;+
; NAME:
;       CLOSE_DEVICE
;
; PURPOSE:
;       CLOSE_DEVICE closes the PostScript device and spawns
;       a print job to the printer specified by the user or
;       it closes a graphics window.
;
; CATEGORY:
;       Input/Output
;
```

```
; CALLING SEQUENCE:
;       CLOSE_DEVICE, OLD_DEVICE, [keywords]
;
; INPUTS:
;       OLD_DEVICE (str) -> Content of !D.NAME before call to OPEN_DEVICE
;           If omitted, 'X' will be used as a default
;
; KEYWORD PARAMETERS:
;       PRINTER    (str) -> name of the printer to send output to
;           Default is 'none', i.e. the postscript file will only be closed
;           and can thenbe manually printed e.g. using the Unix lpr command.
;
;       FILENAME   (str) -> name of the PostScript file
;           Default is 'idl.ps'. NOTE: If a FILENAME was given with
;           OPEN_DEVICE, the same name must be supplied here if the output
;           shall be sent to the printer!
;
;       WINDOW     -> window number to be closed (or -1 if current)
;
;       /TIMESTAMP  -> add a label with filename and system time to the plot
;
; OUTPUTS:
;       If postscript device is active, a *.ps file will be created and/or
;       sent to the printer.
;
; SUBROUTINES:
;
; REQUIREMENTS:
;
; NOTES:
;       The print command (lpr -P) is specific to Unix systems.  Users
;       who are running other operating systems will have to modify this.
;
;       The WINDOW keyword is only evaluated if the current device supports
;       windows [!D.FLAGS AND 256) GT 0]. If you only want to close a
;       postscript file and don't fuss around with your screen windows
;       then simply don't set this keyword.
;
;       The OLD_DEVICE parameter can be misused to set the output device
;       to anything ! Therefore, if you are working on a Unix system it's
;       probably safest to not use it and stick with the 'X' default.
;
;
; EXAMPLE:
;       CLOSE_DEVICE, OLD_DEVICE, PRINTER='pro', FILENAME='myplot.ps'
;          If current device name is PS then 'myplot.ps' will be closed
;          and spooled to printer 'pro'
;
```

```
;       CLOSE_DEVICE, OLD_DEVICE, WIN=-1
;          If current device name is PS then the postscript file will
;          be closed. If the current device is a screen device (that
;          supports windows), then the active window will be deleted.
;
;       CLOSE_DEVICE
;          restores the plotting device to 'X'
;
; MODIFICATION HISTORY:
;       bmy, 18 Aug 1997: VERSION 1.00
;       bmy, 19 Aug 1997: VERSION 1.01
;       mgs, 20 Aug 1997: VERSION 1.02
;
;-
; Copyright (C) 1997, Bob Yantosca, 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 bmy@io.harvard.edu
; with subject "IDL routine close_device"
 ;-------------------------------------------------------------- --


pro close_device, OLD_DEVICE, PRINTER=PRINTER, FILENAME=FILENAME, $
            WINDOW=WINDOW, TIMESTAMP=TIMESTAMP

     on_error, 2

     if (n_params() le 0) then OLD_DEVICE = 'X'

     if (not keyword_set(PRINTER))  then PRINTER  = 'none'
     if (not keyword_set(FILENAME)) then FILENAME = 'idl.ps'

     if (!d.name eq 'PS') then begin   ; if postscript device active

; add timestamp if desired
         if(keyword_set(TIMESTAMP)) then begin  ; draw timestamp if desired
             timelabel = systime(0)
             xyouts,0.98,0.007,filename+'  '+timelabel,color=1,   $
                   align=1,/norm,charsize=0.4
         endif



    device, /close      ; close postscript file
```

```
        if (PRINTER ne 'none') then begin   ; spawn postscript file to printer
         TRIM_PRINTER = strtrim(PRINTER,2)
           print, 'Sending output to printer ' + TRIM_PRINTER
           spawn, 'lpr -P ' + TRIM_PRINTER + ' ' + FILENAME
        endif

    endif else begin     ; no postscript device active

        ; check if device supports windows and if a window shall be closed
        if(n_elements(window) gt 0) then begin
           if(window lt 0) then window = !D.WINDOW
           if( (!D.FLAGS AND 256) GT 0 AND window ge 0) then $
              wdelete,window
        endif

    endelse

    set_plot, OLD_DEVICE

return
end



;+
; NAME:
;    TVIMAGE
;
; PURPOSE:
;    This purpose of TVIMAGE is to allow you to display an image
;    on the display or in a PostScript file in a particular position.
;    The position is specified by means of the POSITION keyword. In
;    this respect, TVIMAGE works like other IDL graphics commands.
;    Moreover, the TVIMAGE command works identically on the display
;    and in a PostScript file. You don't have to worry about how to
;    "size" the image in PostScript. The output on your display and
;    in the PostScript file will be identical. The major advantage of
;    TVIMAGE is that it can be used in a natural way with other IDL
;    graphics commands in resizeable IDL graphics windows. TVIMAGE
;    is a replacement for TV and assumes the image has been scaled
;    correctly when it is passed as an argument.
;
; AUTHOR:
;      FANNING SOFTWARE CONSULTING:
;      David Fanning, Ph.D.
;      2642 Bradbury Court
;      Fort Collins, CO 80521 USA
```

