Subject: DirectColor on linux
Posted by doosh on Mon, 22 Nov 1999 08:00:00 GMT
View Forum Message <> Reply to Message

[This message was originally posted to comp.lang.idl; thanks for the redirect]

One of the professors I support has a procedure he's using on Solaris with IDL that he wants to move over to a linux machine. However, there appears to be some fundamental differences in the way Solaris handles colors, and the procedure isn't working.

Basically, what the procedure does is use direct color to change the color table, thereby changing the colors of the images in window 0. This is useful in astronomy because it allows you to highlight different parts of the spectrum. So if you diddle the red from 255 down to 0, all the red will be drained out of the image. I've seen this work on a Solaris machine. I am trying to do it on a Linux machine with a Diamond Viper 770 (32MB) video card.

The problem is, it doesn't work. I can set device,direct=24 under XFree86 or Accelerated X, and IDL reports that it's using direct color, but running the diddle procedure (attached below) doesn't change the colors in window 0.

I don't really understand the issues involved; is it something IDL needs to support, or the X server, or the video card? Is it possible to do this with IDL on Linux at all?
-Tom

```
;***** CARLSTRETCH*****************************************
pro carlstretch, r, g, b, low, high, gamma
;like IDL's 'stretch', but does color independently.
common colors, r_orig, g_orig, b_orig, r_curr, g_curr, b_curr
nc = !d.table_size      ;# of colors entries in device

;help, r, g, b, low, high, gamma
;return
slope = 1. / (float(high) - float(low))         ;Range of 0 to 1.
intercept = -slope * float(low)
p = findgen(nc) * slope + intercept > 0.0
p = long(nc * (p ^ gamma)) < 255

if (r ne 0) then r_curr = r_orig[p]
if (g ne 0) then g_curr = g_orig[p]
if (b ne 0) then b_curr = b_orig[p]

;tempwindow = !d.window
```

```
;wset, 0
;plot, r_curr, xstyle=1, xrange=[0,255], ystyle=1, yrange=[0,255], color=255
;oplot, g_curr, color=255l*256l
;oplot, b_curr, color=255l*256l*256l
;wset, tempwindow

tvlct, r_curr, g_curr, b_curr
return
end

;***** DIDDLE ****************************************
pro diddle, colors, lo2, hi2, gamma2
;diddles gamma and stretch of images in 3 channels independently or in all.
;which color depends on colors. r, g, b or any combination.

;BEGIN WITH THE INPUT STRETCH PARAMETERS IF THEY HAVE BEEN ENTERED
;OTHERWISE, USE THE DEFAULT VALUES.

;FIRST THE COLORS TO PROCESS...
common colors, r_orig, g_orig, b_orig, r_curr, g_curr, b_curr
if (n_params() eq 0) then colors='rgb'
r=0l & g=0l & b=0l
if (strpos( colors, 'r') ne -1) then r=255l
if (strpos( colors, 'g') ne -1) then g=255l
if (strpos( colors, 'b') ne -1) then b=255l
colorout = r + 256l*( g + 256l*b)
print, r, g, b, colorout

;MAKE SURE A COLOR TABLE HAS BEEN LOADED...
catch, err_in
if (err_in ne 0) then begin
loadct, 0
endif
txtst = r_curr(0)

;THEN THE OTHER PARAMETERS...
catch, err_in
if (err_in ne 0) then begin
lo2 = 0
endif
lo1 = float(lo2)/255.

catch, err_in
if (err_in ne 0) then begin
hi2 = 255
endif
hi1 = float(hi2)/255.
```

```
catch, err_in
if (err_in ne 0) then begin
gamma2 = 1.0
endif
gamma1 = 0.5*( 1. + alog10( gamma2))

catch, err_in, /cancel
;lo1 = 0.
;hi1 = 1.
;gamma1 = 0.5
lo2 = byte( lo1 * 255)
hi2 = byte( hi1 * 255)
gamma2 = 0.1*(100.^gamma1)
;print, colors, lo1, hi1, gamma1
;print, lo2, hi2, gamma2

;SAVE THE ORIGINAL WINDOW NUMBER SO THAT WE GO BACK TO IT UPON RETURN...
windownr = !d.window
print, 'original window = ', windownr

;DEFINE AND POPULATE THE DIDDLE WINDOW...
window, xsize=200, ysize=100, xpos=50, ypos=50, /free, retain=2
windowdiddle = !d.window
plot, [0,1], [0,1], xrange=[0, 1], yrange=[0,1], /nodata, $
 xstyle=1, ystyle=1, position=[.05, .05, .95, .95], color=colorout
xyouts, .1, .2, lo2, /normal, charsize=1.5, color=colorout
xyouts, .1, .5, hi2, /normal, charsize=1.5, color=colorout
xyouts, .1, .8, gamma2, /normal, charsize=1.5, color=colorout
lo2old = lo2
hi2old = hi2
gamma2old = gamma2

!err = 2
print, 'left button controls min, mid button gamma, rght button max'
print, 'the horixontal position of the cursor gives the value '
print, 'any two buttons simultaneously leaves the routine.
print, ' '
print, 'STARTING low, high, gamma = ', lo2, hi2, gamma2, $
 '; color = ', colors

beginagn:

;print, r, b, g, lo2, hi2, gamma2
carlstretch, r, g, b, lo2, hi2, gamma2

;stop
cursor, xx, yy, 0, /data
```

```
;print, !err
if ( (!err ne 0) and (!err ne 1) and (!err ne 2) and (!err ne 4)) then begin
wset, windownr
wdelete, windowdiddle
print, 'return to window number ', windownr
print, 'ENDING   low, high, gamma = ', lo2, hi2, gamma2
;stop
return
endif

xx = (0. > xx) < 1.0

if (!err eq 1) then begin
lo2 = byte( xx * 255)
carlstretch, r, g, b, lo2, hi2, gamma2
endif

if (!err eq 4) then begin
hi2 = byte( xx * 255)
carlstretch,  r, g, b, lo2, hi2, gamma2
endif

if (!err eq 2) then begin
gamma2 = 0.1*(100.^xx)
carlstretch,  r, g, b, lo2, hi2, gamma2
endif

xyouts, .1, .2, lo2old, /normal, charsize=1.5, color=0
xyouts, .1, .5, hi2old, /normal, charsize=1.5, color=0
xyouts, .1, .8, gamma2old, /normal, charsize=1.5, color=0
xyouts, .1, .2, lo2, /normal, charsize=1.5, color=colorout
xyouts, .1, .5, hi2, /normal, charsize=1.5, color=colorout
xyouts, .1, .8, gamma2, /normal, charsize=1.5, color=colorout
lo2old = lo2
hi2old = hi2
gamma2old = gamma2
wait, 0.01
goto, beginagn

return
end
```