Subject: Object Graphics Fonts
Posted by jamiesmyth_uni on Mon, 03 May 2004 21:54:47 GMT
View Forum Message <> Reply to Message

Hi,

Does anyone know if it is possible to use postscript fonts in direct/object graphics? I have been asked by an editor to change the text of my figures to match the book text and I am at a bit of a loss...

Thanks,
Jamie

Subject: Re: Object Graphics fonts
Posted by David Fanning on Mon, 07 Jun 2004 19:56:40 GMT
View Forum Message <> Reply to Message

Michael Wallace writes:

> Okay, time for another question from the Object Graphics novice. I have
> managed to create some simple 2D histograms which look pretty good,
> except I'm not too pleased with the fonts.
>
> First, how do I change the fonts of the labels on an axis? I can easily
> change the font of the text annotation for the axis, but I can't seem to
> figure out how to easily change the actual labels.

Once you create the axis, you can use the GetProperty method and the TickText keyword to get the text objects used to create the axis annotations. You are free to manipulate these text objects anyway you like. I'm sure you could rotate them upside down and stand them on their heads, if you like. :-)

>
> Second, while fonts running horizontally look fine (mostly), fonts
> running vertically look like crap. Is there any way to make a vertical
> font, say for a Y axis annotation, look like the X axis equivalent but
> rotated? Even the horizontal fonts don't look that great if the font
> size is too small. Is there any way to remedy this?

Are you not using IDL 6.0? Fonts there look very nice. I can see why you are complaining if you are using a previous version of IDL. You can certainly rotate Y axis fonts if that's what you want to do. All kinds of text manipulation keywords are available. One usually uses the "random change" method until you understand which

keywords to use and how.

> Thirdly, I thought I might do better by using PostScript.

What could possibly make you think that!?

PostScript is wonderful if you plan to print something.
But unless you have one of those antique Next computers,
there is absolutely no advantage to using PostScript for
anything else.

> This might be a dumb question, but if I were to generate a PostScript
> file and then use a utility such as the convert program, would the
> resulting bitmap (e.g. PNG, GIF) look better than the version IDL would
> create by using Write_PNG or Write_GIF with the data from the
> IDLgrWindow or IDLgrBuffer? I guess I could try this out and see if it
> looks any better, but I'd rather not figure out how to get PostScript to
> work if it doesn't make any improvement.

Don't waste your time. "Looking good" is a matter of how much
resolution you have. You computer has about 75 pixels per inch
PostScript has anywhere from 300-1200 pixels per inch. Nothing
you display on the computer is going to look as good as something
you print. End of story. :-)

(Of course you might have one of those 21 inch LCD flat-panel
monitors I've been drooling over lately. Images look pretty good
on *those*!)
>
> I guess I should have said this at the top, but my specific problem is
> that I need to create plots which will be viewable online and so need to
> be a GIF, PNG, JPEG or the like. Right now, my code is creating an
> IDLgrBuffer, drawing everything, reading the data from the buffer and
> sending it on to Write_PNG. It's working without any problem, but some
> fonts (vertical, small horizontal) are hard to read. I was hoping that
> I could either figure out a better way to handle the IDLgrFonts or save
> the files as something like EPS and then convert them.

Well, have you tried making your buffer bigger and Congriding it down
for the JPEGs? That might improve resolution some, especially if you
are using the older polygon fonts.

Cheers,

David

--

David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Object Graphics fonts
Posted by Michael Wallace on Mon, 07 Jun 2004 21:15:29 GMT
View Forum Message <> Reply to Message

> Once you create the axis, you can use the GetProperty method
> and the TickText keyword to get the text objects used to
> create the axis annotations. You are free to manipulate these
> text objects anyway you like. I'm sure you could rotate them
> upside down and stand them on their heads, if you like. :-)

Some things are so easy when you know what to do. This works beautifully. Thanks, David!

