Subject: Re: Colliding galaxies in 3-D object?
Posted by Dick Jackson on Wed, 18 Feb 2004 06:59:01 GMT
View Forum Message <> Reply to Message

"Richard G. French" <rfrench@wellesley.edu> wrote in message
news:BC581B0F.1C17%rfrench@wellesley.edu...
> Hi, Folks -
> For an astrophysics class I'm teaching, I'm having the students write
a
> simple simulation of colliding galaxies (in IDL). They've got the
basic code
> working in standard graphics mode, with color coding for the stars in
the
> two galaxies and front and side views of the collision as it unfolds
over
> hundreds of time steps. What I'd like to do next is to view the
collision in
> 3-D with mouse control over the viewing angle and (ideally) the zoom
as
> well. The input information is a set of arrays of 3-D coordinates for
N red
> stars and N green stars, for M time steps. Can someone point me to the
3-D
> object routine in IDL that should be able to handle this, and (better
yet)
> alert me to any gotchas I should worry about as I try to implement
this?
>
> Thanks for any suggestions!
> Dick French
> rfrench@wellesley.edu

Hi Dick,

Rick Towler and I gave similar replies to a similar question last week
(thread: "An Interactively rotating 3D animation ?"... my only
half-usable post can be found by Googling for the word AnimateXObjView).
This might be a starting point for you, and the objects to put in the
XObjView could be:

```
n = 100  ; Number of stars in each galaxy
t = 2    ; If you want more than single-pixel stars
xyzRed = RandomN(seed, 3, n) - 1    ; (3, n) coords
xyzGreen = RandomN(seed, 3, n) + 1  ; (3, n) coords
oRed = Obj_New('IDLgrPolygon', xyzRed, Color=[255,0,0], $
             Style=0, Thick=t)
oGreen = Obj_New('IDLgrPolygon', xyzGreen, Color=[0,255,0], $
              Style=0, Thick=t)
```

(to simply see what these looks like, just paste the above into the
command line and do:
 XObjView, [oRed, oGreen]
)

Then, at each time point (my section labeled "Modify objects viewed in
XObjView window"), update the objects by:

oRed -> SetProperty, Data=newXYZRed
oGreen -> SetProperty, Data=newXYZGreen

Hope this helps!

Cheers,
--
-Dick

Dick Jackson / dick@d-jackson.com
D-Jackson Software Consulting / http://www.d-jackson.com
Calgary, Alberta, Canada / +1-403-242-7398 / Fax: 241-7392

Subject: Re: Colliding galaxies in 3-D object?
Posted by Rick Towler on Thu, 19 Feb 2004 01:56:51 GMT
View Forum Message <> Reply to Message

"Richard G. French" wrote...

> For an astrophysics class I'm teaching, I'm having the students write a
> simple simulation of colliding galaxies (in IDL). They've got the basic
code
> working in standard graphics mode, with color coding for the stars in the
> two galaxies and front and side views of the collision as it unfolds over
> hundreds of time steps. What I'd like to do next is to view the collision
in
> 3-D with mouse control over the viewing angle and (ideally) the zoom as
> well. The input information is a set of arrays of 3-D coordinates for N
red
> stars and N green stars, for M time steps. Can someone point me to the 3-D
> object routine in IDL that should be able to handle this, and (better yet)
> alert me to any gotchas I should worry about as I try to implement this?

Dick Jackson's suggestion is the place to start. Using XOBJVIEW for
animations is easy, but it doesn't provide a very good interface.

You may want to look at my camdemo_examine program. It provides the

interface you are looking for, you would just have to provide the objects to view and add a timer event for the animation.

There are two versions available, one packaged with RHTgrCamera and one packaged with the older camera__define package. Since this is for students, you may want to use the older camera__define code as it doesn't require any DLMs. The programs can be found here (older version at the bottom of the page):

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

Another suggestion would be to use my modified orb object to represent your stars. The standard orb object will rebuild its vertices for every call to Orb::SetProperty which you definitely do not want to do. Use the DENSITY to control polygon count as nStars gets large to keep the application running smoothly. To animate the orbs, use their Translate method.

The modified orb object can be found here:

http://www.acoustics.washington.edu/~towler/programs/orb__de fine.pro

-Rick
