Subject: Comparing Speed of Graphical Output
Posted by David Fanning on Sat, 05 Nov 2011 17:29:04 GMT
View Forum Message <> Reply to Message

Folks,

Mark Piper has posted an interesting article on his blog
this week about speeding up the function graphics system:

http://idldatapoint.com

The comments are especially interesting and Mark shows
a interesting way to speed up the new function graphics
system dramatically using a polyline solution. (Forget
for the moment that this solution doesn't produce the
same graphic as the slower solution, and that it
produces a different, different solution every time it
runs! I presume he will get this to work, eventually.)

My interest in the article was to compare the Coyote
Graphics system to these other systems. The values
that I obtain from running the slightly modified program
on Mark's page produces timings that vary a good deal,
but here is a typical run of the program:

using direct graphics took 0.0060000420 seconds
using coyote graphics took 0.15000010 seconds
using new graphics took 12.226000 seconds
using new graphics (refresh disabled) took 7.1469998 seconds
same as above with polyline took 0.34099984 seconds

Generally speaking the Coyote Graphics routines are
about 10 times slower than the direct graphics routines
(I am showing a particular slow run here).

In some preliminary testing, I have discovered that this
is almost always due to the amount of code required to
make the Coyote Graphics routines backward compatible
with the comparable direct graphics routines. In other words,
code to make then work in indexed color mode using color
index numbers to specify colors.

In the next generation of these routines, I have been thinking
about eliminating this type of backward compatibility and make
(as the function graphics system does) all colors specified by
either a color name or by a color triple.

Now, I realize that waiting a hundredth of a second or two-tenths

of a second doesn't seem like much in the grand scheme of things (you can't exactly get a cup of coffee while you wait, as you can with the function graphics system), but there is no reason the Coyote Graphics system can't be exactly as fast as the direct graphics system.

What do you think? For those of you using the Coyote Graphics system, is compatibility an issue for you?

Cheers,

David

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