> Are you not using IDL 6.0? Fonts there look very nice.
> I can see why you are complaining if you are using a
> previous version of IDL. You can certainly rotate
> Y axis fonts if that's what you want to do. All kinds of
> text manipulation keywords are available. One usually
> uses the "random change" method until you understand which
> keywords to use and how.

I am running IDL 6.0, but I was trying to use small fonts to maximize the amount of space in the plot for the important stuff, i.e. the data. I bit the proverbial bullet and made my fonts a little larger and sacrificed some plot size and things look good.

>> This might be a dumb question, but if I were to generate a PostScript
>> file and then use a utility such as the convert program, would the
>> resulting bitmap (e.g. PNG, GIF) look better than the version IDL would
>> create by using Write_PNG or Write_GIF with the data from the
>> IDLgrWindow or IDLgrBuffer? I guess I could try this out and see if it
>> looks any better, but I'd rather not figure out how to get PostScript to
>> work if it doesn't make any improvement.
>
>
> Don't waste your time. "Looking good" is a matter of how much
> resolution you have. You computer has about 75 pixels per inch
> PostScript has anywhere from 300-1200 pixels per inch. Nothing
> you display on the computer is going to look as good as something
> you print. End of story. :-)

Yep. I always seem to forget that little tidbit that it depends on resolution.

> (Of course you might have one of those 21 inch LCD flat-panel
> monitors I've been drooling over lately. Images look pretty good
> on *those*!)

Actually, I'd prefer dual flat-panel monitors. ;-) I wish we could afford two 21-inch monitors, but for now, I'll settle for something a little smaller, just as long as there are two of them.

> Well, have you tried making your buffer bigger and Congriding it down
> for the JPEGs? That might improve resolution some, especially if you
> are using the older polygon fonts.

Ah, the old make-um-big trick I've employed so many times in direct graphics land. I did try that just to see what would happen and things did look a little better in terms of letter shape, but the letters were part black and part gray due to the rebin. Whatever. I made the fonts bigger anyway, so this isn't an issue.

-Mike

Subject: Re: Object Graphics fonts
Posted by Rick Towler on Mon, 07 Jun 2004 21:36:00 GMT
View Forum Message <> Reply to Message

"David Fanning" wrote ...
> Michael Wallace writes:
>
>>
>> I guess I should have said this at the top, but my specific problem is
>> that I need to create plots which will be viewable online and so need to
>> be a GIF, PNG, JPEG or the like. Right now, my code is creating an
>> IDLgrBuffer, drawing everything, reading the data from the buffer and
>> sending it on to Write_PNG. It's working without any problem, but some
>> fonts (vertical, small horizontal) are hard to read. I was hoping that
>> I could either figure out a better way to handle the IDLgrFonts or save
>> the files as something like EPS and then convert them.
>
> Well, have you tried making your buffer bigger and Congriding it down
> for the JPEGs? That might improve resolution some, especially if you
> are using the older polygon fonts.

I have found that rendering to the screen provides far better results than rendering to IDLgrBuffer (IDLgrBuffers are rendered using the software renderer). And better graphics adapters will do the anti-aliasing for you (check your graphics adapter settings). Render to the screen, read the window, and save as PNG.

-Rick

Subject: Re: Object Graphics fonts
Posted by Michael Wallace on Mon, 07 Jun 2004 22:12:59 GMT
View Forum Message <> Reply to Message

> I have found that rendering to the screen provides far better results than
> rendering to IDLgrBuffer (IDLgrBuffers are rendered using the software
> renderer). And better graphics adapters will do the anti-aliasing for you
> (check your graphics adapter settings). Render to the screen, read the
> window, and save as PNG.

Hmmm... interesting. Things do look better using and IDLgrWindow, however is it possible to use the hardware rendering without a display screen? This plot will be part of an automated process, so there will never been a screen available. I remember that Direct Graphics will bomb out and complain about not having a display if you try to create a graphics window. I assumed that an Object Graphics window would also complain about not having a display.

Let's test this theory.... time passes .... Hmmm, it seems that creating an object graphics window without a display available causes it to just sit there ... no error is coming back, but the program is stuck at the Obj_New('IDLgrWindow').

