

Folks,

Late last night I began pulling on a tiny little piece of thread, and before I knew it nearly every graphics program in the Coyote Library had been changed!

The issue I wanted to fix was one that on occasion can make Coyote Graphics routines much slower than normal:

http://www.idlcoyote.com/cg_tips/slowcoyote.php

Basically, this is caused by the way Coyote Graphics routines handle color. Most Coyote Graphics routines (e.g., `cgPlot`, `cgContour`, `cgText`) do all their drawing with color decomposition turned on. That way, they don't have to contaminate the one physical color table by loading drawing colors.

But, because thousands of IDL programmers still routinely cripple their graphics cards and force them to work in 1970 mode, I have to allow drawing colors to be specified with color indices in addition to color names (e.g., "red", "yellow", etc.)

In the past, what I have done is take a color index, like 10, and turn it into a string, "10", in `cgDefaultColor`. Then, `cgColor` (which only works with strings) turns this into a real color by using the color table values at index 10 to either construct a 24-bit value, or load that color into a color index of my choosing. (I load colors anywhere BUT color index 0 or 255.)

The upside is that this makes sure everyone gets the color they are looking for under all scenarios and in all devices. The downside is that it requires trips though both `cgDefaultColor` and `cgColor` every time I need to "convert" a byte color index value to a color I can work with. In normal operation, this is not a problem. It can be a problem, however, if you are calling a routine like `cgPlotS` or `cgColorFill` hundreds or thousands of times in a tight loop!

The "typical" way to handle this, since there are

only 256 possible byte index values, is to convert those values with `cgColor`, and use the colors `cgColor` produces as a look-up table for your byte values:

```
colors = LargeByteArray
theseColors = cgColor(Bindgen(256))
drawingColors = theseColors[colors]
```

Then, use the drawing colors in your loop:

```
FOR j=0, LargeNumber DO cgColorFill, ..., Color = drawingColors[j]
```

This was what I was trying to speed up last night. (Well, is 2AM last night? It was dark anyway, when I woke up thinking about this!)

My idea was to modify `cgColor` and `cgDefaultColor` so that they would let byte, integer, and some long values "fall through" the code, without converting them to strings. But, of course, this only works if you are using indexed color. If you are using decomposed color, I have to do the conversion to a long integer anyway.

So, OK, I made those changes about 2:30AM. Then, all hell broke loose!

Some of my most basic programs, like `cgPlot` and `cgContour`, no longer worked correctly! Yikes!

After poking around for another half hour, I discovered the problem was a boneheaded error by Coyote! (I should have suspected this sooner!) In fixing a previous problem with the PostScript device he implemented a fix in `cgDefaultColor` which clearly indicated an abysmal lack of programming skills on his part. So, ..., I spent another hour removing all that crap from Coyote Library routines.

The result, this morning, is 18 changed Coyote Graphics routines, which I am hoping (please, God!) work correctly. This makes me EXCEEDINGLY nervous! I DO have back-ups! If this goes south, we can recover together.

But, it *will* speed some Coyote Graphics programs up. That much I know. In the example I was using as a test case, the original code ran in about 0.5 seconds. The first Coyote Graphics version ran in about 12 seconds, and the modified Coyote Graphics version (using `cgColorFill`, but with improved `cgDefaultColor` and `cgColor` programs, ran

in just under four seconds. Simply replacing cgColorFill with PolyFill (to avoid even short trips though cgDefaultColor and cgColor), but modifying the drawing colors in the way described above to make them device and color independent, ran in about 0.7 seconds, which is perfectly acceptable to me.

What I don't know are the unknown unknowns. Those scare me. In my experience, it is dangerous to make changes to well-used and revered programs. This is why I don't change PSConfig for love or money. Well, that, and because the code is so awful! (My first object program, and it shows!)

So, if you have problems, let me know immediately. I am at your beck and call to fix them ASAP. I work weekends, nights, and holidays. Basically 24/7. Let me know.

Oh, and thanks for being my guinea pigs. ;-)

You can find the new programs here:

http://www.idlcoyote.com/programs/zip_files/coyoteprograms.zip

Cheers,

David

P.S. I do have one possible area of concern. cgColor has always been VERY careful not to load drawing colors at color indices 0 and 255. Doing so totally screws up your chances of making a PostScript file correctly. Now that I am allowing byte and integer values to "fall through" cgColor, I am no longer enforcing this restriction. Which means some users are going to be on their own. I'm not sure this is a Good Thing. Let me know if this is causing problems.

--

David Fanning, Ph.D.

Fanning Software Consulting, Inc.

Coyote's Guide to IDL Programming: <http://www.idlcoyote.com/>

Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: Coyote Library Update - Color Handling
Posted by [Craig Markwardt](#) on Sun, 16 Dec 2012 07:16:16 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Friday, December 14, 2012 10:40:55 AM UTC-5, David Fanning wrote:

> Folks,
>
> Late last night I began pulling on a tiny little piece
> of thread, and before I knew it nearly every graphics
> program in the Coyote Library had been changed!
> ...

So I see that your retirement from IDL has allowed you to get a lot more IDL work done! :-J

Craig

Subject: Re: Coyote Library Update - Color Handling
Posted by [David Fanning](#) on Sun, 16 Dec 2012 07:33:57 GMT
[View Forum Message](#) <> [Reply to Message](#)

Craig Markwardt writes:

> So I see that your retirement from IDL has allowed you to get a lot more IDL work done! :-J

Alas! :-(

Cheers,

David

P.S. Let's just say bad investment advice from Coyote
has put off retirement yet again. Sigh...

--

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