JAPLA 研究会資料 2014/9/20

J-OpenGL による花のグラフィックスーユリの花
OOP(オブジェクト指向)方式の J-OpenGL プログラム

西川　利男

０．はじめに

花のグラフィックスと題して、Ｊのプログラムでアサガオ、ヒマワリなど花の絵を描いたのは、もう５年ほど前になる。志村正人氏から戸川隼人先生の「花のＣＧ」なる BASIC の本を見せられて、Ｊでもどうかとけしかけられ始めた。そのときはその本にのっていた BASIC のコーディングをむりやりＪで同じように書きなおしただけであった。しかし、これをきっかけに、OpenGL の威力と考え方に目覚め、Ｊでもこれが可能であり、J-OpenGL のプログラミングにのめり込み、今にいたっている。

ようやくその技法を自分のものとして、使いこなせるようになったと思っている。このようなわけで、上のような題目にたどりついた。ユリの花はアサガオと違って、花びらの内側が、外に向けて裏返って、ウラとオモテの２つの面がある。幾何学の図形としてみれば、３次元空間内での２次元の曲面の表示である。

先の直接、Ｊの gl2 を使ったプログラムと異なり、今回のＪの gl3 の OpenGL プログラムでは、いろいろな方向から視点を変えて眺めたり、ずっと広範囲の高度のグラフィックスが可能である。花のグラフィックスはそのためのかっこうの課題である。

さらに今回は、OOP(オブジェクト指向)方式のプログラムとした。つまりウィンドウズ・グラフィクスの表示、ボタンによる入出力インターフェースなど共通部分は別のクラス・ファイルとし、個々の花の形データの作成などは自由に変えられよう別のプログラムとし、そこから起動実行するよう、２つに分けたシステムとして構築した。

１．Ｊの通常グラフィックスと J-OpenGL グラフィックスとの違い

Ｊで数学の関数値を表示したり、多量の統計データを図示したりするには、'plot' をはじめとするいろいろなツールがあり、これについては志村氏の多数の報告がある。

しかし、図形そのものを自由に描いたりするには、もっとプリミティブな図形命令を用いておこなわざるを得ない。これまでは、'gl2' なる命令セットで行ってきた。

J-OpenGL では 'gl3' の命令セットを用いるが、これを OpenGL なる書式仕様のもとで行い、従来のグラフィックスとは、プログラミングの考え方がかなりの点で違う。

従来のグラフィックスでは、図形を描くのには、図形の輪郭となる値を計算した上で、例えば line(X0, Y0, X1, Y1, X2, Y2, ...)のように直接、線を引く。

これに対して、OpenGL では最初に右手系３次元の空間を決めて、その中で頂点座標を、V0(X0, Y0, Z0)、V1(X1, Y1, Z1) 、V2(X2, Y2, Z2)、… のように指示する。その上で、あらためて

```
glBegin GL_POLYGON
  glVertex V0
  glVertex V1
  glVertex V2
glEnd ''
```

のようにして、図形を描く。

一般にOpenGLでは、決められた書式に従ってモジュールに分けてプログラムする。J-OpenGLでもこれに倣い、次の構成で作成する。

(1) グラフィックフォームの設定　A　　プログラムのフォームを名詞Aとして定義
(2) プログラムの実行　　a_run　　Aに対応する動詞aの実行
　　aのチャイルドgをグラフィックスとして起動
(3) 図形のペイント　　a_g_paint
(4) 図形の投影条件　　a_g_size
(5) コマンド文字の入力　a_g_char

細かい具体的なコーディングは稿末に示す。

また、これらをOOPとしてプログラムする手順はJの[File]メニューの[New Class]と [Edit]メニューの[Form Editor]とで、ウィザードを使って行われるが、この詳細についても付録として稿末に示す。

２．花のグラフィックス作成のプログラムの流れとポイント

２．１

花の形を３次元の幾何学図形とみるとき、これは円錐台の一種であって、その母線が直線ではなく、曲がった曲線になっているものと考えることができる。したがって、グラフィックスの第一歩としてこのような曲線を数式で表すことから始める。

戸川先生のBASICのコーディングにならい、母線の３次元の座標値U, V, Wを計算するプログラムをJでつぎのように作った。

```
require 'trig'

NB. Global Values ====================================================
NN =: 4
MM =: 16
PI =: 1p1
TH0 =: PI%36
TH1 =: PI*1.2
UZERO =: 210
VZERO =: 70
DU =: UZERO%MM
DW =: DU

lily0 =: 3 : 0    NB. lily flower generating line =====================
'u v w' =: 0
'U V W' =: 0
i =: 1
while. i <: MM
  do.
    S =. i%MM
    THETA =. TH0 + (TH1-TH0)*S^2
    u =. +/ u, DU*sin(THETA)
    v =. VZERO*S*(1-S)
```

```
    w =. +/ w, DW*cos(THETA)
NB.     wr i, u, v, w
    U =: U, u
    V =: V, v
    W =: W, w
    i =. i + 1
  end.
NB. U,. V,. W
)
```