Of course the fallback plan is to start up Xvfb and let IDL create its windows there, unless someone knows a better solution.

-Mike

Subject: Re: Object Graphics fonts
Posted by Michael Wallace on Tue, 08 Jun 2004 02:59:53 GMT
View Forum Message <> Reply to Message

>> Second, while fonts running horizontally look fine (mostly), fonts
>> running vertically look like crap. Is there any way to make a vertical
>> font, say for a Y axis annotation, look like the X axis equivalent but
>> rotated? Even the horizontal fonts don't look that great if the font
>> size is too small. Is there any way to remedy this?
>
>
> Are you not using IDL 6.0? Fonts there look very nice.
> I can see why you are complaining if you are using a
> previous version of IDL. You can certainly rotate

> Y axis fonts if that's what you want to do. All kinds of
> text manipulation keywords are available. One usually
> uses the "random change" method until you understand which
> keywords to use and how.

I have found part of my problem and have devised a solution. The object graphics fonts in IDL 6.0 do look good, however they won't necessarily look good automatically. At least not for me and my particular setup. I wouldn't rule out the possibility that my install or configuration might be to blame. In any case, I have seen some strange behavior that has left me scratching my head.

Maybe someone out there can explain this. It seemed that the appearance of a Y axis annotation would change depending on the size of the strings in the text labels. For example, when one of my plots had a yrange of 0 - 50, the Y axis text looked really good. However, if I made a plot where the data range was past 100, the position of the Y axis would get pushed over as expected. However, the rendering of the text would degrade tremendously. The thing that I couldn't understand was why the text would change in appearance just because it was rendered in a difference location.

Whatever the case, I am now adding the axis annotations directly to the plot instead of to the axis. The fonts look great and they don't change on me. Of course, I had to go through several minutes of pain trying to get BASELINE and UPDIR figured out. Eventually, the "random change" approach yielded some good results. ;-)

-Mike

Subject: Re: Object Graphics fonts
Posted by David Fanning on Tue, 08 Jun 2004 03:18:58 GMT
View Forum Message <> Reply to Message

Michael Wallace writes:

> I have found part of my problem and have devised a solution. The object
> graphics fonts in IDL 6.0 do look good, however they won't necessarily
> look good automatically. At least not for me and my particular setup.
> I wouldn't rule out the possibility that my install or configuration
> might be to blame. In any case, I have seen some strange behavior that
> has left me scratching my head.

I seriously doubt installation or configuration issues.
More likely (*much* more likely!) just object graphics programming problems.

> Maybe someone out there can explain this. It seemed that the appearance
> of a Y axis annotation would change depending on the size of the strings
> in the text labels. For example, when one of my plots had a yrange of 0
> - 50, the Y axis text looked really good. However, if I made a plot
> where the data range was past 100, the position of the Y axis would get
> pushed over as expected. However, the rendering of the text would
> degrade tremendously. The thing that I couldn't understand was why the
> text would change in appearance just because it was rendered in a
> difference location.

Ah, I think what has been happening to you is that
you were changing the scale of the Y axis after you
created it. But you weren't recomputing new text properties
after doing so. This could indeed make the text appear "ugly".
I recommend you set the RECOMPUTE_DIMENSION keyword to 2 on
all of your text objects (including those you get from the axis
itself with TICKTEXT). That should solve a lot of problems for
you. :-)

> Whatever the case, I am now adding the axis annotations directly to the
> plot instead of to the axis. The fonts look great and they don't change
> on me. Of course, I had to go through several minutes of pain trying to
> get BASELINE and UPDIR figured out. Eventually, the "random change"
> approach yielded some good results. ;-)

It always does if you have enough time and you can actually
see *something* on the display. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Object Graphics fonts
Posted by Rick Towler on Tue, 08 Jun 2004 16:41:09 GMT
View Forum Message <> Reply to Message

"Michael Wallace" wrote...
>> I have found that rendering to the screen provides far better results
than
>> rendering to IDLgrBuffer (IDLgrBuffers are rendered using the software
>> renderer). And better graphics adapters will do the anti-aliasing for

you
>> (check your graphics adapter settings). Render to the screen, read the
>> window, and save as PNG.
>
> Hmmm... interesting. Things do look better using and IDLgrWindow,
> however is it possible to use the hardware rendering without a display
> screen? This plot will be part of an automated process, so there will
> never been a screen available.

