Subject: Re: pixmap drawables in Object Graphics?
Posted by David Fanning on Tue, 18 Dec 2001 00:54:34 GMT
View Forum Message <> Reply to Message

Martin Downing (martin.downing@ntlworld.com) writes:

> This is similar to previous queries on object graphics and pixmaps but I
> would appreciate running this by the experts.
> I am writing a program to fit projections of a 3d surface model to its
> silhouette in an image (e.g. a radiograph). This method allows an estimate
> of object position to be recovered from the knowledge of the object shape
> and the image. My 3d data is a triangulated mesh which can be best stored as
> a IDLgrPolygon object. This is attractive as you can then easily specify a
> graphics model to render the object at specific rotations, and projections
> of complicated polygon objects can then be drawn rapidly using OpenGL.
>
> However, as this is part of a fitting process, I then read the drawable back
> into an image buffer using say tvrd(), do some image processing to get a
> goodness of fit quantity and repeat until a sufficient fit is found.

Well, I don't think you are going to be doing
any TVRDing in object graphics windows. :-)

The TVRD equivalent in object graphics is probably
the READ method on a window object, but that returns
an image object (with 24-bit image data). Not the sort
of thing you will be doing a lot of image processing
on, probably.

> I do
> not need to see each projection in an exposed draw widget, but as far as I
> can gather, pixmaps are not implemented in object graphics.

The IDLgrBuffer object is the object graphics equivalent
of a pixmap. But, again, this is no 2D graphics window of
the sort you seem to expect. *Everything* in object graphics
is 3D. The object graphics system *is* a 3D system. That is
the point of it.

> So as I see it,
> my only option using object graphics is to use normal draw widgets, which
> seems like overkill.

I'm not sure I understand this statement. Object graphics
and "normal" draw widgets are mutually exclusive. You can
use one or the other, not both. Some things (rendering
complex polygons come to mind) are perfect for object
graphics. Other things (say, working with 2D images) often

work better in direct graphics windows. Which you use depends
entirely on what makes sense.

Here, it might make sense to have overlapping widget hierarchies
with both object and normal draw widgets mapped into the same
real estate in your GUI. Then, depending upon what you wish
to display, you can choose one window or the other.

Come to think of it, that might be a good excuse
to write a combination window compound widget. Of
course, it should be written as an object. :-)

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: pixmap drawables in Object Graphics?
Posted by Martin Downing on Tue, 18 Dec 2001 11:57:48 GMT
View Forum Message <> Reply to Message

"David Fanning" <david@dfanning.com> wrote in message
news:MPG.168841b7b95a011b9897af@news.frii.com...
>> into an image buffer using say tvrd(), do some image processing to get a
>> goodness of fit quantity and repeat until a sufficient fit is found.
>
> Well, I don't think you are going to be doing
> any TVRDing in object graphics windows. :-)
>
> The TVRD equivalent in object graphics is probably
> the READ method on a window object, but that returns
> an image object (with 24-bit image data). Not the sort
> of thing you will be doing a lot of image processing
> on, probably.

Hi David,

Its a while since I've been in the depths of my few object graphics
programs, and I had forgotten about the window object :(
So that's right, what I am thinking of is

- Create object,model, view and Graphics window
- do transformations
- Render (draw) in the object window
- then read back to an image:

```
oImage = oWindow->Read()
oImage->GetProperty, data=imtc
IDL> help, imtc
IMTC BYTE = Array[3, 950, 950]
```

This is certainly usable, ok its true color, but for what I'm interested in
is a binary rendering of the object which is quite easy to produce:

```
im = imtc[0,*,*] gt 0
```

>> can gather, pixmaps are not implemented in object graphics.
>
> The IDLgrBuffer object is the object graphics equivalent
> of a pixmap. But, again, this is no 2D graphics window of
> the sort you seem to expect. *Everything* in object graphics
> is 3D. The object graphics system *is* a 3D system. That is
> the point of it.

Ah ha - IDLgrBuffer - this is *exactly* what I want! Thanks. Erm - I *do*
understand about 3d graphics tho, thats why I am doing this application ;),
still if your working on a system that renders into 3D aka CP3O put me down
for one!!!

>> seems like overkill.
>
> I'm not sure I understand this statement. Object graphics.....

I'm not surprised, not sure I do, but you answered my question anyway!

> Here, it might make sense to have overlapping widget hierarchies
> with both object and normal draw widgets mapped into the same
> real estate in your GUI. Then, depending upon what you wish
> to display, you can choose one window or the other.

Oh boy, now I'm lost too!

