JAPLA 研究会資料 20013/6/15

J-OpenGL による 3D グラフィックスーその 10*)
メビウスの帯へ向けて
ーJ-OpenGL をどう理解するかー

西川　利男

Jで OpenGL を利用することで高度の三次元グラフィックスの処理が可能となった。
前回の例会で志村正人氏よりメビウスの帯やクラインの壷など、アートとして眺めるだけでも楽しいグラフィックスがいろいろ紹介された。しかし、私としては OpenGL を3次元空間内でのものの位置や運動の幾何学を理解する道具として使いたいと思う。そして3次元の世界をコンピュータの2次元のディスプレイ画面にどう表示するか、という投影幾何学(Projective Geometry)を手軽に実現する場と位置づけたい。

Jで OpenGL を楽しむだけであれば、J の Lab の Demo でいろいろなものが見られる。プログラムのコードもついている。いくつかは Stand Alone で動く J のコードも添えられているが、それを理解するのは容易ではない。

1.J-OpenGL の環境

私の OpenGL への攻略法としては、以下のような成書を読みつつ、一方では J の Demo を参考に、自分でプログラムを作り、実験しながら理解する道をとる。

・Mason Wao, Jackie Neider, Tom Davis 著 (株)アクロス訳
「OpenGL プログラミングガイド(ver. 1.1)」アジソンウェスレイジャパン(1997).
・酒井幸市著「OpenGL でつくる3次元 CG & アニメーション」森北出版(2008).
・プロジェクションマッピング制作「OpenGL 入門」
http://wisdom.sakura.ne.jp/system/opengl/index.html

OpenGL は汎用性のグラフィックス・プログラムの仕様とはいうものの、元々は C++言語を前提にしていることから、その知識と OpenGL の特別の書式に慣れる必要がある。

J の OpGL ではこの仕様に準拠した、以下のライブラリが備えてある。

gl3 …… J の OpenGL へのプリミティブ
jzopengl…… J でプログラミングのための名詞、動詞
jzopenglutil… J6 のためのさらに広範囲のユーティリティ

つまり、J では gl3 と jzopengl より

glRotate, glTranslate, gluSphere, gluCylinder などの OpenGL 命令を同じように用いて処理をおこなう。このように OpenGL の命令は gl(大文字)、glu(大文字)のような名前である。実際の書式は J に合わせてあるが、違和感はない。

さらに gl(小文字)、gls(小文字)のような J 独自の多くの命令がある。J の Demo プログラムでは、これらが縦横に使われていて、さらに OpenGL の本来の構文とも異なり
解読は容易ではない。

Lab の Demo は多くの例を次々と、いわば劇場の大舞台で演ずるものである。しかし、OpenGL を理解する上からは、手作りの小屋掛けで行う方がよい。以下このような考え方で話しを進めていく。

*) 以前(2009 年)の報告(J-OpenGL とは、正 12 面体、正 20 面体、サッカーボール、フラードーム、回転するサイコロ)から、新たな再開とし、番号付けをこのようにした。

2.J-OpenGL のプログラム構成

プログラムは OOP として OpenGL 機能を取り込んだイベントドリブンのウィンドウズプログラムとして作成される。

```
require 'gl3'                OpenGL プリミティブの導入
load 'jzopengl'                    OpenGL プリミティブの導入
coinsert 'jzopengl'          プログラム便利な動詞、名詞が使えるようになる

A=: noun define              ウィンドウズのフォーム作成のデータ
pc a closeok;
xywh 0 0 340 300;cc g isigraph ws_clipchildren ws_clipsiblings;
 ---
)

run=: a_run                  ウィンドウズ・プログラムの起動実行
a_run=: verb define
wd A
glaRC''
 (初期設定)
wd 'pshow;ptop'
)

a_g_size=:verb define              表示画面の大きさ、投影条件など
wh=.glqwh''
glViewport 0 0,wh
glMatrixMode GL_PROJECTION
glLoadIdentity''
gluPerspective 45, (%/wh),1 8
)

a_g_char =: verb define      キー入力、コマンドによる選択、実行のため
key =. 0 { sysdata
)

a_g_paint =: verb define     画面への表示条件など
glClearColor 1 1 1 0
glTranslate T
glRotate R ,. 3 3 $ 1 0 0 0
glScale 3#SC
light ''                     照明の条件
demo1 ''                     図形の作成、頂点の座標位置など
glaSwapBuffers ''
```

)
　a_run "を起動すると、a_g_size、a_g_char、a_g_paint も同時に実行される。
これにより、最小限の OpenGL のウィンドウズ環境が表示される。
　いろいろな図形の作成などほとんどの処理は a_g_paint にプログラムすることでなされる。

