Subject: Re: NVidia Quadro4 980 XGL card + IDL
Posted by Karl Schultz on Fri, 03 Feb 2006 17:06:46 GMT
View Forum Message <> Reply to Message

On Thu, 02 Feb 2006 22:22:05 -0800, raval.chintan wrote:

> Karl Schultz wrote:
>> On Thu, 02 Feb 2006 00:01:57 -0800, raval.chintan wrote:
>>
>>> Dear All,
>>>
>>> I am using NVidia Quadro 4 980 XGL. One can visualize stereo images
>>> using Quad Buffer , available on this card. Proper interface
>>> (functions) for rendering images using this buffer are given in
>>> OpenGL.. Quad Buffer includes Left and Right Buffer ,which will be
>>> rendered one by one.For Visulization of images one needs, shutter
>>> glasses with infrared emitter to be connected with graphics card.
>>>
>>>
>>> Will it be possible to render an image with use of this buffer on IDL
>>> Draw widget? or in other way ,Will it be possible call this
>>> interface(functions) of OpenGL through IDL?
>>
>> This is not supported in IDL 6.2, nor will it be in 6.3. We're getting
>> enough requests for it that it may be in a release soon.
>>
>> Although it is not documented or supported, you can make OpenGL calls from
>> C-language DLM's if you do things carefully. To do what you want, you'll
>> probably have to subclass IDLgrView with IDL code where you would override
>> the Draw method. You would also write a C DLM that calls glDrawBuffer
>> with GL_BACK_RIGHT.
>>
>> By the time IDLgrView::Draw is called, the correct GL context is in
>> place and IDL has already called glDrawBuffer with GL_BACK (implies
>> GL_LEFT as well). Your new Draw method would call the C DLM code to call
>> the superclass Draw method, call the C DLM code to set GL_BACK_RIGHT, and
>> then call the superclass Draw method again.
>
>
> If i am correct with my limited knowledge of IDL, IDLgrView class does
> not have the draw
> method. IDLgrWindow , IDLgrModel,IDLgrBuffer have draw method. Now the
> question comes that which class's draw method i should override.
> IDLgrWindow , IDLgrModel or IDLgrBuffer? Where will C dlm render the
> Left and right image? Will it be on IDL's window or on C's window?

I did say you need to do it for IDLgrView. Actually, IDLgrScene would
probably be more correct, but I tend to not use Scenes or Viewgroups in

anything but the more complex programs.

You can't do this by overriding a destination (Window or Buffer) object's Draw method because you need to sneak in the glDrawBuffer call *after* IDL issued it for the destination object and before it actually starts drawing objects.

You could do it at the grModel level, but the glDrawBuffer call would not take effect until IDL reached that Model in the scene graph, so anything before that would end up in the wrong buffer.

So Scene or View is the Right Place.

The C DLM does nothing but slip in a call to glDrawBuffer at the right place. IDL contines to render as it did before, but just to the GL buffer specified in the glDrawBuffer call.

glDrawBuffer just sets GL state (the current draw buffer). It does not actually perform a Draw on a Buffer.

>
> It would be helpful if you can provide sample code or detailed
> explaination of your suggested technique.

okay....

I don't have any stereo hardware at the moment. So, I'm going to make a grView subclass that renders the scene graph twice into the same destination object, but with a slight offset the second time so that you can see the second rendering.

Notice that only because I'm doing this to the same GL buffer, I need to keep the view from erasing on the second draw, so I set the TRANSPARENT flag. You would not do this in the real stereo case because you would want to go ahead and paint the view with the view background color in the "other" (right) stereo buffer.

```
pro IDLstereoView::Draw, dest

 self->IDLgrView::Draw, dest
 self->GetProperty, VIEWPLANE_RECT=vpr, TRANSPARENT=trans
 vprsave = vpr
 vpr[0] += 0.1
 self->SetProperty, VIEWPLANE_RECT=vpr, TRANSPARENT=1
 ;; call DLM that calls glDrawBuffer(GL_BACK_RIGHT) here!
 self->IDLgrView::Draw, dest
 self->SetProperty, VIEWPLANE_RECT=vprsave, TRANSPARENT=trans
```

```
end


pro IDLstereoView__define
 struct = { IDLstereoView, $
     INHERITS IDLgrView}
end
```

test program:

```
pro test_stereo_view

oWin = OBJ_NEW('IDLgrWindow')
oView = OBJ_NEW('IDLstereoView')
oModel = OBJ_NEW('IDLgrModel')
oPolyline = OBJ_NEW('IDLgrPolyline', [[-0.9,-0.9],[0.9,0.9]])

oView->Add, oModel
oModel->Add, oPolyline

oWin->Draw, oView

end
```

All you'd have to do to try the stereo stuff is to code a DLM C procedure that does absolutely nothing else execpt:

```
glDrawBuffer(GL_BACK_RIGHT);
```

and call it from where I indicated above. You don't need to "put things back" by calling glDrawBuffer(GL_BACK_LEFT) after the second draw because IDL will do that on the next draw. But it may be good form to do so.

Oh, and you'd have to figure out the eye transform stuff (see other posts in this thread) and how to activate your hardware to go into stereo mode.

Please refer to the External Design Guide and the accompanying samples for more information on the DLM.

Karl