２．２

つぎに、母線をいくつかに区切る。そのｉ番目の点U(i), V(i), W(i)を通って、Z軸の回りに半径√U(i)^2 + W(i)^２で、XY平面に平行な円を作る。この円を円周に沿って、例えば１２等分する。このようにして、花の面のメッシュ点の座標値を計算する。そのプログラムはつぎのようになる。

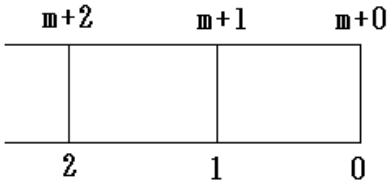
```
NB. Global Values ================================================
UVW =: U,. V,. W
UVW0 =: (U,. V) ,. 0
K =: MM + 1   NB.  stem interval section ==========================
DA0 =: (0.01, 0.06, 1) *"(1, 1) UVW0

lily4 =: 3 : 0
X0 =. 0{"(1) DA0
Y0 =. 1{"(1) DA0
R =. 12      NB. radial section
XYZ =: ''
i =. 0
while. i < K
  do.
    Xi =. (i{X0) * cos((i.R)*(1r6p1))
    Yi =. (i{Y0)
    Zi =. (i{X0) * sin((i.R)*(1r6p1))
    XYZi =. Xi ,. Yi ,. Zi
    XYZ =. XYZ, XYZi
    i =. i + 1
  end.
XYZ
)
```

２．３

花のメッシュ点が得られたとしても、それをつないだだけでは、花びらの形にはならない。メッシュ点の各部分ごとに長方形や三角形でつないで微小図形の集合とすることで－戸川先生のことばを借りれば、有限要素法式に－花の全体の形になる。

このような花の曲面を OpenGL で曲面をメッシュで表示するには、つぎのようにすべての頂点座標を、長方形の４頂点として指示することが必要である。かつ、その順序は反時計方向でなければならない。

(0,m+0,m+1,1),(1,m+1,m+2,2),...

