Subject: Re: IDL Objects Graphics cache and crash
Posted by karl_schultz on Tue, 12 Mar 2002 01:11:01 GMT
View Forum Message <> Reply to Message

"Rick Towler" <rtowler@u.washington.edu> wrote in message
news:<a6iuea$oc6$1@nntp6.u.washington.edu>...
>>> 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.)
>
> I whipped up a little hack to test this and I do see the benefits in high
> polygon scenes. I suspect that what is being cached are the vertex normals
> used to compute lighting. Low poly scenes aren't limited by these extra
> calculations so you don't see the benefit. My test used an orb with a
> density of 10. Without changing the vertex data I would get around 65
> frames/second. When I did change the vertex data on each frame I got around
> 15 fps (you need to comment out the "self->BuildPoly" line in
> orb::setproperty for this to work).

Right. I forgot about the normals when I said that caching didn't
help polygons as much. Nice results.

>>> 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.
>>
>> 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.
>>
>
> Not totally true. Visibility of OG atoms is controlled by position in 3D
> space AND by drawing order. The latter only comes into play when your atoms
> are transparent (textured with a image containing an alpha channel). Karl
> might have to correct me but as the view heirarchy is traversed the atoms
> are drawn negative z verts to positive. This has implications both in the
> visibility of the scene as a whole (seeing thru one atom to another) and for
> the visibility of the atoms themselves (seeing the back side of a
> transparent 3d object). For a full discussion on this search the newsgroup
> for "pimento problems".

Oh no, not the pimento!!!! :-)

The graphic atom drawing order depends completely on the order in which models are placed in the view, and the order in which atoms are placed into models.

Within an atom, the order in which vertices are drawn happens to depend on the order that they are described in the primitive, and not by their Z ordering. For example, if you have an IDLgrPolygon object with a connectivity list that starts: [4, 200,201,202,203...], the first polygon drawn will use verts 200, 201, 202, and 203, regardless of the Z value of these vertices as compared to the other verts in the object.

As Rick mentions, this is really only gets important when working with transparency.

Part of the pimento discussion was about the Orb object. The Orb object generates a mesh which happens to draw the triangles from negative Z to positive Z. That is a property of the Orb object and nothing else.

I have a little test program that draws three parallel polygons with transparency. The program uses a trackball and changes the order of the polygon drawing as the model "flips" from one side to another so that the transparency always looks right. The problem is pretty easy to solve for simple data like this, but gets really messy for more complex scenes. I'll clean up the program and send it to anyone who is interested.

Karl
