Subject: A defense of decomposed color
Posted by JD Smith on Mon, 30 Oct 2006 23:46:58 GMT
View Forum Message <> Reply to Message

I wonder if those of you using decomposed color can persuade me of its utility. Though color tables are perfect for image visualization, they are wanting for "system" colors for plot symbols, overlays, etc. It's frustrating to keep track of them, and different apps have different conventions, and can step on each other's feet, causing various undesirable effects.

I presume the reason many of us still use undecomposed color is from the 8bit heritage, when there was no such thing as decomposed color. Do '00FFFF'x-loving people simply assume everyone has a device capable of interpreting 24bit, decomposed color (probably about 95% true, these days)? How do you handle switching back and forth from decomposed (for plot symbols, say) to indexed (for displaying images)? Do you find it really solves the headaches associated with saving a few colors for drawing in high indices, vs. the added juggling needed to switch back and forth among decomposed and non-decomposed color, etc.? What happens if you switch to decomposed color on an 8-bit display system?

I'm ready to come around to embracing direct color specification with no color table intermediary, but I think I need a bit of persuasion.

JD

Subject: Re: A defense of decomposed color
Posted by David Fanning on Mon, 06 Nov 2006 18:17:42 GMT
View Forum Message <> Reply to Message

JD Smith writes:

> Though color tables are stone-age tools, now I can simply:
>
> a) Set up a color table with N image colors and N_max-N "system" or
> plotting colors.
>
> b) Reload that color table on enter events.
>
> c) Plot and/or display whenever, wherever, without having to futz the
> color model beforehand.
>
> Obviously, this requires a device,decomposed=0 in the startup file
> (which is simply documented), but it saves considerable time and
> trouble. The drawback is, of course, different tools have different

> conventions, and you always end up with conflicts.

I think you are right about this, if loading color
tables is faster than changing color models (I haven't
run the tests), and you don't work with true-color images
(so you don't run into differences between Windows and UNIX),
and you can guarantee you are always using the indexed color
model.

I think my motivation for the "don't give a damn WHAT
color model you are using" solution I dreamed up is
having to teach the subject to people who (I swear
to God!) are *never* in the proper color model to see
on their display what I see on mine. 45 minutes of
a class can go by just trying to get everyone to see
the same yellow color I do. :-(

With TVImage and FSC_Color I can at least move along
to more interesting things and everyone moves along
with me. :-)

I think the bottom line is that if you know what you
are doing, any solution that gives you consistent
results is a good solution. If you don't know what
you are doing, some solutions are obviously superior
to others.

Cheers,

David

P.S. And with respect to "knowing what you are doing",
I will admit that even after doing this for nearly
20 years I can *still* run into color results in
IDL courses that completely baffle me and which I
can't explain. Coyote and his tricks *always* comes
to the rescue in times that these. :-)

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")