Subject: Re: Displaying 3-D vector fields
Posted by Rick Towler on Mon, 18 Nov 2002 19:54:39 GMT
View Forum Message <> Reply to Message

"Jim" <jim.blackwell@gsfc.nasa.gov> wrote
> "Rick Towler" <rtowler@u.washington.edu> wrote
>> "Jim" <jim.blackwell@gsfc.nasa.gov> wrote
>>
>>> what scales the size of the arrow (the length of the arrow relative to
>>> the shaft)?
>>
>> The vector is scaled according to the sqrt() of the sum of the squares
of
>> the components of the magnitude. The *entire* vector is scaled, head
and
>> all since I am using the model's transformation matrix to do the work
for
>> me. The vector is not drawn if the magnitude is 0.
>>
>>
>>> Is the default vector that is drawn of unit length ?
>>
>> Yes, the default vector is 1 unit long. It's default magnitude is
[0,0,-1]
>> which makes it parallel to the z axis pointing away from the viewer.
It's
>> default location is [0,0,0].
>>
>> -Rick
>
> I see. Say have you looked at the IDL program "vector_field" ? Seems
> to do something similar in that it creates vectors, however I don't
> know offhand how to implement this in IDL object graphics to see what
> it looks like ?

No. I haven't looked at it. I was satisfied with my vector as is. Well,
sort of.

After reading Mark's comments I was curious as to the impact of having large numbers of IDLgrModel objects in a scene. In our case it comes down to the cost of calculating the numerous transforms and internal procedural overhead.

Ideally, as Mark has implemented in his barb plot, you'll have one model and one atom. Ignoring any parent models, IDL calculates a transform based on the one model and one atom transform (3x3 coord_conv transform) and applies it to all the vertices. Very efficient, but limiting.

On the other end of the spectrum is a scene with 20k of my vector objects. Assuming we put our 20k models into a parent model, we need to do 40k matrix multiplies then apply each of those 20k transforms to the 4 verts that make up the vector. Not a model of efficiency but infinitely flexible.

Somewhere in the middle is a single model which contains 20k IDLgrPolyline objects. I modified my original vector object and created a 3d vector field object following this design. I end up calculating the transform and applying it to the vertex data manually when the location or magnitude data is changed. What little I did play with it showed around a 30-40% increase in redraw speed with 20k vectors. Although unfinished, you are welcome to this code as well. The meat is there, you'll just need to add the dressing.

In your case, none of these improvements will get you to the point where you will be able to interact in real time. It takes a long time to draw nearly 80K vertices. I think that you could go with any of them.

But back to your question. You still seem to be looking for an answer. What do you need that Mark's barb plot or my vector can't provide?

-Rick