That's a stickler... As you found out you do need some sort of display.

> Of course the fallback plan is to start up Xvfb and let IDL create its
> windows there, unless someone knows a better solution.

This would only work if IDL was able to use the hardware renderer with Xvfb and I don't think it can (just a guess). You can also configure VNC as a dummy X server but I haven't done this myself and there still is the question of hardware rendering.

Unless quality and speed are paramount, I would probably render to a very large buffer and resample the image yourself (as David suggested). Probbaly isn't worth mucking with Xvfb or VNC when you don't even know if it will work.

-Rick

Subject: Re: Object Graphics fonts
Posted by Karl Schultz on Tue, 08 Jun 2004 21:15:25 GMT
View Forum Message <> Reply to Message

"Rick Towler" <tsehai@comcast.net> wrote in message
news:F_lxc.57351$3x.8757@attbi_s54...
>
> "Michael Wallace" wrote...
>>> I have found that rendering to the screen provides far better results
> than
>>> rendering to IDLgrBuffer (IDLgrBuffers are rendered using the software
>>> renderer). And better graphics adapters will do the anti-aliasing for
> you
>>> (check your graphics adapter settings). Render to the screen, read
the
>>> window, and save as PNG.
>>
>> Hmmm... interesting. Things do look better using and IDLgrWindow,

>> however is it possible to use the hardware rendering without a display
>> screen? This plot will be part of an automated process, so there will
>> never been a screen available.

Is it possible that using IDLgrWindow is giving you better results only
because the dimensions are larger? The default IDLgrBuffer size is 640x480.

Aside from the frame buffer size difference, I don't see how hardware
rendering would produce a better display, where text is concerned. I could
see how turning on line or polygon or even full-screen AA might help lines
and polygons, but text is now drawn with texture maps and I'm not sure that
graphics cards should be messing with texture contents other than performing
the usual interpolations.

Anyway-

IDL 6.0 was the first release with the FreeType-based texture mapping
rendering for IDLgrText. We've made a few bug fixes and a lot of
improvements in this area for IDL 6.1. Without an exact testcase I can use
to compare 6.0 and 6.1 text rendering, I can't tell if Michael's specific
concerns are addressed or not. I looked through the bug database and found
that I fixed a problem that sounds very much what Michael is describing.
The problem would occur when the model space was scaled unequally and a
non-default baseline and updir was used. In any event, 6.1 may address the
problem.

Karl

Subject: Re: Object Graphics fonts
Posted by Rick Towler on Wed, 09 Jun 2004 18:42:16 GMT
View Forum Message <> Reply to Message

"Karl Schultz" wrote...
>
> "Rick Towler" wrote...
>>
>> "Michael Wallace" wrote...
>>>> I have found that rendering to the screen provides far better
>>>> results than rendering to IDLgrBuffer...
>>>
>>> Hmmm... interesting. Things do look better using and IDLgrWindow,
>>> however is it possible to use the hardware rendering without a display
>>> screen? This plot will be part of an automated process, so there will
>>> never been a screen available.
>
> Is it possible that using IDLgrWindow is giving you better results only
> because the dimensions are larger? The default IDLgrBuffer size is

640x480.

> Aside from the frame buffer size difference, I don't see how hardware
> rendering would produce a better display, where text is concerned. I
could
> see how turning on line or polygon or even full-screen AA might help lines
> and polygons, but text is now drawn with texture maps and I'm not sure
that
> graphics cards should be messing with texture contents other than
performing
> the usual interpolations.

