
Subject: Re: help with transformations and scaling
Posted by [Richard Tyc](#) on Fri, 25 Jun 1999 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

Richard Tyc wrote:
I forgot to add the example code. Here it is.

```
.,*****  
,
```

Function ComputeBounds, HeadTarr

```
s = size(HeadTarr)  
ind = where (HeadTarr gt 0)  
  
indz = ind/(s[1]*s[2])  
indy = (ind-(indz*s[1]*s[2])) / s[1]  
indx = (ind-(indz*s[1]*s[2])) - (indy * s[1])  
  
maxTz = max(indz)  
minTz = min(indz)  
  
boundt=[min(indx),min(indy),minTz,max(indx),max(indy),maxTz]  
  
return, boundt
```

END

```
.,*****  
,
```

Function CubeData,xdim,ydim,zdim

```
testC = dist(xdim,1)  
testc = (testc/max(testc)) * 150.0  
cube = bytarr(xdim,ydim,zdim)  
  
for i=0,zdim-1 do begin  
  for j=0,ydim-1 do cube[* ,j,i] = byte(testc)  
endfor  
  
cube = cube+100  
  
return,cube
```

END

```

; Here's the test sample code, I run it as main
; It is definitely NOT optimized yet for memory leaks etc. !!
; Changed transformations to global !

;Pro testcutplaneNETPOST, oWindow

; This version draws cannula and its coordinate system.
; The cannula here is variable based on xc,yc,zc

;volume data dimension

DataDim = [20,20,10]

xfac = 0.5
yfac = 0.5
zfac = 0.4

; define cannula tip position
xc = xfac*DataDim[0]
yc = yfac*DataDim[1]
zc = zfac*DataDim[2]

cube = CubeData(DataDim[0],DataDim[1],DataDim[2])

boundT = computebounds(cube)

s=size(cube)
xs=s[1]-1
ys=s[2]-1
zs=s[3]-1

; I changed it to this although it doesn't seem to matter
LenVox = sqrt((xc*xs/DataDim[0])^2 + (yc*ys/DataDim[1])^2 + $
              (zc*zs/DataDim[2])^2)

cutplnVox = [(xc*xs/DataDim[0])/LenVox, (yc*ys/DataDim[1])/LenVox, $
              (zc*zs/DataDim[2])/LenVox, -LenVox ]

cpNorm    = cutplnVox[0:2]

cpNormCtr = [xc/s[1],yc/s[2],zc/s[3]]

Scale3, XRange=[0,xs], YRange=[0,ys], ZRange=[0,zs]

rh = fltarr(4,4)

```

```

oView = OBJ_NEW('IDLgrView', color=[0,0,0],viewplane_rect=[0,0,1,1], $
    dimensions=[512,512])

oModel = OBJ_NEW('IDLgrModel')
oView->add,oModel

oModelC = OBJ_NEW('IDLgrModel')
oView->add,oModelC

if (not obj_valid(oWindow)) then $
    oWindow = OBJ_NEW('IDLgrWindow',dimensions=[512,512])

rgb = bytarr(256,3)

a=bindgen(256)
rgb[* ,0] = a
rgb[* ,1] = a
rgb[* ,2] = 0

opac = bindgen(256)

oVolCube = OBJ_NEW('IDLgrVolume', cube, bounds=boundt, $
    xcoord_conv=!x.s,ycoord_conv=!y.s, zcoord_conv=!z.s )
oVolCube->setproperty,opacity_table0=opac
oVolCube-> setproperty,rgb_table0=rgb,interpolate=1,composite_function= 0
oVolCube->setproperty,cutting_planes=[cutplnVox]
oVolCube->setproperty,/zbuffer

oXaxis = OBJ_NEW('IDLgrAxis',0,color=[255,0,0],range=[0,xs],thick=2, $
    major=0,minor=0,xcoord_conv=!x.s,ycoord_conv=!y.s, $
    zcoord_conv=!z.s,/notext)

oYaxis = OBJ_NEW('IDLgrAxis',1,color=[0,255,0],range=[0,ys],thick=2, $
    major=0,minor=0,xcoord_conv=!x.s,ycoord_conv=!y.s, $
    zcoord_conv=!z.s,/notext)

oZaxis = OBJ_NEW('IDLgrAxis',2,color=[0,0,255],range=[0,zs],thick=2, $
    major=0,minor=0,xcoord_conv=!x.s,ycoord_conv=!y.s, $
    zcoord_conv=!z.s,/notext)

oCannula = OBJ_NEW('IDLgrPolyLine', color=[0,255,0], $
    [[0,0,0],[cpNormCtr]], thick=5)

; Toggle this line on off to display/not display volume
oModel->Add,oVolCube

oModel->Add,oXaxis
oModel->Add,oYaxis

```

```
oModel->Add,oZaxis
oModel->Add,oCannula
```

```
CanAxLen = byte(0.55*DataDim[0])
```

```
oXCaxis = OBJ_NEW('IDLgrAxis',0,color=[0,0,255],range=[0,CanAxLen], $
    major=0,minor=0,xcoord_conv=!x.s,ycoord_conv=!y.s, $
    zcoord_conv=!z.s,/notext)
oYCaxis = OBJ_NEW('IDLgrAxis',1,color=[0,255,0],range=[0,CanAxLen], $
    major=0,minor=0,xcoord_conv=!x.s,ycoord_conv=!y.s, $
    zcoord_conv=!z.s,/notext)
```

```
oModelC->Add,oXCaxis
oModelC->Add,oYCaxis
```

```
; reset view to fit
t3d,/reset
t3d,trans=[-.5,-.5,-.5],scale=[.577,.577,.577]
t3d,rot=[-90,300,0]
t3d,rot=[220,0,0]
t3d,trans=[.5,.5,.5]
```

```
; Set cube data (global) view transformation
oModel->SetProperty,transform=!p.t
```

```
; Now define local coordinate system at tip of oCannula
; First setup transformation to bring cannula coord system
; out to tip of cannula drawn above
```

```
Rxy = sqrt(cpNorm[0]^2 + cpNorm[1]^2 )
```

```
if ((cpNorm[0] eq 0.0) and (cpNorm[1] eq 0.0)) then Az = 0.0 $
else Az = ATAN(cpNorm[1], cpNorm[0]) * !radeg
```

```
if ((Rxy eq 0.0) and (cpNorm[2] eq 0.0)) then Ay = 0.0 $
else Ay = ATAN(cpNorm[2], Rxy) * !radeg
```

```
t3d,/reset,rot=[0,90-Ay,AzG]
t3d,trans=[cpNormctr[0],cpNormctr[1],cpNormctr[2]]
```

```
ctrans = !p.t
```

```
; this transformation redisplay image in better format for
;viewing. It is used for both coordinate systems.
```

```
t3d, /reset, trans=[-0.5,-0.5,-0.5],scale=[.577,.577,.577]
!p.t = ctrans # !p.t
t3d,rot=[-90,300,0]
```

```
t3d,rot=[220,0,0]
t3d,trans=[.5,.5,.5]
```

```
oModelC->SetProperty,transform=!p.t
oWindow->Draw,oView,/draw_instance
```

```
;OBJ_DESTROY,oWindow
OBJ_DESTROY,oView
```

END

File Attachments

- 1) [testcutplaneNETPOST.pro](#), downloaded 88 times
 - 2) [richt.vcf](#), downloaded 84 times
-