
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 17:27:50 GMT

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

On Tuesday, 26 May 2015 01:14:55 UTC-7, Nuno Ferreira wrote:

> Thanks Dick, but in that case the black lines are drawn also in the front face, hiding the colors of the polygons (and with thousands of very small polygons, one would see mainly black in the front... you can see a sample of my data here, shown with option 3:

<https://drive.google.com/file/d/0B6Ti5FMqve-da0RXNIIHUFJfSlk/view?usp=sharing>).

Ah, of course. :-)

> What I would like to have, looking at the other image (

<https://drive.google.com/file/d/0B6Ti5FMqve-dRzRuMk9yY1FjM2c/view>) is this: option 3 in the front and option 2 in the back.

>

> In more general terms, I guess I would like to have the possibility of viewing points, lines and filled polygons with certain settings in the front face, and different settings in the back face.

Then I think we have to get tricky... To avoid the wireframe lines showing through to the *front*, we can't use DEPTH_OFFSET in a way that works from every point of view (DEPTH_OFFSET lets you push one object's apparent draw-ordering "away" from you at all times). Here's a way to make a shifted set of vertices for the wireframe that is always "inside" the mesh of the shaded surface:

```
oPolygon = OBJ_NEW('IDLgrPolygon', vertices, $ ; Shaded surface
  POLYGONS = connectivity, SHADING=1, STYLE=2, $
  vert_colors=vert_colors, bottom=[200,200,200], $
  depth_offset=0)
normalsXYZ = Compute_Mesh_Normals(vertices, connectivity)
vertices2 = vertices - normalsXYZ * 0.02 ; Magic number!
oPolygon2 = OBJ_NEW('IDLgrPolygon', vertices2, $ ; Wireframe
  POLYGONS = connectivity, STYLE=1, $
  depth_offset=1)
```

That magic number happens to work well here. Try 0.01 (some bits of lines poke through the front) and 0.1 (noticeable gap between wireframe and shaded surface) to reveal the mystery. Your models may require a different value for good performance.

Also, see my "****" notes for a couple of other tips to allow you to keep windows on the screen for easy comparison.

> Many thanks!

>

> Nuno

You're welcome!

-Dick

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

;-----

PRO nuno_question_example_1_event, ev
; (not really needed for this example)
END

PRO nuno_question_example_1
; 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, $ ; Shaded surface
  POLYGONS = connectivity, SHADING=1, STYLE=2, $
  vert_colors=vert_colors, bottom=[200,200,200], $
  depth_offset=0)
normalsXYZ = Compute_Mesh_Normals(vertices, connectivity)
vertices2 = vertices - normalsXYZ * 0.02 ; Magic number!
oPolygon2 = OBJ_NEW('IDLgrPolygon', vertices2, $ ; Wireframe
  POLYGONS = connectivity, STYLE=1, $
  depth_offset=1)
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", retain=2) ;***
d2 = widget_draw(base, graphics_level=2, xsize=s, ysize=s, $
  tooltip="Back", retain=2) ;***

```

```

; 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, 'nuno_question_example_1', base, /no_block ;***

```

END