My comment was more of a general statement of image quality, not specifically text quality. A little bit off topic I suppose. But I am not totally off, pre-6.0 text benefits greatly from FSAA.

And yes, I do use FSAA (big improvement in image quality) so in the future I will qualify my statements :).

-Rick

Subject: Re: Object Graphics fonts
Posted by Michael Wallace on Wed, 09 Jun 2004 19:46:02 GMT
View Forum Message <> Reply to Message

> Is it possible that using IDLgrWindow is giving you better results only
> because the dimensions are larger? The default IDLgrBuffer size is 640x480.
>
> Aside from the frame buffer size difference, I don't see how hardware
> rendering would produce a better display, where text is concerned. I could
> see how turning on line or polygon or even full-screen AA might help lines
> and polygons, but text is now drawn with texture maps and I'm not sure that
> graphics cards should be messing with texture contents other than performing
> the usual interpolations.

Well, maybe I spoke too quickly. Using an IDLgrWindow gives me different results than using and IDLgrBuffer. The fonts look good in both, but the fonts are larger in the IDLgrWindow than they are in the IDLgrBuffer. I did make sure to have the DIMENSIONS keyword set to the same value for both of my tests. If I look at the differences between the two images, there are very subtle changes in pixel position (usually just one pixel off) for an axis or other lines, but there is a definite difference in the font size. This is why I made my previous comment about the fonts looking "better" in the IDLgrWindow... the fonts were slightly bigger and so they appeared cleaner. I didn't realize that the size of the fonts was just slightly larger.

It does seem odd that only changing one line of code to use an
IDLgrBuffer instead of an IDLgrWindow would cause these little one pixel
differences and a slight font size difference. Is there an explanation
for this or does this get chalked up to the mystery that is IDL?

> IDL 6.0 was the first release with the FreeType-based texture mapping
> rendering for IDLgrText. We've made a few bug fixes and a lot of
> improvements in this area for IDL 6.1. Without an exact testcase I can use
> to compare 6.0 and 6.1 text rendering, I can't tell if Michael's specific
> concerns are addressed or not. I looked through the bug database and found
> that I fixed a problem that sounds very much what Michael is describing.
> The problem would occur when the model space was scaled unequally and a
> non-default baseline and updir was used. In any event, 6.1 may address the
> problem.

Well, since the fonts I was concerned with were Y axis fonts, they would
have non-standard UPDIR and BASELINE. And in my code I would usually
create the axis in one command and then use SetProperty to set the text
of the axis. I don't know enough to know if anything I've done has
caused the model space to scale unequally or not.

Speaking of 6.1, when will be able to get our grubby little hands on it?
 ;-)

-Mike

Subject: Re: Object Graphics fonts
Posted by David Fanning on Wed, 09 Jun 2004 20:06:30 GMT
View Forum Message <> Reply to Message

Michael Wallace writes:

> It does seem odd that only changing one line of code to use an
> IDLgrBuffer instead of an IDLgrWindow would cause these little one pixel
> differences and a slight font size difference. Is there an explanation
> for this or does this get chalked up to the mystery that is IDL?

This mystery has a long history in IDL. For example, the Z-graphics
buffer and an IDL window of the same size has different size fonts
and axis placement. But not too much. Just enough to think you
probably shouldn't have started drinking so early in the day. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Object Graphics fonts
Posted by Michael Wallace on Wed, 09 Jun 2004 20:15:37 GMT
View Forum Message <> Reply to Message

>> Maybe someone out there can explain this. It seemed that the appearance
>> of a Y axis annotation would change depending on the size of the strings
>> in the text labels. For example, when one of my plots had a yrange of 0
>> - 50, the Y axis text looked really good. However, if I made a plot
>> where the data range was past 100, the position of the Y axis would get
>> pushed over as expected. However, the rendering of the text would
>> degrade tremendously. The thing that I couldn't understand was why the
>> text would change in appearance just because it was rendered in a
>> difference location.
>
>
> Ah, I think what has been happening to you is that
> you were changing the scale of the Y axis after you
> created it. But you weren't recomputing new text properties
> after doing so. This could indeed make the text appear "ugly".
> I recommend you set the RECOMPUTE_DIMENSION keyword to 2 on
> all of your text objects (including those you get from the axis
> itself with TICKTEXT). That should solve a lot of problems for
> you. :-)

