JAPLA 研究会資料 2012/8/2

ローレンツなどカオスの３Ｄグラフィックス
—J-OpenGL により、カオスの実行を段階的に観察する—

西川　利男

ローレンツ、レスターなどカオスの図形は、志村氏によりJの簡便かつ強力なグラフィックス機能を示す例としてたびたび紹介されている。これらのカオス現象の物理は、それ自身私にとってもおおいに興味をそそられるテーマである。

J の OpenGL グラフィックスを用いて、カオスの実行の詳細を途中経過を追って、特に視覚的に分かりやすく表示するシステムを構築した。

1.　非線形ダイナミックス－Lorentz の式と Roessler の式

位相空間（x, y, z）内の以下のような時間に関する微分方程式において、非線形（＝変数の交差項がある）の場合、時間の経過に従って決まるその軌道はカオス（Strange Attractor）として挙動する。

ここでは、有名な２つのカオスの式を取り上げた。

Lorentz の式

$$\frac{dx}{dy} = -sx - sy \qquad where$$
$$\frac{dy}{dt} = -xz + rx - y \qquad s = 10,\ r = 28,\ b = 8/3$$
$$\frac{dz}{dt} = xy - bz$$

MIT の気象学者 Lorentz は北ヨーロッパの気象変動のモデルとして、上の式をあらわした。

Roessler の式

$$\frac{dx}{dt} = -y - z \qquad where$$
$$\frac{dy}{dt} = x + a\,y \qquad a = 0.2,\ b = 0.2,\ c = 5.7$$
$$\frac{dz}{dt} = b\,x - (c - x)\,z$$

化学者 Roessler はもう少し簡単な上の式でもカオスとなることを示した。

２．J OpenGL によるカオスの３Ｄグラフィックス

今回のグラフィックス・プログラムの特徴は次のとおりである。

①　完成した図形だけでなく、カオスが成長していく途中経過を段階を追って表示するようにした。

② もちろん、OpenGL の 3D グラフィックス機能により、その時点でさまざまな方向より視点を変えてその形を観察できる。

③ 初期値や制御関数のパラメータ、さらにはローレンツの式やレスラーの式など制御関数そのものを引数として選択使用できるようにした。

4 そのため、プログラムはオブジェクト・指向プログラミング(OOP)として作成した。つまり、J のプログラムのうち OpenGL の大部分は OOP のクラス・プログラムとし、起動プログラムは制御式や初期値などごくわずかにし、OOP のインスタンスとして実行する。

すなわち、このようにして、より汎用的なカオス観察のグラフィックス・システムとして構築した。

３．J OpenGL のプログラムの構成

通常の J プログラマにとってなじみ難いのは、gl2 グラフィックスでは例えば gllines X0, Y0, X1, Y1 や glrect X0, Y0, X1, Y1 などのように、直接グラフィック命令を書くのに対して 、J OpenGL では、たとえ gl3 を使っても、OpenGL の定型的なブロック書式に従って、プログラミングすることにある。

ここでは、J OpenGL プログラム全体の構成について箇条書きで大要をのべる。

(1) require 'gl3'

(2) ウィンドウズ・グラフィックスのための form の作成、定義。

名詞（例えば A）として定義する。

```
A =: 0 : 0
pc a;
xywh 200 100 20 10;cc Run button;   NB. 実行ボタンの定義
xywh 10 10 200 200;cc g isigraph     NB. グラフ画面の定義
 - - - -
)
```

(3) ウィンドウズ・グラフィックスのプログラムの実行動詞の定義。

```
run =: 3 : 0
wd 'A'
いろいろな初期設定など
wd 'pshow;ptop'
 - - - -
)
```

(4) グラフィックスの表示

```
a_g_paint =: 3 : 0
glBegin GL_LINES
  glVertex X0, Y0, Z0
```

```
  glVertex X1, Y1, Z1
 glEnd "
 glaSwapBuffers "
 )
```