```
;       Phone: 970-221-0438
;       E-mail: davidf@dfanning.com
;       Coyote's Guide to IDL Programming: http://www.dfanning.com
;
; CATEGORY:
;    Graphics display.
;
; CALLING SEQUENCE:
;
;    TVIMAGE, image
;
; INPUTS:
;    image:    A 2D or 3D image array. It should be byte data.
;
; KEYWORD PARAMETERS:
;    _EXTRA:   This keyword picks up any TV keywords you wish to use.
;
;    KEEP_ASPECT_RATIO: Normally, the image will be resized to fit the
;          specified position in the window. If you prefer, you can
;          force the image to maintain its aspect ratio in the window
;          (although not its natural size) by setting this keyword.
;          The image width is fitted first. If, after setting the
;          image width, the image height is too big for the window,
;          then the image height is fitted into the window. The
;          appropriate values of the POSITION keyword are honored
;          during this fitting process. Once a fit is made, the
;          POSITION coordiates are re-calculated to center the image
;          in the window. You can recover these new position coordinates
;          as the output from the POSITION keyword.
;
;    MINUS_ONE: The value of this keyword is passed along to the CONGRID
;          command. It prevents CONGRID from adding an extra row and
;          column to the resulting array.
;
;    POSITION: The location of the image in the output window. This is
;          a four-element floating array of normalized coordinates of
;          the type given by !P.POSITION or the POSITION keyword to
;          other IDL graphics commands. The form is [x0, y0, x1, y1].
;          The default is [0.15, 0.15, 0.85, 0.85]. Note that this can
;          be an output parameter if the KEEP_ASPECT_RATIO keyword is
;          used.
;
; OUTPUTS:
;    None.
;
; SIDE EFFECTS:
;    Unless the KEEP_ASPECT_RATIO keyword is set, the displayed image
;    may not have the same aspect ratio as the input data set.
```

```
;
; RESTRICTIONS:
;    If the POSITION keyword and the KEEP_ASPECT_RATIO keyword are
;    used together, there is an excellent chance the POSITION
;    parameters will change. If the POSITION is passed in as a
;    variable, the new positions will be returned as an output parameter.
;
; EXAMPLE:
;    To display an image with a contour plot on top of it, type:
;
;       filename = FILEPATH(SUBDIR=['examples','data'], 'worldelv.dat')
;       image = BYTARR(360,360)
;       OPENR, lun, filename, /GET_LUN
;       READU, image
;       FREE_LUN, lun
;
;       TVIMAGE, image, POSITION=thisPosition, /KEEP_ASPECT_RATIO
;       CONTOUR, image, POSITION=thisPosition, /NOERASE, XSTYLE=1, $
;          YSTYLE=1, XRANGE=[0,360], YRANGE=[0,360], NLEVELS=10
;
; MODIFICATION HISTORY:
;     Written by:     David Fanning, 20 NOV 1996.
;     Fixed a small bug with the resizing of the image. 17 Feb 1997. DWF.
;     Removed BOTTOM and NCOLORS keywords. This reflects my growing belief
;       that this program should act more like TV and less like a "color
;       aware" application. I leave "color awareness" to the program
;       using TVIMAGE. Added 24-bit image capability. 15 April 1997. DWF.
;     Fixed a small bug that prevented this program from working in the
;        Z-buffer. 17 April 1997. DWF.
;     Fixed a subtle bug that caused me to think I was going crazy!
;        Lession learned: Be sure you know the *current* graphics
;        window! 17 April 1997. DWF.
;     Added support for the PRINTER device. 25 June 1997. DWF.
;     Extensive modifications. 27 Oct 1997. DWF
;        1) Removed PRINTER support, which didn't work as expected.
;        2) Modified Keep_Aspect_Ratio code to work with POSITION keyword.
;        3) Added check for window-able devices (!D.Flags AND 256).
;        4) Modified PostScript color handling.
;     Craig Markwart points out that Congrid adds an extra row and column
;        onto an array. When viewing small images (e.g., 20x20) this can be
;        a problem. Added a Minus_One keyword whose value can be passed
;        along to the Congrid keyword of the same name. 28 Oct 1997. DWF
;-

PRO TVIMAGE, image, KEEP_ASPECT_RATIO=keep, POSITION=position, $
  MINUS_ONE=minusOne, _EXTRA=extra

ON_ERROR, 1
```

```
   ; Check for image parameter.

np = N_PARAMS()
IF np EQ 0 THEN MESSAGE, 'You must pass an image argument.'

   ; Check image size.

s = SIZE(image)
IF s(0) LT 2 OR s(0) GT 3 THEN $
   MESSAGE, 'Argument does not appear to be an image. Returning...'

   ; 2D image.