Thanks, David. Setting RECOMPUTE_DIMENSION didn't really seem to change
that much. The text now looks the same no matter where it gets
positioned because of the tick labels, however it doesn't look much like
the equivalent text written directly on the model. So, the mysterious
text changing has been solved, but the text still doesn't look right.
Is this due to some ordering/creation problem that may exist in my code?

Could you (or anyone else) explain a little bit about how object
graphics really work in this regard? Specifically, what gotchas are
present if a keyword is set in the object instantation rather than being
set afterwards via SetProperty? What about the order of adding objects
to a model and then setting other properties of the object? For
example, say I have an IDLgrModel and add an IDLgrAxis to it. I then
choose to set something via SetProperty of the IDLgrAxis. Could this
potentially cause a different image to get drawn than if I had set all
of the properties before adding the axis to the model?

Of course, there's the issue of making sure things are added to the

model in the correct order, especially when you have an IDLgrImage (the topic of another recent thread), but are there any other ordering problems or anything else that the object graphics novice such as myself wouldn't know about? I'm sure there's plenty I don't know, but the Top 10 Object Graphics Gotchas might help me and some other aspiring object graphics programmers.

-Mike

Subject: Re: Object Graphics fonts
Posted by Karl Schultz on Wed, 09 Jun 2004 21:11:05 GMT
View Forum Message <> Reply to Message

"Michael Wallace" <mwallace.no.spam@no.spam.swri.edu.invalid> wrote in message news:10ceq7qsnl25ub3@corp.supernews.com...
>> Is it possible that using IDLgrWindow is giving you better results only
>> because the dimensions are larger? The default IDLgrBuffer size is 640x480.
>>
>> Aside from the frame buffer size difference, I don't see how hardware
>> rendering would produce a better display, where text is concerned. I could
>> see how turning on line or polygon or even full-screen AA might help lines
>> and polygons, but text is now drawn with texture maps and I'm not sure that
>> graphics cards should be messing with texture contents other than performing
>> the usual interpolations.
>
> Well, maybe I spoke too quickly. Using an IDLgrWindow gives me
> different results than using and IDLgrBuffer. The fonts look good in
> both, but the fonts are larger in the IDLgrWindow than they are in the
> IDLgrBuffer. I did make sure to have the DIMENSIONS keyword set to the
> same value for both of my tests. If I look at the differences between
> the two images, there are very subtle changes in pixel position (usually
> just one pixel off) for an axis or other lines, but there is a definite
> difference in the font size. This is why I made my previous comment
> about the fonts looking "better" in the IDLgrWindow... the fonts were
> slightly bigger and so they appeared cleaner. I didn't realize that the
> size of the fonts was just slightly larger.
>
> It does seem odd that only changing one line of code to use an
> IDLgrBuffer instead of an IDLgrWindow would cause these little one pixel
> differences and a slight font size difference. Is there an explanation
> for this or does this get chalked up to the mystery that is IDL?

The explanation is that the devices have different resolutions:

IDL> oWin = OBJ_NEW('IDLgrWindow')
IDL> oWin->GetProperty, Resolution=winres

IDL> print, winres

0.026437500 0.026416666

IDL> oBuff = OBJ_NEW('IDLgrBuffer')

IDL> oBuff->GetProperty, Resolution=buffres

IDL> print, buffres

0.035277778 0.035277778

IDL does its best to draw the glyphs at the requested size. A common size is 12 points and that does not mean 12 pixels, although on most displays the size of a pixel is pretty close to a point (1/72nd of an inch). IDL uses the device resolution (size of a pixel) to decide how many pixels to use to draw the glyphs. If the resolutions are different, then the number of pixels used will be different, in order to get the same physical size.