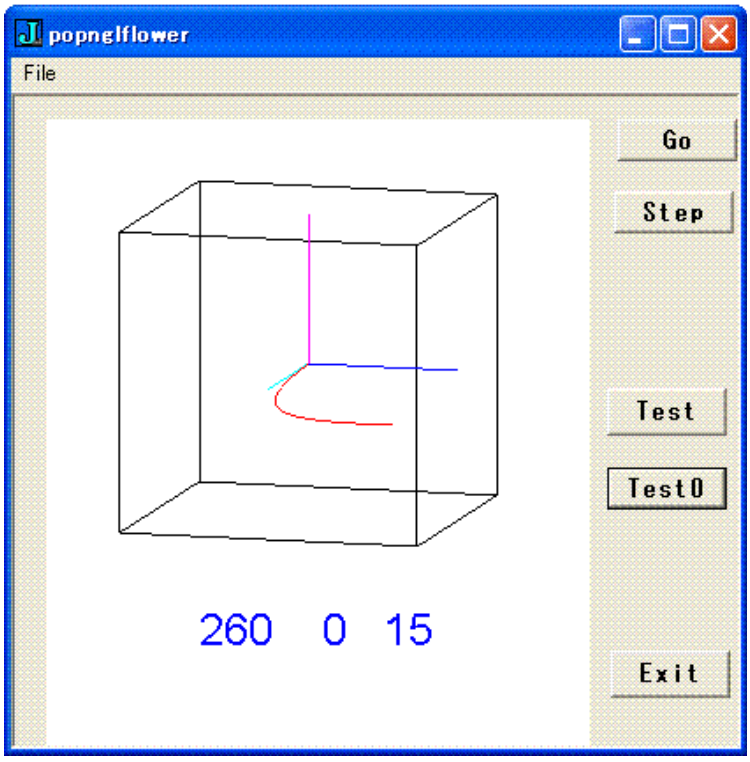


そのための結合順位を決めるプログラムを作った。

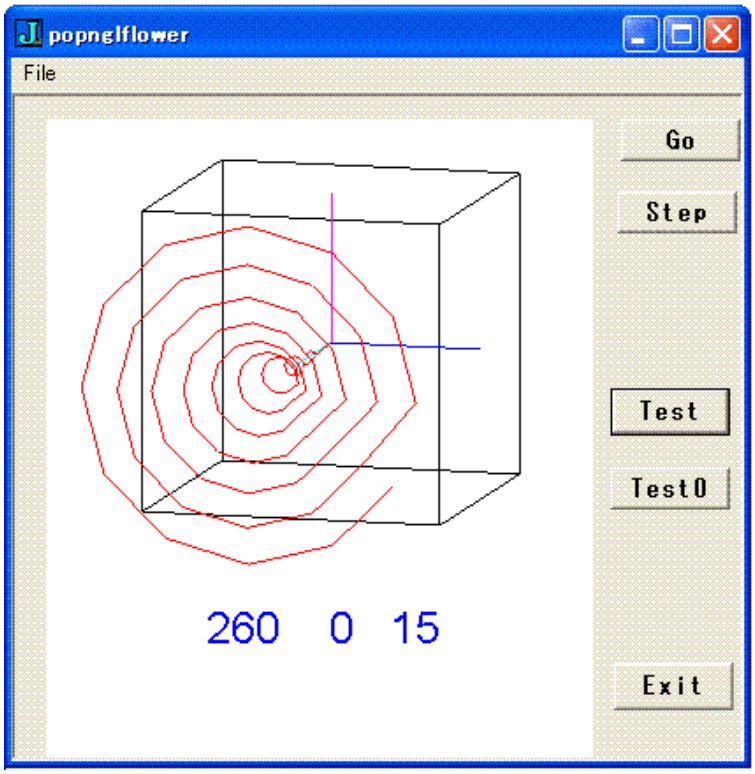
```
quad_round =: 3 : 0
:
RA =. x.
p =. y.
   VA =. p
   VB =. p + 1
   VC =. p + RA
   VD =. p + RA + 1
   VA, VB, VD, VC
)
```

３．プログラミング実行の過程－花のグラフィックスを一歩ずつ作っていく

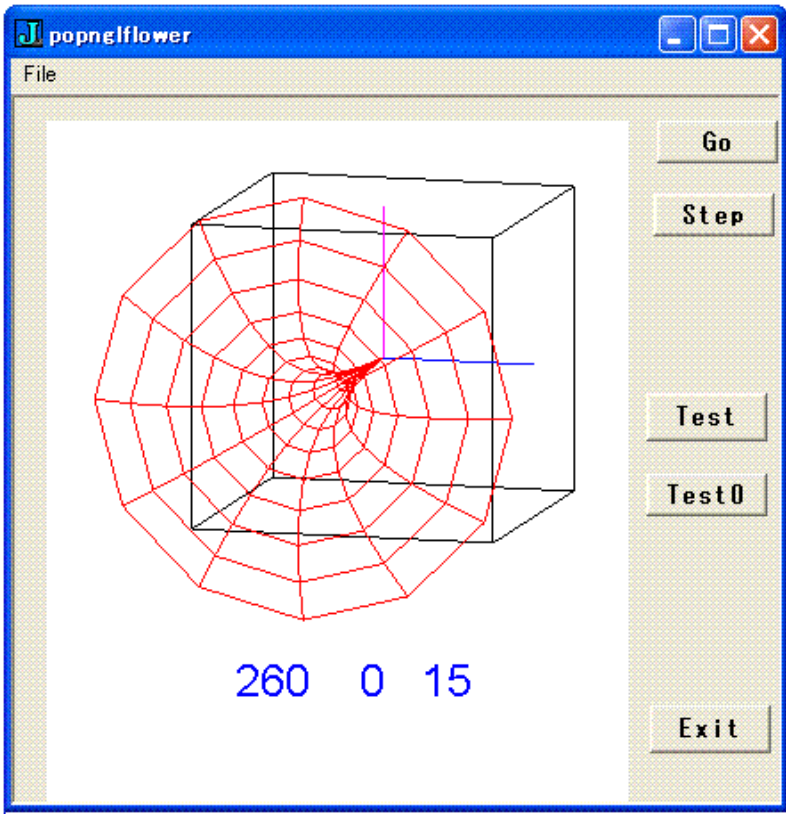
花のグラフィックスを前節のプログラミングの過程を追って示してみた。まず、元となる母線は次のようになる。（２．１）



