Subject: Re: Bug in IDLgrPolygon ?
Posted by Karl Schultz on Fri, 13 Dec 2002 14:30:49 GMT
View Forum Message <> Reply to Message

"Thomas Gutzler" <tgutzler@ee.uwa.edu.au> wrote in message
news:3DF947BE.F454E7E@ee.uwa.edu.au...
> Karl Schultz wrote:
>>
>> What you want to do is replace:
>>
>>   [4, 4, 5, 7, 6]]   ; vertices 5687
>>
>> with:
>>
>>   [0, 0, 0, 0, 0]]   ; vertices 5687
>>
>> If you try:
>>
>>   [0, 4, 5, 7, 6]]   ; vertices 5687
>>
>> then IDL skips over the 0 and thinks that the 4 is the vertex count for
the
>> next polygon.
>
> Yeah, I should have noticed this. doh!
>
>> -1 is the other special value which
>> means "stop here, even if there is more data after me" and really only
makes
>> sense to put in a list after a polygon definition.
>
> Hm, this was my last idea. Just move the 5 elements describing the
> polygon I want not be shown to the end and put a -1 in front. But I
> thought thats bad programming style, isn't it ?

I'm not sure about style, but appending the "disabled" polygon and the -1 to
the end of your polygon list is going to cause an alloc/free and a copy of
the entire list. This might be bad if the list is really long. If you
needed to put the disabled polygon back, you would also have to store
someplace the position of the disabled polygon in the main list. If you
accomplished this by also adding this index to your list after the -1, then
you're starting to store odd things in this list, which might make it
confusing to someone reading the code.

Another way to reenable the polygon would be to just change the -1 to a 0.
So you could disable a polygon by copying -1 and that polygon to the end,
and set the original polygon indices to 0. Change the -1 to 0 to turn the
polygon back on. After you do this a few times, you'll have a bunch of 0's

in the list that you can clean up with a periodic call to MESH_VALIDATE.
This approach would work best for short lists and if you disabled only one
polygon at a time and/or were OK with reenabling all of them at once.

If it were me, I'd think about just copying the disabled polygon out to
another data structure, along with its starting index. Then, I zap the poly
in the original list to zero. I copy the polygon back to the same place to
reenable it.

Hopefully one of these approaches will work well for your application.

I don't think that the polygon list was designed to get very sophisticated
with editing polygon lists like we're discussing. I think that the '0'
feature is there to allow the easy disabling of polygons that contain
vertices with bad data or NaN's. The -1 is used by some of the MESH_*
functions to indicate the the result is empty - there are no polygons in the
function's result.

One cool thing I thought of is that a negative value for a vertex count
would mean skip over the (positive) number of vertices. That is, a -3 in a
list would mean skip the next three elements. This would let you skip a
polygon by just changing the sign in the count. -1 would still mean "end of
list", since a polygon with 1 vertex isn't valid anyway. I don't know what
the utility/confusion ratio would be here...

HTH,
Karl