If your window pixel size is smaller than the buffer pixel size, as is the case above, then it would take more pixels to draw the same size glyph. That's why the text looked better in the window. Bitmap glyph rendering systems like FreeType do a much better job with more pixels in the glyph box. When you get down to really small boxes like 4 or 5 pixels high, the glyph quality really goes down.

You can set the RESOLUTION property of IDLgrBuffer to match that of the window.

Also, IDL does its best to get an accurate value for the resolution of the display, but some display driver interfaces don't return very accurate values and there's a lot of potential for inexact values. But they are pretty close. Printers tend to be more accurate in this area. And people don't tend to put their rulers up to the screen.
>> IDL 6.0 was the first release with the FreeType-based texture mapping
>> rendering for IDLgrText. We've made a few bug fixes and a lot of
>> improvements in this area for IDL 6.1. Without an exact testcase I can use
>> to compare 6.0 and 6.1 text rendering, I can't tell if Michael's specific
>> concerns are addressed or not. I looked through the bug database and found

>> that I fixed a problem that sounds very much what Michael is describing.
>> The problem would occur when the model space was scaled unequally and a
>> non-default baseline and updir was used. In any event, 6.1 may address
the
>> problem.
>
> Well, since the fonts I was concerned with were Y axis fonts, they would
> have non-standard UPDIR and BASELINE. And in my code I would usually
> create the axis in one command and then use SetProperty to set the text
> of the axis. I don't know enough to know if anything I've done has
> caused the model space to scale unequally or not.
>
> Speaking of 6.1, when will be able to get our grubby little hands on it?
> ;-)

The beta program is almost over, so real soon now. But I'm not sure all the
bug fixes mentioned above are in the beta code.

Karl

Subject: Re: Object Graphics fonts
Posted by Karl Schultz on Wed, 09 Jun 2004 21:35:22 GMT
View Forum Message <> Reply to Message

"Michael Wallace" <mwallace.no.spam@no.spam.swri.edu.invalid> wrote in
message news:10cervakueidj58@corp.supernews.com...
>>> Maybe someone out there can explain this. It seemed that the appearance
>>> of a Y axis annotation would change depending on the size of the strings
>>> in the text labels. For example, when one of my plots had a yrange of 0
>>> - 50, the Y axis text looked really good. However, if I made a plot
>>> where the data range was past 100, the position of the Y axis would get
>>> pushed over as expected. However, the rendering of the text would
>>> degrade tremendously. The thing that I couldn't understand was why the
>>> text would change in appearance just because it was rendered in a
>>> difference location.

I think I explained this before. It isn't the location, it is the scaling
of your model space, and that's what the bug (fixed for 6.1) was about.
Actually, I can't be certain because I can't tell from your post if you
changed the range from [0, 50] to [0, 100] or to [50,100]. If the former,
then the size of the range has changed, and that may have been enough to
cause the quality problem.

>> Ah, I think what has been happening to you is that
>> you were changing the scale of the Y axis after you
>> created it. But you weren't recomputing new text properties

>> after doing so. This could indeed make the text appear "ugly".
>> I recommend you set the RECOMPUTE_DIMENSION keyword to 2 on
>> all of your text objects (including those you get from the axis
>> itself with TICKTEXT). That should solve a lot of problems for
>> you. :-)
>
> Thanks, David. Setting RECOMPUTE_DIMENSION didn't really seem to change
> that much. The text now looks the same no matter where it gets
> positioned because of the tick labels, however it doesn't look much like
> the equivalent text written directly on the model. So, the mysterious
> text changing has been solved, but the text still doesn't look right.
> Is this due to some ordering/creation problem that may exist in my code?

Probably not.

One thing you can try is setting the RENDER_METHOD property to 1 on your
axis text objects. The bug I mentioned above had to do with computing the
size of the glyph box that Freetype uses to render the glyph. If the box is
miscalculated to be too small, the glyph quality will decline. Setting
RENDER_METHOD to 1 goes back to the pre-6.0 method of tessellating the glyph
outlines into a lot of small triangles and drawing those. This method had
its own problems, the biggest being naive anti-aliasing of the glyph
outlines. You can just try it to see how it looks.

