Subject: Re: plotting on a sphere
Posted by Mark Hadfield on Sun, 07 Jul 2002 22:19:15 GMT
View Forum Message <> Reply to Message

"Chris O'Dell" <odell@cmb.physics.wisc.edu> wrote in message
news:3D260AC8.3050406@cmb.physics.wisc.edu...

> I am new to 3D graphing in IDL. I would like to plot various
> scalar fields on the surface of a sphere, displayed in 3D using
> color contours. Ideally, I would be able to then use my mouse to
> rotate the sphere to different orientations.

If you want 3D with rotations then you want object graphics.

To create a sphere in object graphics, you use MESH_OBJ to create a list of vertex positions and a connectivity list (i.e. a list specifying which vertices have to be connected to draw the shape). Then you feed these to an IDLgrPolygon object. Here's an example that creates & displays a plain-coloured sphere:

```
pro sphere_example
  compile_opt IDL2
  if n_elements(n_lon) eq 0 then n_lon = 20
  if n_elements(n_lat) eq 0 then n_lat = 20
  mesh_obj, 4, vert, conn, replicate(1, n_lon, n_lat)
  help, vert, conn
  sphere = obj_new('IDLgrPolygon', DATA=vert, POLY=conn, COLOR=[0,0,255],
STYLE=2)
  xobjview, sphere
end
```

To give the sphere a non-uniform colour you use the IDLgrPolygon's VERT_COLORS property. You will see that the above example creates a mesh with 400 vertices. The X, Y & Z positions of the vertices are held in the columns of a [3,400] floating-point array. The VERT_COLORS array should be a [3,400] byte array, with the columns corresponding to red, green and blue respectively. Into this you need to load the color at each vertex, expressed as a function of the X, Y and Z position at that vertex.

--
Mark Hadfield "Ka puwaha te tai nei, Hoea tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)

Subject: Re: plotting on a sphere
Posted by Christopher W. O'Dell on Mon, 08 Jul 2002 16:18:55 GMT

Thanks to Mark and Dick Jackson!
You both effectively gave me the same answer, and i used it and it
works. Of course now i am required to learn object graphics somewhat --
like how do I get a title and a color bar on the "XOBJVIEW" screen?
But the basics work great!

Cheers,
Chris

Mark Hadfield wrote:

> "Chris O'Dell" <odell@cmb.physics.wisc.edu> wrote in message
> news:3D260AC8.3050406@cmb.physics.wisc.edu...
>
>
>> I am new to 3D graphing in IDL. I would like to plot various
>> scalar fields on the surface of a sphere, displayed in 3D using
>> color contours. Ideally, I would be able to then use my mouse to
>> rotate the sphere to different orientations.
>>
>
> If you want 3D with rotations then you want object graphics.
>
> To create a sphere in object graphics, you use MESH_OBJ to create a list of
> vertex positions and a connectivity list (i.e. a list specifying which
> vertices have to be connected to draw the shape). Then you feed these to an
> IDLgrPolygon object. Here's an example that creates & displays a
> plain-coloured sphere:
>
> pro sphere_example
>   compile_opt IDL2
>   if n_elements(n_lon) eq 0 then n_lon = 20
>   if n_elements(n_lat) eq 0 then n_lat = 20
>   mesh_obj, 4, vert, conn, replicate(1, n_lon, n_lat)
>   help, vert, conn
>   sphere = obj_new('IDLgrPolygon', DATA=vert, POLY=conn, COLOR=[0,0,255],
> STYLE=2)
>   xobjview, sphere
> end
>
> To give the sphere a non-uniform colour you use the IDLgrPolygon's
> VERT_COLORS property. You will see that the above example creates a mesh
> with 400 vertices. The X, Y & Z positions of the vertices are held in the
> columns of a [3,400] floating-point array. The VERT_COLORS array should be a
> [3,400] byte array, with the columns corresponding to red, green and blue
> respectively. Into this you need to load the color at each vertex, expressed
> as a function of the X, Y and Z position at that vertex.

>
> --
> Mark Hadfield "Ka puwaha te tai nei, Hoea tatou"
> m.hadfield@niwa.co.nz
> National Institute for Water and Atmospheric Research (NIWA)
>
>
>

Subject: Re: plotting on a sphere
Posted by Mark Hadfield on Mon, 08 Jul 2002 21:32:49 GMT
View Forum Message <> Reply to Message

"Chris O'Dell" <odell@cmb.physics.wisc.edu> wrote in message
news:3D29BB6F.10405@cmb.physics.wisc.edu...

> Thanks to Mark and Dick Jackson! You both effectively gave me the
> same answer, and i used it and it works. Of course now i am
> required to learn object graphics somewhat -- like how do I get a
> title and a color bar on the "XOBJVIEW" screen?

To display more than one graphics atom in XOBJVIEW, you can collect the references to those atoms in an array & pass that to XOBJVIEW.

For the title you can use an IDLgrText object. For the colour bar you can use an IDLgrColorbar (though I've always found the IDLgrColorbar's facilities for positioning rather clumsy).

Another tip: XOBJVIEW does not destroy the objects passed to it, so you have to do that yourself (*after* the XOBJVIEW application has been closed). In an interactive session you may have lost the object references so you can use HEAP_GC.

But XOBJVIEW is rather limited and not extensible. If you want to go further with object graphics I suggest you do one or both of two things:

- Check out David Fanning's WWW site at http://www.dfanning.com/ and look at his object graphics programs. FSC_SURFACE is a good place to start.

- If you have IDL 5.5, download my Motley library from ftp://ftp.niwa.cri.nz/incoming/m.hadfield/, copy all the files to a directory on your IDL path, run the setup routine MGH_MOTLEY, then run and look at some of the example routines like MGH_EXAMPLE_SURFACE.

--
Mark Hadfield "Ka puwaha te tai nei, Hoea tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)

Subject: Re: plotting on a sphere
Posted by Paul Sorenson on Fri, 12 Jul 2002 02:09:23 GMT
View Forum Message <> Reply to Message

"Mark Hadfield" <m.hadfield@niwa.co.nz> wrote in message
>
> But XOBJVIEW is rather limited and not extensible.

Actually, XOBJVIEW is a widget object, or more accurately it is a top level base with a widget object inside of it. Because it is object oriented, all of the usual OOP techniques to extend code can be applied. You could probably customize the object's behavior by deriving a new class to suit your needs. Or you could include that object in a program of your own.

The underlying design of XOBJVIEW is a variation of the Bridge design pattern from the Gamma, et al book "Design Patterns." As applied to XOBJVIEW, the pattern separates the GUI widgets that you see on the screen from the object graphics classes that implement their behavior, thus separating interface from implementation.

This is useful if you want to create your own widget "skin" for XOBJVIEW's features. For example, you could probably write your own widget interface in Visual Basic and use the object graphics classes that underly XOBJVIEW for the implementation/behavior of those widgets via ActiveX. (You may need to write some simple wrapper classes to get this done, but I suspect it would not be hard to do.)

-Paul Sorenson