3.Lab Demo のメビウスの帯の実行


```
NB. Moebius Original Program ========================
TDATA=: moebius"
TCLR=: (0 0 0 ,: 1 1 1) hue fit01 2{"1 TDATA
TPATTERN=: 16 16$, RGBA WHITE,GRAY,YELLOW,:WHITE
TPATPOS=: */˜ int01 <:#TDATA
ROTXYZ=: 325 205 155
CLEARCOLOR=: 0.9 0.9 1
COLORMATERIAL=: 0
AMBIENT=: 0.8
DIFFUSE=: LIMEGREEN
SPECULAR=: DIFFUSE
SHININESS=: 50 0 0 1

demo1=: 3 : 0
stdlistlight"
gentexture2D TPATTERN
demo1x "        NB. revised / call demo1x
NB. 2 drawsurface makenewlist TDATA;TCLR;TPATPOS  Original Version
)
```

```
demo1x=: 3 : 0   NB. revised / explicit OpenGL syntax
glNewList DEMOX, GL_COMPLIE
 2 drawsurface TDATA;TCLR;TPATPOS
glEndList ''
)
```

Jの Lab Demo のプログラムリストは上のとおりである。demo1 のように修正することで、先のわれわれのウィンドウズ環境で Stand Alone で実行できる。しかし、上のコーディングはあまりにコンパクトであり、解読できるものではない！

多くの見慣れない動詞、名詞は'jzopengl'に定義されているが、本来の OpenGL のコマンドは隠されてしまっている。また構文もコンパクトに変えられている。

さらにここではベジエ・エバリュエータ、NURBS(Non-Uniform Rational B-Spline)による曲面の平滑化など OpenGL の最高のテクニックが駆使されている。

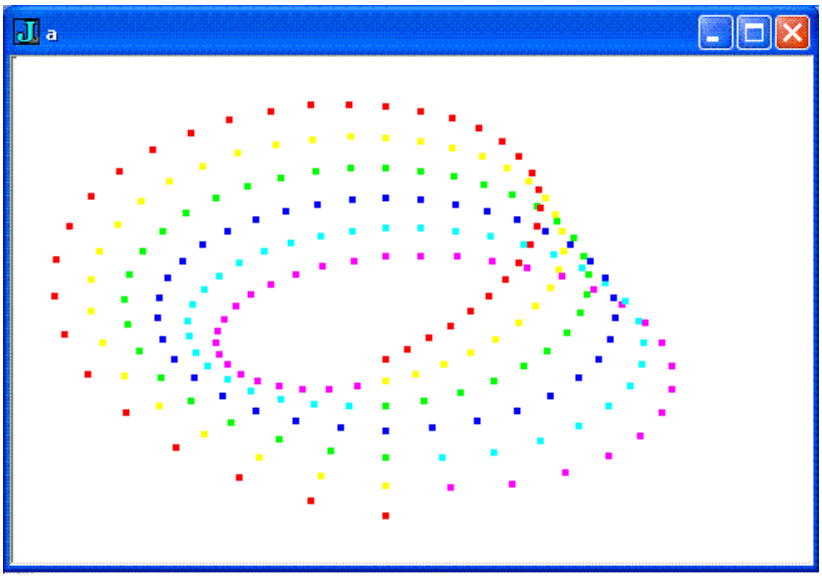
私としては、このレベルを目指すものの、もっと初歩のレベルの OpenGL から一歩ずつ攻略して行くしかない。

4.メビウスの帯の頂点座標の計算と初歩の OpenGL

メビウスの帯の座標値(x, y, z)は次の式で計算される。

$$x = (2 + t\cos(s/2))\sin s$$
$$y = (2 + t\cos(s/2))\cos s$$
$$z = t\sin(s/2)$$
$$where \quad 0 \le s \le 2\pi, \quad -1 \le t \le 1$$

まずはこれを元にパラメータ s と t とをそれぞれ分割して、色を付けた点の集合として、OpenGL の本来のコマンド用いたプログラムにより図形を表示してみよう。

