Subject: Re: changing contrast and brightness on the fly
Posted by John-David T. Smith on Wed, 20 Jun 2001 20:26:08 GMT
View Forum Message <> Reply to Message

"Liam E. Gumley" wrote:
>
> JD Smith wrote:
>> A standard feature of astronomical viewers. For a solution which uses
>> only colormap fiddling (vs. full image rescaling), see atv:
>>
>> http://cfa-www.harvard.edu/~abarth/atv/atv.html
>>
>> This brings up the age old question of how best to dynamically redisplay
>> images (brightnest/contrast/etc.). With only 255 colors, making the top
>> 100 white to increase brightness is pretty wasteful, and cuts down on
>> the dynamic visual range... much better (and slower) is to rescale the
>> image range of interest into the full colormap.
>>
>> Here's how Andrew (and cohorts) did it:
>>
>> ++
>> pro atv_stretchct, brightness, contrast, getmouse = getmouse
>>
>> ; routine to change color stretch for given values of
>> ; brightness and contrast.
>> ; Complete rewrite 2000-Sep-21 - Doug Finkbeiner
>> ; This routine is now shorter and easier to understand.
>>
>> common atv_state
>> common atv_color
>>
>> ; if GETMOUSE then assume mouse positoin passed; otherwise ignore
>> ; inputs
>>
>> if (keyword_set(getmouse)) then begin
>> state.brightness = brightness/float(state.draw_window_size[0])
>> state.contrast = contrast/float(state.draw_window_size[1])
>> endif
>>
>> x = state.brightness*(state.ncolors-1)
>> y = state.contrast*(state.ncolors-1) > 2 ; Minor change by AJB
>> high = x+y & low = x-y
>> diff = (high-low) > 1
>>
>> slope = float(state.ncolors-1)/diff ;Scale to range of 0 : nc-1
>> intercept = -slope*low
>> p = long(findgen(state.ncolors)*slope+intercept) ;subscripts to select
>> tvlct, r_vector[p], g_vector[p], b_vector[p], 8

>>
>> end
>> ++
>
> Is it fair to say that this method will only give satisfactory results
> when IDL is running in 8-bit display mode?

Umm, not necessarily. ATV for instance just issues a redisplay if you're using 24 bit color. The actual breakdown I've discovered (please correct any errors) is:

Psuedo-Color: Shared colormap (usally 8bit), no redisplay necessary. The hardware colormap is read-writeable.

Direct-Color: One large colormap for each of R/G/B, no redisplay necessary. The hardware colormap is read-writeable.

True-Color: No real colormap, colors are expressed in absolute terms. Redisplay necessary (which will usually be fairly much slower than direct manipulation of the hardware color table). IDL will maintain a software translation colormap for you, with decomposed=0. A linear ramp of RGB colors is presumed, and you can't change this directly (the hardware colormaps is read only).

Some machines (usually commercial unices) support multiple visuals at once, a capability termed "overlays". In this case, you can say device,PSUEDO=8, and be up and running, able to write the underlying hardware colormap for fast color-changing. On the PC side, typically you only have one visual class available at a time (and this includes linux).

The pros and cons are:

1. TrueColor:

Pros: Millions of colors, and straightforward to get exactly the color you want, since noone can muck with the underlying colormap (a linear ramp in R,G, and B space)

Cons: Read-only colormap, requiring slow redraws and software-only color translation for normal color table operation.

2. PsuedoColor:

Pros: Lightening fast color manipulation, since you're just loading a table into the display hardware. No redisplay required.

Cons: Usually only 255 colors. Colormap flashing may result (even with

IDL widgets... yuck). This is not a problem for overlayed PsuedoColor visuals (which get their own little 8-bit colorspace to play in). Overlays have limited availability.

3. DirectColor:

Pros: Colormaps can be fiddled, separately for RGB. Redisplay probably *not* required.

Cons: Not as fast at color operations as Psuedo-color. It's quite possible that when using the DECOMPOSED=0 flag with DirectColor, IDL performs the exact same color translation as for TrueColor (though it doesn't need to), and then sets the hardware colormap.

JD
