
Subject: Re: 3D point cloud visualization: filled polygons in the front, different fill colour + lines in the back

Posted by [Dick Jackson](#) on Tue, 26 May 2015 01:03:26 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hi Nuno,

You're nearly there, with option 3... on the oPolygon2, you had VERT_COLORS set, which gave every vertex its own colour, but if you just remove that, the lines will be black. Or, if you want a certain colour to be used (with this STYLE=1 wireframe), COLOR=[r,g,b] works, as does VERT_COLORS with a single [r,g,b] triple. I would go with COLOR as the simpler option.

Nice job!

Cheers,
-Dick

Dick Jackson Software Consulting Inc.
Victoria, BC, Canada --- <http://www.d-jackson.com>

On Monday, 25 May 2015 11:10:48 UTC-7, Nuno Ferreira wrote:

> And there was another error, sorry (wrong version...)! Here is the right version:

>

>

>

> PRO question_example_event, ev

> ; (not really needed for this example)

> END

>

>

> PRO question_example

> ; create a test object

> vertices = [[0, 0, 0], [1, 0, 0], [1, 1, 0], [0, 1, 0], [0.5, 0.5, 1]]

> connectivity = [3, 0, 1, 4, \$

> 3, 1, 2, 4, \$

> 3, 2, 3, 4, \$

> 3, 3, 0, 4]

> vert_colors = [[0,0,0],[255,0,0],[0,255,0],[0,0,255],[255,255,255]]

>

> ; Initialize model for display (2 views).

> oModel1 = OBJ_NEW('IDLgrModel')

> oModel2 = OBJ_NEW('IDLgrModel')

>

> ; Initialize polygon and/or polyline

> option=3

> CASE option of

>

> 1: begin

```

> oPolygon = OBJ_NEW('IDLgrPolygon', vertices, $
>   POLYGONS = connectivity, SHADING=1, $
>   vert_colors=vert_colors)
> oPolyline = OBJ_NEW('IDLgrPolyline', vertices, $
>   POLYLINES = connectivity, COLOR = [0, 0, 0])
> end
>
> 2: begin
>   oPolygon = OBJ_NEW('IDLgrPolygon', vertices, $
>     POLYGONS = connectivity, SHADING=1, $
>     vert_colors=vert_colors, bottom=[200,200,200], $
>     depth_offset=1)
>   oPolyline = OBJ_NEW('IDLgrPolyline', vertices, $
>     POLYLINES = connectivity, COLOR = [0, 0, 0])
>   end
>
> 3: begin
>   oPolygon = OBJ_NEW('IDLgrPolygon', vertices, $
>     POLYGONS = connectivity, SHADING=1, STYLE=2, $
>     vert_colors=vert_colors, bottom=[200,200,200], $
>     depth_offset=1)
>   oPolygon2 = OBJ_NEW('IDLgrPolygon', vertices, $
>     POLYGONS = connectivity, SHADING=1, STYLE=1, $
>     vert_colors=vert_colors, $
>     depth_offset=0)
>   end
> endcase
>
> ; Add the polygon(s) and/or polyline to the model.
> oModel1 -> Add, oPolygon
> if option EQ 1 or option EQ 2 then oModel1 -> Add, oPolyline
> if option EQ 3 then oModel1 -> Add, oPolygon2
> oModel2.Add, oModel1, /alias ; create an alias for 2nd model
>
> ; used for display:
> ov1 = idlgrview(viewplane_rect=[-2, -2, 4, 4], zclip=[2, -2], eye=2.1)
> ov1.add, oModel1
> ov2 = idlgrview(viewplane_rect=[-2, -2, 4, 4], zclip=[2, -2], eye=2.1)
> ov2.add, oModel2
>
> ; create GUI
> s = 256
> base = widget_base(/row, Title='Option '+string(option))
> d1 = widget_draw(base, graphics_level=2, xsize=s, ysize=s, tooltip="Front")
> d2 = widget_draw(base, graphics_level=2, xsize=s, ysize=s, tooltip="Back")
>
> ; show GUI and model
> widget_control, base, /realize

```

```
> widget_control, d1, get_value=ow1
> widget_control, d2, get_value=ow2
> oModel1 -> Rotate, [1, 0, 0], 45 ; Rotate to better show the front side
> ow1.setproperty, graphics_tree=ov1
> ow1.draw
> oModel1 -> Rotate, [1, 0, 0], 180 ; Rotate again to show the back side
> ow2.setproperty, graphics_tree=ov2
> ow2.draw
>
> xmanager, 'question_example', base
> END
```
