Subject: Re: idl obj gaphics and opengl - how similar?
Posted by David Fanning on Thu, 14 Nov 2002 23:32:59 GMT
View Forum Message <> Reply to Message

R.G. Stockwell (sorry@noemail.now) writes:

> I was just wondering...
> how similar are the IDL object graphics functions,
> and the OpenGL library?
>
> i.e. if one knows how to do objects gaphics in IDL,
> will they be (reasonably easily) be able to do similar
> things in c and opengl?

I don't really know the answer to this, because I've never tried to write an OpenGL-type of program. But I do spend quite a bit of time with my nose in an OpenGL-oriented graphics book trying to figure out what the heck is going on in object graphics.

My impression is that as low-level as IDL's object graphics API is, it is still at a higher level than OpenGL. At least I'm always thinking to myself, "Oh, I see how IDL does that!", and I'm always glad I don't have to do it myself.

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
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: idl obj gaphics and opengl - how similar?
Posted by James Kuyper on Fri, 15 Nov 2002 15:48:22 GMT
View Forum Message <> Reply to Message

"R.G. Stockwell" wrote:
>
> Hi All,
> I was just wondering...
> how similar are the IDL object graphics functions,

> and the OpenGL library?
>
> i.e. if one knows how to do objects gaphics in IDL,
> will they be (reasonably easily) be able to do similar
> things in c and opengl?

C itself doesn't have any graphics capabilities. There are several different graphics systems that have C interfaces, (and OpenGL is one of them), but you need to specify the library, not the C language, as the point of reference.

That said, I bow out: I know very little about using those libraries. I can say that IDL typically operates at a higher level than those libraries, with the typical ease-of-use vs. performance trade-offs that apply when you change levels that way.

Subject: Re: idl obj gaphics and opengl - how similar?
Posted by R.G. Stockwell on Fri, 15 Nov 2002 16:41:14 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
> R.G. Stockwell (sorry@noemail.now) writes:
>
>
>> I was just wondering...
>> how similar are the IDL object graphics functions,
>> and the OpenGL library?
>>
>> i.e. if one knows how to do objects gaphics in IDL,
>> will they be (reasonably easily) be able to do similar
>> things in c and opengl?
>
>
> I don't really know the answer to this, because
> I've never tried to write an OpenGL-type of
> program. But I do spend quite a bit of time with
> my nose in an OpenGL-oriented graphics book trying
> to figure out what the heck is going on in object
> graphics.
>
> My impression is that as low-level as IDL's object
> graphics API is, it is still at a higher level than
> OpenGL. At least I'm always thinking to myself, "Oh,
> I see how IDL does that!", and I'm always glad I don't
> have to do it myself.
>
> Cheers,

>
> David
>

Ah, openGL sounds deep and tedious. Perfect!
Your description is much as I expected, in that IDL
is similar, but higher level.

Hey, when that IDL to C++ translator is finished (from
another poster), that would be a great way to
generate C++ OpenGl code. ;)

Thanks,
bob stockwell

Subject: Re: idl obj gaphics and opengl - how similar?
Posted by Pavel A. Romashkin on Fri, 15 Nov 2002 18:11:33 GMT
View Forum Message <> Reply to Message

Now, this is no gentle humor anymore. This is plane sarcasm! :-)
BTW, if you look at a few posts by Karl, you will find some pointers at
how is OGL used in IDL OG. And I would not say I have a clue other than
"I don't want to know".
Cheers,
Pavel

"R.G. Stockwell" wrote:
>
> Hey, when that IDL to C++ translator is finished (from
> another poster), that would be a great way to
> generate C++ OpenGl code. ;)

Subject: Re: idl obj gaphics and opengl - how similar?
Posted by Karl Schultz on Fri, 15 Nov 2002 23:31:31 GMT
View Forum Message <> Reply to Message

"David Fanning" <david@dfanning.com> wrote in message
news:MPG.183de0975833db61989a1d@news.frii.com...
> R.G. Stockwell (sorry@noemail.now) writes:
>
>> I was just wondering...
>> how similar are the IDL object graphics functions,
>> and the OpenGL library?

IDL Object Graphics is built on top of OpenGL, but OG wasn't specifically designed to be similar to OpenGL. OG can theoretically be built on top of other graphics packages. It has a VRML "driver", so that is an existance proof of the possibility, although VRML is pretty similar to OpenGL anyway (most VRML viewers are written in OpenGL). The idea here is to be able to move OpenGL to another graphics system if we had to. A few years ago, the life expectancy of OpenGL on IDL platforms wasn't very optimistic. But now OpenGL is pretty solid, especially now that it is integrated into XFree86 and no other graphics package is making any bids for the cross-platform package of choice. We try to be careful about exposing too much OpenGL-specific function in IDL OG, in case we someday need to move to another package that might not have these features.

The amount of similarity between an IDL OG object and OpenGL depends on the object. The IDLgrPolyline object really doesn't do too much more than store the vertex and related data. It's Draw method basically just submits the vertex and other data to OpenGL . OTOH, an object like IDLgrSurface has tons of code to implement all that surface functionality, including stuff like Lego mode and all that. At the end of the day, it just submits lines and polygons to OpenGL, but there was a bit of work involved to come up with these lines and polygons. In other words, there is no "surface" OpenGL procedure. The IDL "surface" needs to be broken down into small convex polygons for OpenGL .

>> i.e. if one knows how to do objects gaphics in IDL,
>> will they be (reasonably easily) be able to do similar
>> things in c and opengl?

OpenGL is a very procedural graphics library and therefore doesn't really have the concept of objects. So a LOT of the code in OG is in the object implementation, which is primarily storing the graphical data and the graphic attributes related to the data. A big part of OG is also in the "Draw" code which draws all the Views, Models, and atoms in your graphics tree. These things aren't strictly necessary for a C/OpenGL program to get just something on the screen, but are needed to present a framework for more complex scenes. It is also all there so you can just call Draw to draw the entire scene without walking through all your data each time.

So, yeah, if you don't need the object structure and have fairly simple data, it isn't that hard to get an image on the screen with C/OpenGL . You have to set up your view and coordinate system, based on your data. You have to walk through your data, passing OpenGL the information on a vertex by vertex basis. Pretty easy so far. But if you want to get interactive or start drawing things like surfaces or contours, you're in for a lot of code.

> I don't really know the answer to this, because
> I've never tried to write an OpenGL-type of
> program. But I do spend quite a bit of time with

> my nose in an OpenGL-oriented graphics book trying
> to figure out what the heck is going on in object
> graphics.

Yes, this can help a lot.

> My impression is that as low-level as IDL's object
> graphics API is, it is still at a higher level than
> OpenGL. At least I'm always thinking to myself, "Oh,
> I see how IDL does that!", and I'm always glad I don't
> have to do it myself.

Right. Perhaps the biggest things that IDL OG adds to OpenGL are the "graphics tree" (Views, Models, etc), the higher function primitives, and persistant/retained nature of the objects.

Karl