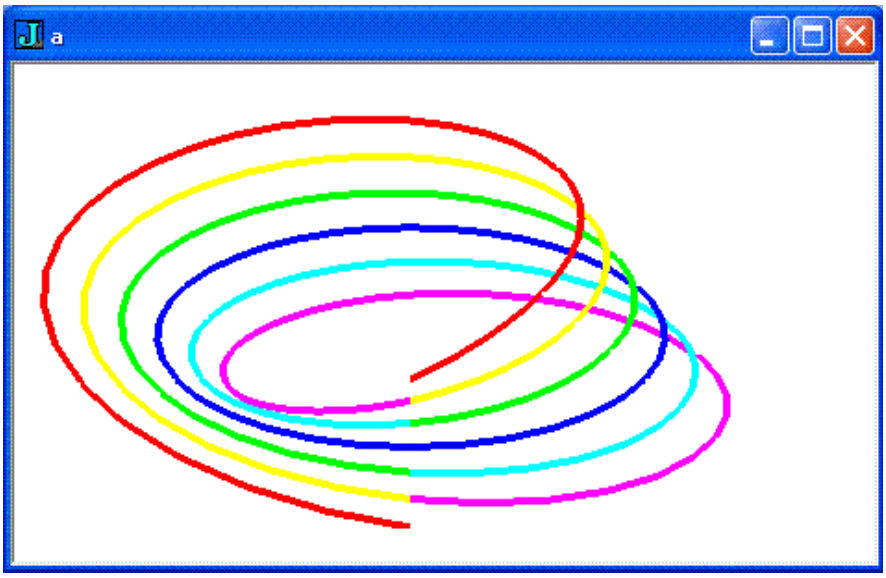


そのプログラムは以下のようになる。先の a_g_paint の中で demo1 ''に代わり test1 ''として、実行される。

```
NB. test moebius by TN / 2013/5/29 ========================
makedata =: 3 : 0                NB. メビウスの計算を行う
's t' =. y.
x =. (2 + t*cos(s%2)) * sin s
y =. (2 + t*cos(s%2)) * cos s
z =. t*sin(s%2)
cleanz |: > x;y;z
)

stepn =: 3 : 0                   NB. f から t までを n に分割した値を得る
'f t n' =. y.
f + (t - f) * (i. >: n) % n
)

SS =: stepn 0, 2p1, 36           NB. SS:0 から 2π までの s を 36 等分した値
TT =: stepn _1, 1, 6             NB. TT:_1 から 1 までの t を 6 等分した値
NDATA1 =: |: makedata L:0 { SS ; TT   NB. メビウスの頂点座標
NCOLOR =: 1 0 0 1;1 1 0 1;0 1 0 1;0 0 1 1;0 1 1 1;1 0 1 1  NB. 6色の値

ntest1 =: 3 : 0                  NB. 図形の設定(レンダリング)
glPointSize 4                    NB. 点の大きさ
i =. 0
while. i < 6
  do.
    glBegin GL_POINTS            NB. 単独の点として設定
      glColor  >(6|i){NCOLOR     NB. 色の指定
      glVertex >i{NDATA1         NB. 頂点の座標
    glEnd ''
  i =. i + 1
end.
)
```

次に、上の頂点の代わりに以下のように線に変えて、実験してみよう。

```
  glLineWidth 4
    glBegin GL_LINE_STRIP        NB. 4 角形の連続として設定
      glColor  >(6|i){NCOLOR     NB. 色の指定
      glVertex >i{NDATA1         NB. 頂点の座標
    glEnd ''
```



さらに、詳細な検討は次回以降に行いたいと思う。

```
NB. OpGLN_Moebius.ijs
NB.   modified executable standalone by T.N 2013/5/28

require 'gl3'

load 'jzopengl'

coinsert=: 3 : 0
n=. ;: :: ] y.
p=. ; (, 18!:2) @ < each n
p=. ~. (18!:2 coname''), p
(p /: p = <,'z') 18!:2 coname''
)

coinsert 'jzopengl'

A=: noun define
pc a closeok;
xywh 0 0 340 300;cc g isigraph ws_clipchildren ws_clipsiblings rightmove
bottommove;
pas 0 0;
rem form end;
)

run=: a_run
a_run=: verb define
wd A
glaRC''
R =: 120 0 0
NB. R =: 90 0 0
NB. T =: 0 0 0
T =: 0 0 _4
SC =: 0.5
glaFont 'arial 30'
glaUseFontBitmaps 0 32 26 32
wd 'pshow;ptop'
)

a_g_size=:verb define
wh=.glqwh''
glViewport 0 0,wh
```

```
glMatrixMode GL_PROJECTION
glLoadIdentity''
gluPerspective 45, (%/wh),1 8
)
a_g_char =: verb define
R =: 360 | R + 2 * 'xyz' = 0 { sysdata
R =: 360 | R - 2 * 'XYZ' = 0 { sysdata
SC =: SC * 1 + 0.125 * 'b' = 0 { sysdata  NB. bigger
SC =: SC * 1 - 0.125 * 's' = 0 { sysdata  NB. smaller
glpaintx''
)

NB. indicate rotated angle x, y, z in degree
drawtext =: verb define
glMatrixMode GL_MODELVIEW
glLoadIdentity ''
glColor 1 0 0 0
glRasterPos _16 16 _4
glCallLists 5 '': R
glpaintx''
)

a_g_paint =: verb define
glClearColor 1 1 1 0
glClear GL_COLOR_BUFFER_BIT + GL_DEPTH_BUFFER_BIT
glEnable GL_DEPTH_TEST
glMatrixMode GL_MODELVIEW
glLoadIdentity''
glTranslate T
glRotate R ,. 3 3 $ 1 0 0 0
glScale 3#SC
light ''
demo1 ''
glaSwapBuffers ''
)

demo1=: 3 : 0
stdlistlight''
gentexture2D TPATTERN
demo1x ''       NB. revised / call demo1x
NB. 2 drawsurface makenewlist TDATA;TCLR;TPATPOS  Original Version
```

```
)

demo1x=: 3 : 0   NB. revised / explicit OpenGL syntax
glNewList DEMOX, GL_COMPLIE
 2 drawsurface TDATA;TCLR;TPATPOS
glEndList ''
)
```