Subject: Re: IDL Objects Graphics cache and crash
Posted by karl_schultz on Mon, 11 Mar 2002 16:48:01 GMT
View Forum Message <> Reply to Message

"Mark Hadfield" <m.hadfield@niwa.co.nz> wrote in message
news:<a6ghfk$mj6$1@newsreader.mailgate.org>...
> "Altyntsev Dmitriy" <alt@iszf.irk.ru> wrote in message
> news:6b9fda50.0203100016.76b8433d@posting.google.com...
>> I am trying now to turn to IDL Objects Graphics so a pair of
>> questions to IDL gurus.
>
> None of the gurus seem to have replied yet, so I'll have a go.
>
>> In IDL online help topic "The Graphics Object Hierarchy -> The
>> Rendering Process" one can read about draw cache: "Subsequent draws
>> of this graphic atom to the same destination can then be drawn very
>> efficiently."
>
> Hey cool! I never noticed that paragraph before. (I've never noticed
> the behaviour it describes, either.)

Most Object Graphics objects create caches of graphic primitives that
are essentially vertex lists that are suitable for direct submission
to OpenGL. Especially when working with hardware accelerated cards,
it is important to submit this vertex information to OpenGL as quickly
as possible. So, we try to avoid any computations or conversions that
can get in the way of rapid submission by creating the caches. This
was perhaps more important a few years ago when CPU's were slower.

Like so many other things in the graphics world, the acutal benefit
you see will vary greatly, depending on your hardware and exactly what
your program is doing. For some objects, like IDLgrPolyline and
IDLgrPolygon, the caches are not much different than the vertex data
stored in the object. But for other objects like IDLgrText and
IDLgrSurface, there is a significant difference between the data you
store in the object and the cache contents. So, the caches are a
bigger win for these objects.

Also (important for the discussion below), IDL can build the caches
"without notice", although it generally does so only when it needs to.
Most objects wait until they are actually drawn for the first time.
Some might do it whenever the data changes. The point of the caches
is to achieve fast drawing in situations where the application is only
changing trackball transforms, for example.

>
>> But it seems not to work as declared. When I change HIDE property of
>> some graphic atom and then draw window it takes the same time as

>> first drawing. What's wrong?
>
> Probably the documentation.

It is really hard to measure stuff like this, especially with some unpredictability surrounding cache rebuilds.

I would suggest measuring the time it takes to draw your model a few hundred times. Then hide the atom in question, and then measure again. If the atom is sufficently complex, you should see a difference. If the graphic atom is something really simple, like a single polyline, you may not measure much of a difference.

>
> By the way, I find the paragraph immediately before the one you refer
> to particularly amusing--the one that begins, "The order in which
> objects are added to the hierarchy will have an impact...". It fails
> to mention the fact that the visibility of all OG atoms, except
> images, is controlled by position in 3D space, not by drawing order.

That paragraph probably assumed that the reader expected the "natural" result of depth sorting/buffering. It could be clearer. Rendering order makes the biggest difference when things are at the same depth or transparency is involved.

>
>> I tried different renderer and retain, but can not get any
>> acceleration. I use Win98, IDL 5.4, video ASUS AGP3800.
>
> Well, you've already anticipated my only suggestion, which was to try
> hardware & software renderers.

We'd need to learn more about why he can't get the expected acceleration. A lot also depends on the OpenGL implementation for that card. Sometimes, the software implementation beats the "hardware" implementation if the OpenGL implementation is poor. One quick way to get a rough feel for this OpenGL implementation is to run some other OpenGL application or demo and see how it compares. Or, just see what the vendor claims about this product. A lot of times, the vendor won't supply an OpenGL driver and so the system ends up using the Microsoft OpenGL, which is a mostly software implemenation.

>
>> Using of IDLgrROI sometimes crashes IDL. It is very unpleasant. May
>> be anyone can advice how avoid these crashes and other OG bugs if
>> any. I use Win98, IDL 5.4
>
> Sorry, I have no suggestion on this specific bug. Can you upgrade to

> IDL 5.5? With object graphics, later is generally better. Do the
> crashes occur with the software renderer? IDL display problems are
> sometimes caused by the video driver and the hardware renderer
> exercises this driver much more than the software one. And of course
> upgrading Windows to one of the NT family (2000, XP) is generally a
> good idea if you want stability.

The IDLgrROI object did have a few bugs in 5.4 (fixed in 5.5), many of which can be worked around if we know enough about the problem. For example, avoiding setting the vertex data in IDLgrROI with SetProperty (only set the vertex data at Init time) helps avoid many of the problems. This means that you'd have to destroy and recreate the object if you wanted to change the data in it, but at least that's a workaround that could help until upgrading to 5.5.

Karl