> Could you (or anyone else) explain a little bit about how object
> graphics really work in this regard? Specifically, what gotchas are
> present if a keyword is set in the object instantation rather than being
> set afterwards via SetProperty?

Should be none. Any property interactions should be documented.

> What about the order of adding objects
> to a model and then setting other properties of the object? For
> example, say I have an IDLgrModel and add an IDLgrAxis to it. I then
> choose to set something via SetProperty of the IDLgrAxis. Could this
> potentially cause a different image to get drawn than if I had set all
> of the properties before adding the axis to the model?

No.

> Of course, there's the issue of making sure things are added to the
> model in the correct order, especially when you have an IDLgrImage (the
> topic of another recent thread),

Right, the drawing order depends on the order of the objects in the Model.

Note that there are IDL_Container (a superclass of IDLgrModel) methods that
let you move/shuffle objects around in a model. Also, the IDLgrModel::Add

method has a POSITION kw that lets you insert objects into the model in places other than the end. So, it isn't strictly the order in which things are added to the model that determines drawing order, but the final order of the objects in the model at the time of the draw.

> but are there any other ordering
> problems or anything else that the object graphics novice such as myself
> wouldn't know about?

Transparency - there are other threads about this, but if you have specific questions, ask.

> I'm sure there's plenty I don't know, but the Top
> 10 Object Graphics Gotchas might help me and some other aspiring object
> graphics programmers.
>
> -Mike

Subject: Re: Object Graphics fonts
Posted by David Fanning on Wed, 09 Jun 2004 21:51:52 GMT
View Forum Message <> Reply to Message

Michael Wallace writes:

> Could you (or anyone else) explain a little bit about how object
> graphics really work in this regard?

Ah, well, Karl is probably a better source than I am.

I will say this. I don't think you can learn everything you need to know about object graphics by reading the IDL documentation. :-)

I think you need to get a more informative source. I've found Computer Graphics: Using Open GL by F.S. Hill, Jr. to be useful. Sometimes it just helps to be a little more familiar with the vocabulary.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Object Graphics fonts
Posted by Michael Wallace on Thu, 10 Jun 2004 16:01:46 GMT
View Forum Message <> Reply to Message

> The explanation is that the devices have different resolutions:
>
> IDL does its best to draw the glyphs at the requested size. A common size
> is 12 points and that does not mean 12 pixels, although on most displays the
> size of a pixel is pretty close to a point (1/72nd of an inch). IDL uses
> the device resolution (size of a pixel) to decide how many pixels to use to
> draw the glyphs. If the resolutions are different, then the number of
> pixels used will be different, in order to get the same physical size.

Oh, I get it now. No matter how long I work in this profession, it
seems I always have to relearn that points != pixels, although usually
points ~= pixels. For some reason, my little ol' head just can't seem
to remember details like this. Maybe it's because I only try to call up
that knowledge when I need it, which is not very often.

> If your window pixel size is smaller than the buffer pixel size, as is the
> case above, then it would take more pixels to draw the same size glyph.
> That's why the text looked better in the window.

Yep, my window pixel size was smaller. Okay, this mystery is solved, at
least for me. I don't know if anyone else got anything out of this
discussion, but I feel better knowing that I can move one IDL nuance
from the "mystery" pile to the "makes sense" pile. :-)

-Mike

Subject: Re: Object Graphics fonts
Posted by David Fanning on Thu, 10 Jun 2004 16:07:59 GMT
View Forum Message <> Reply to Message

Michael Wallace writes:

> Yep, my window pixel size was smaller. Okay, this mystery is solved, at
> least for me. I don't know if anyone else got anything out of this
> discussion, but I feel better knowing that I can move one IDL nuance
> from the "mystery" pile to the "makes sense" pile. :-)

Whew! And if we can all do this once or twice a month,
we'll all be experts by Christmas!! :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/