Subject: Re: graphics formats, triangulisation & coordinate systems
Posted by Rick Towler on Thu, 28 Aug 2003 23:34:41 GMT
View Forum Message <> Reply to Message

"Neil" wrote in message ...
> "Rick Towler" wrote in message >...
>> "Neil" wrote in message ...

>>> I also need to convert all polygons to triangles, is there a simple
>>> triangulisation code i can use. I work in IDL so could this code be
>>> used in the conversion. To take care of reflections i need to ensure
>>> all surface are plane by definition, so i would like to convert all
>>> otherwise polygons to triangles
>>
>> What polygons? How are they defined? If they are the polygons
contained in
>> your .3ds files then they are already defined as a collection of
triangles
>> by the vertices and connectivity data.
>>

> many thanks for you helpful comments. As for polygons other than
> triangles, i was wanting a code that could read files that had say
> rectangles, pentagons, hexagons or what ever might exist out there.
> Not being such a specialist in this area, i am not really aware of how
> common other type of polygons are in this graphics business. However,
> for me to deal with reflections i would prefer something that had a
> plane surface by definition.

> Are polygons in graphics always triangles?

At some point, yes (more or less). It all depends on your 3d graphics API.
I say more or less because the "popular" 3d APIs OpenGL and DirectX eat
triangles for breakfast, lunch and dinner. (they also eat "quads" which are
triangle pairs.) You could build your own 3d software based rendering
system where you passed objects comprised of any type 2d polygon to your
renderer but if you are planning on doing that in IDL you must be a
masochist.

IDL's object graphics is an object oriented interface to openGL, IDL style.
Crack open the documentation and have a look. Ronn Kling has a book "Power
Graphics with IDL" which is a beginners guide to object graphics and is a
good book to have on hand. You can pick it up at his website:
www.kilvarock.com

> Isnt there a routine in IDL called "tessellation" that i can use to
> triangulise polygons of an object?

Yes, and this is what you would need to do if you had, say, a mesh comprised of octagons. But I think that a mesh like this would be pretty rare. Since most graphics applications are built on openGL or DirectX everthing is eventually decomposed into tris. Often, the earlier you do that, the better.

> I also see a coordinate system being RH, but with x being towards the
> right, y being away from the viewer and z being upwards, and in fact
> started to right my code in that convention. Such a coordinate system
> seems easier to visulise. However, now i'm thinking possibly i should
> change to RH with z azis coming towards the viewer. Shouldnt be too
> much trouble at this early stage.

Regardless of any coordinate systems you devise, to display anything using object graphics you'll need to work with a RH coordinate system with +Z pointing towards the viewer. While many other coord systems exist, this is what has been adopted by OpenGL and IDL.

> I really want a code which is compatable with other formats as
> possible.

Hummmmm. Don't know what you mean...

> I was also planning on putting the orgin at the viewer's
> position. If i do that with RH with z coming toward the viewer, all z
> values for objects in the scene are negative. Still i suppose that
> does not matter provided you have all your coordinate transformation
> correct.

Before you wrestle with this too much, have a look at my camera object. It is an "improved" viewing system for object graphics. Set it up and point it at your subjects. You don't have to worry so much about the technical details of setting up the view.

http://www.acoustics.washington.edu/~towler/

I have a major update that I'll be posting in the next few days (I am working on the docs now.) It abstracts the view even more and I *hope* it will be even easier to use but you can get started with the old version.

> I am convinced RH coordinate is probably best, but which direction
> should be best for the z-axis?

Again, if you are going to be visualizing your results in IDL you'll need to work with a RH coordinate system with +Z pointing towards the viewer.

All this talk of a coordinate systems and reflections.... What are you trying to do? Pardon my assumption but it seems that you are just getting starting with 3d graphics and IDL's object graphics. If so, you really don't need to worry about this too much. IDL/openGL handles creating normals, lighting and the other details. You just supply the vertices and connectivity (as tris or quads). Unless you are creating procedural texture maps or meshes I can't think of a reason to lose sleep over these things beyond knowing the coordinate system so you can place your objects in the world correctly.

Good luck!

-Rick