IF s(0) EQ 2 THEN BEGIN
   imgXsize = FLOAT(s(1))
   imgYsize = FLOAT(s(2))
   true = 0
ENDIF

   ; 3D image.

IF s(0) EQ 3 THEN BEGIN
IF (s(1) NE 3L) AND (s(2) NE 3L) AND (s(3) NE 3L) THEN $
   MESSAGE, 'Argument does not appear to be a 24-bit image. Returning...'
   IF s(1) EQ 3 THEN true = 1 ; Pixel interleaved
   IF s(2) EQ 3 THEN true = 2 ; Row interleaved
   IF s(3) EQ 3 THEN true = 3 ; Band interleaved
   CASE true OF
      1: BEGIN
         imgXsize = FLOAT(s(2))
         imgYsize = FLOAT(s(3))
         END
      2: BEGIN
         imgXsize = FLOAT(s(1))
         imgYsize = FLOAT(s(3))
         END
      3: BEGIN
         imgXsize = FLOAT(s(1))
         imgYsize = FLOAT(s(2))
         END
   ENDCASE
ENDIF

   ; Check for keywords.

IF N_ELEMENTS(position) EQ 0 THEN position = [0.15, 0.15, 0.85, 0.85] $
   ELSE position = FLOAT(position)
```

```
minusOne = Keyword_Set(minusOne)

   ; Maintain aspect ratio (ratio of height to width)?

IF KEYWORD_SET(keep) THEN BEGIN

      ; Find aspect ratio of image.

   ratio = FLOAT(imgYsize) / imgXSize

      ; Find the proposed size of the image in pixels without aspect
      ; considerations.

   xpixSize = (position(2) - position(0)) * !D.X_VSize
   ypixSize = (position(3) - position(1)) * !D.Y_VSize

      ; Try to fit the image width. If you can't maintain
      ; the aspect ratio, fit the image height.

   trialX = xpixSize
   trialY = trialX * ratio
   IF trialY GT ypixSize THEN BEGIN
      trialY = ypixSize
      trialX = trialY / ratio
   ENDIF

      ; Recalculate the position of the image in the window.

   position(0) = (((xpixSize - trialX) / 2.0) / !D.X_VSize) + position(0)
   position(2) = position(0) + (trialX/FLOAT(!D.X_VSize))
   position(1) = (((ypixSize - trialY) / 2.0) / !D.Y_Size)  + position(1)
   position(3) = position(1) + (trialY/FLOAT(!D.Y_VSize))

ENDIF

   ; Calculate the image size and start locations.

xsize = (position(2) - position(0)) * !D.X_VSIZE
ysize = (position(3) - position(1)) * !D.Y_VSIZE
xstart = position(0) * !D.X_VSIZE
ystart = position(1) * !D.Y_VSIZE

   ; Display the image. Sizing different for PS device.

IF (!D.NAME EQ 'PS') THEN BEGIN

      ; Need a gray-scale color table if this is a true
      ; color image.
```

```
   IF true GT 0 THEN LOADCT, 0, /Silent
   TV, image, xstart, ystart, XSIZE=xsize, $
     YSIZE=ysize, _EXTRA=extra, True=true

ENDIF ELSE BEGIN

     ; If the image is 24-bit but the display is 8-bit
     ; then COLOR_QUAN processing is required.

   IF (!D.Flags AND 256) GT 0 THEN BEGIN
     thisWindow = !D.Window
     Window, XSize=10, YSize=10, /Free, /Pixmap
     WDelete, !D.Window
     WSet, thisWindow
   ENDIF
   ncolors = !D.N_Colors
   IF ncolors LE 256 AND true GT 0 THEN BEGIN
     TV, Congrid(COLOR_QUAN(image, true, red, green, blue, $
       Colors=!D.N_Colors), CEIL(xsize), CEIL(ysize), $
       MINUS_ONE=minusOne), xstart, ystart, _EXTRA=extra
     TVLCT, red, green, blue
     RETURN
   ENDIF

   CASE true OF
     0: TV, CONGRID(image, CEIL(xsize), CEIL(ysize), /INTERP, $
          MINUS_ONE=minusOne), xstart, ystart, _EXTRA=extra
     1: TV, CONGRID(image, 3, CEIL(xsize), CEIL(ysize), /INTERP, $
          MINUS_ONE=minusOne), xstart, ystart, _EXTRA=extra, True=1
     2: TV, CONGRID(image, CEIL(xsize), 3, CEIL(ysize), /INTERP, $
          MINUS_ONE=minusOne), xstart, ystart, _EXTRA=extra, True=2
     3: TV, CONGRID(image, CEIL(xsize), CEIL(ysize), 3, /INTERP, $
          MINUS_ONE=minusOne), xstart, ystart, _EXTRA=extra, True=3
  ENDCASE
ENDELSE
END
```

File Attachments

1) myct.pro, downloaded 106 times
2) open_device.pro, downloaded 116 times
3) close_device.pro, downloaded 117 times
4) tvimage.pro, downloaded 121 times