Thanks David, I'm now on my way. Now I need to get to grips with ViewGroup
objects so I can render multiple views, any simple examples out there?

Martin

Subject: Re: pixmap drawables in Object Graphics?
Posted by karl_schultz on Tue, 18 Dec 2001 16:05:25 GMT
View Forum Message <> Reply to Message

"Martin Downing" <martin.downing@ntlworld.com> wrote in message
news:<82wT7.22741$4e3.3004657@news6-win.server.ntlworld.com>...
> Hi all,
>
> This is similar to previous queries on object graphics and pixmaps but I
> would appreciate running this by the experts.
> I am writing a program to fit projections of a 3d surface model to its
> silhouette in an image (e.g. a radiograph). This method allows an estimate
> of object position to be recovered from the knowledge of the object shape
> and the image. My 3d data is a triangulated mesh which can be best stored as
> a IDLgrPolygon object. This is attractive as you can then easily specify a
> graphics model to render the object at specific rotations, and projections
> of complicated polygon objects can then be drawn rapidly using OpenGL.
> However, as this is part of a fitting process, I then read the drawable back
> into an image buffer using say tvrd(), do some image processing to get a
> goodness of fit quantity and repeat until a sufficient fit is found. I do
> not need to see each projection in an exposed draw widget, but as far as I
> can gather, pixmaps are not implemented in object graphics. So as I see it,
> my only option using object graphics is to use normal draw widgets, which
> seems like overkill. Is this true and does anyone else have a better idea.?

It sounds like you want to do off-screen rendering and then perform
image analysis on the result. Use IDLgrBuffer as the destination
graphics object (instead of an IDLgrWindow). Fish the pixels back out
with the IDLgrBuffer::Read method, which puts the pixels in an
IDLgrImage object. You can then get the pixels out of the IDLgrImage
object for analysis. All this can happen without drawing anything to
the screen.

Something like:

```
oBuffer = obj_new('IDLgrBuffer')
oBuffer->Draw, oView
oImage = oBuffer->Read()
oImage->GetProperty, DATA=imageData
```

What happens under the covers is that IDL renders your scene into a
completely device-independent off-screen frame buffer, using a
software renderer.

Karl

Subject: Re: pixmap drawables in Object Graphics?

Posted by Dick Jackson on Tue, 18 Dec 2001 16:25:31 GMT
View Forum Message <> Reply to Message

"Martin Downing" <martin.downing@ntlworld.com> wrote in message
news:acGT7.25951$4e3.3371804@news6-win.server.ntlworld.com.. .

> Now I need to get to grips with ViewGroup
> objects so I can render multiple views, any simple examples out there?

At the bottom of RSI/IDL55/lib/utilities/idlexobjview__define.pro there's an IDLexObjview__example routine that uses two views in a viewgroup:

```
IDL> .compile idlexobjview__define
IDL> IDLexObjview__example
```

I think it will just involve getting each view from the viewgroup and telling it to draw to the window (or buffer!) in turn.

Best of luck with that!

Cheers,
--
-Dick

Dick Jackson / dick@d-jackson.com
D-Jackson Software Consulting / http://www.d-jackson.com
Calgary, Alberta, Canada / +1-403-242-7398 / Fax: 241-7392

Subject: Re: pixmap drawables in Object Graphics?
Posted by Martin Downing on Wed, 19 Dec 2001 22:43:09 GMT
View Forum Message <> Reply to Message

"Karl Schultz" <karl_schultz@yahoo.com> wrote in message
news:e415b359.0112180805.6069b9ce@posting.google.com...
> "Martin Downing" <martin.downing@ntlworld.com> wrote in message
news:<82wT7.22741$4e3.3004657@news6-win.server.ntlworld.com>...
>> Hi all,
>>
> It sounds like you want to do off-screen rendering and then perform
> image analysis on the result. Use IDLgrBuffer as the destination
> graphics object (instead of an IDLgrWindow). Fish the pixels back out
> with the IDLgrBuffer::Read method, which puts the pixels in an
> IDLgrImage object. You can then get the pixels out of the IDLgrImage
> object for analysis. All this can happen without drawing anything to
> the screen.
>
> Something like:

>
> oBuffer = obj_new('IDLgrBuffer')
> oBuffer->Draw, oView
> oImage = oBuffer->Read()
> oImage->GetProperty, DATA=imageData
>
> What happens under the covers is that IDL renders your scene into a
> completely device-independent off-screen frame buffer, using a
> software renderer.
>
> Karl

Thanks, however although it sounds logical I tried comparisons of readinng
back from windows and buffer drawables today, and the window method was
faster!

Martin