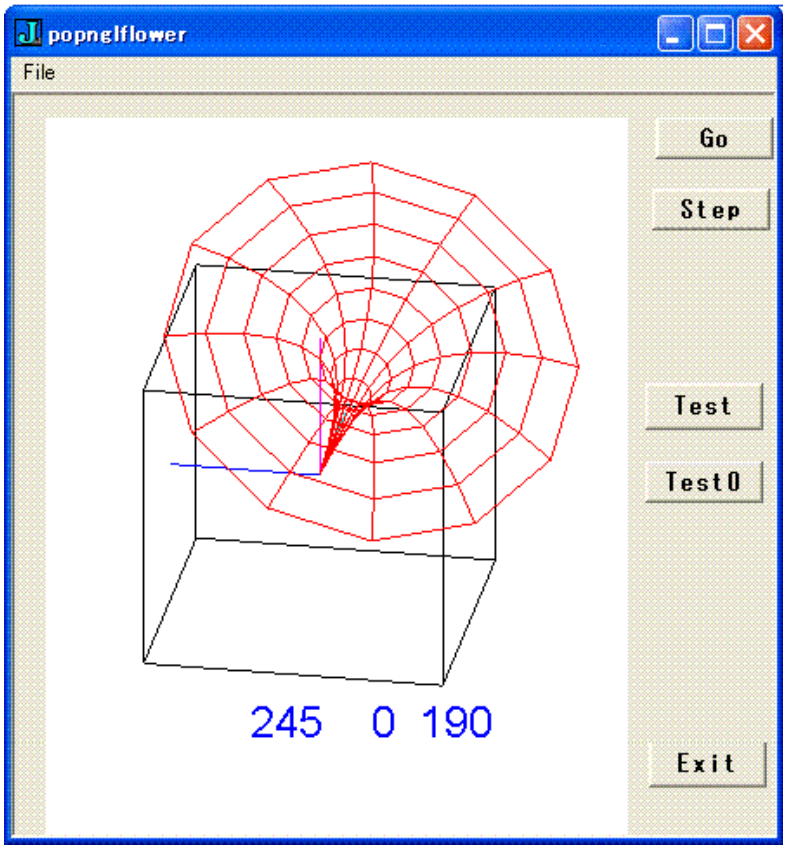
次に、母線を元に円錐台を描く。（２．２）



これでもよいが、これは一筆書きで描いたものである。
続いて、有限要素式にメッシュの微小図形の集合として描く。（２．３）



もちろん、OpenGL グラフィックスの図形であることから、キーコマンドX, Y, Zにより、別の視点から角度を変えて、傾けて見る、などいろいろなことが可能である。



４．さらに続く花のグラフィックス

ユリの花の OpenGL グラフィックスは、さらに続いて、照光表示を行って、花びらのウラオモテが分かるようにする予定でいた。しかし、時間切れになってしまった。メッシュ点の結合順序を反時計方向にするのは、そのための視点ベクトルに必要な処理である。

これらは、楽しみとして、次回にまわしたい。計画ではうまくいくはずが、実際には、なかなか思ったとおりに行かない。

趣味というのは、出来上がった作品を人に見せるというより、一歩ずつ作り上げて行く過程を楽しむというところにあるのだろう。

[1] 西川利男「Jによる花のグラフィクス-1 チューリップ」JAPLA 研究会資料 2009/8/5 夏の蓼科合宿
[2] 西川利男「Jによる花のグラフィクス-2 朝顔」JAPLA 研究会資料 2009/8/5 夏の蓼科合宿
[3] 戸川隼人「花のＣＧ－コンピュータ・グラフィックスによる花の描き方」サイエンス社(1988).

付録　OOP J-OpenGL プログラムの構築の手順

１．フォーム形式のクラス・ファイルを作る。
　これによって、OOP クラスファイルへの Copath が自動的に作られる。
　[File]-[New Class] New Class 作成の Wizard が開く。
　第１画面　[Next] で次へ
　第２画面　class names prefix を[p] 、 [Next] で次へ
　第３画面　 suffix を入力、例えば[opnglFlower]、[Next] で次へ
　第４画面　作成する Fileename を確認　f:¥j402¥user¥classes¥popnglFlower.ijs
　第５画面　Class GUI form を作る、Form Class にチェック、[Next] で次へ
　　Finished class file has been created and will be opened for editing [OK]
　第６画面　Select a template ３種類のクラスプログラムの内容を良く見た上で
　　base.ijs を選んで、[Next] で次へ、CTRL-w で Save する。

実行のための ijs プログラムは　[File]-[NewIJS]で
　例えば　　OpenGLN_Flower.ijs
　　load ’ f:¥j402¥user¥classes¥popnglFlower.ijs’
　　run =: 3 : 0
　　 d1 =: ’’ conew ’ popnglFlower’
　　)

２．クラスファイル classes¥popnglFlower.ijs に戻って、
　[Edit]-[Form Editor]を使って、手動で手直し、追加修正をする。

```
require ’gl3’
POPNGLFLOWER =: 0 : 0
 ----
xywh -- -- cc g isigraph ws_clipchildren ws_clipsiblings;
 (追加)
 ----
)

create =: 3 : 0
 ----
glaRC ’’ (追加)
 ----
)
```

３．その他、OpenGL の基本と必要項目を、追加プログラミングする。
　create へ初期値の追加
　popngl_g_paint
　　draw
　　drawtext
　　glaSwapBuffers
　popngl_g_char
　popngl_g_size