(5) コマンド文字入力 a_g_char

(6) レンダーリング＝投影方法の指示 a_g_size

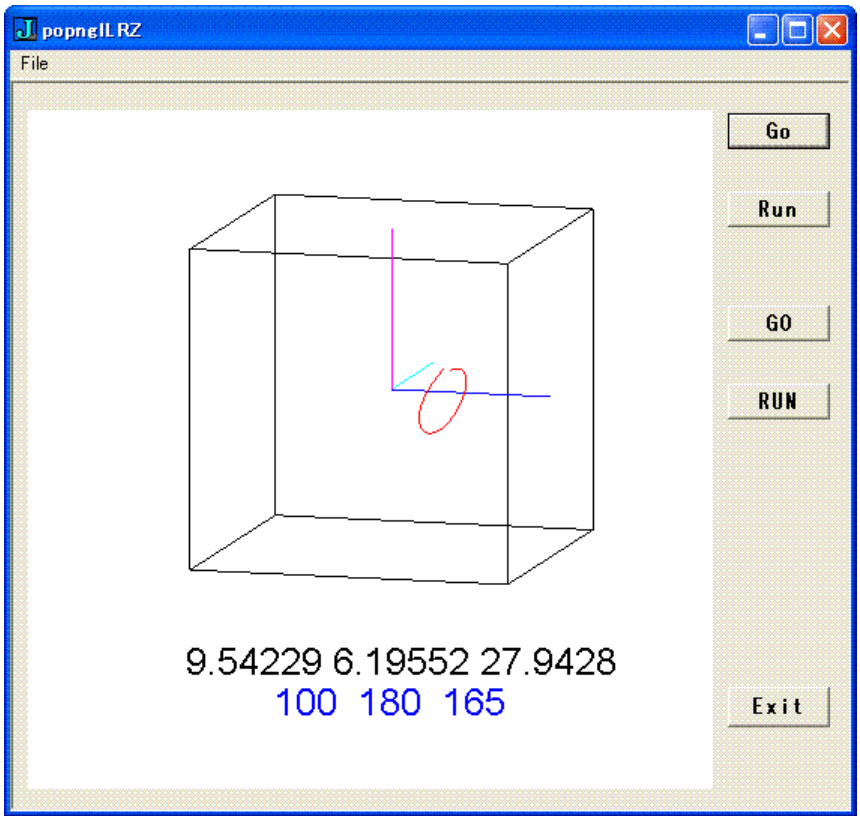
プログラムの詳細は、稿末に挙げてある。

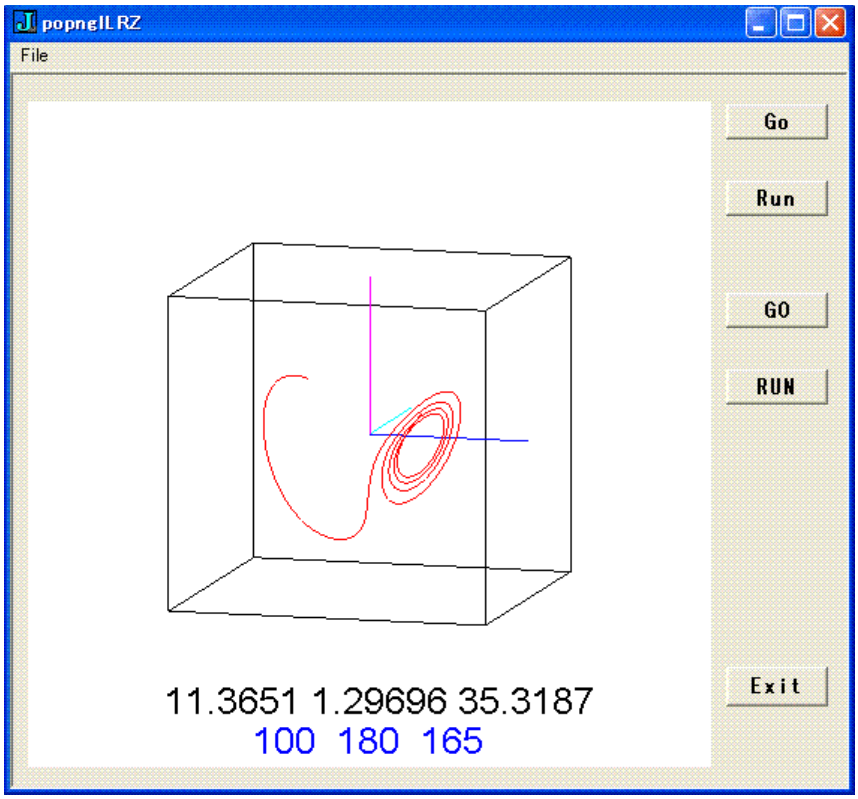
４．起動プログラム＝OOP のインスタンスプログラム

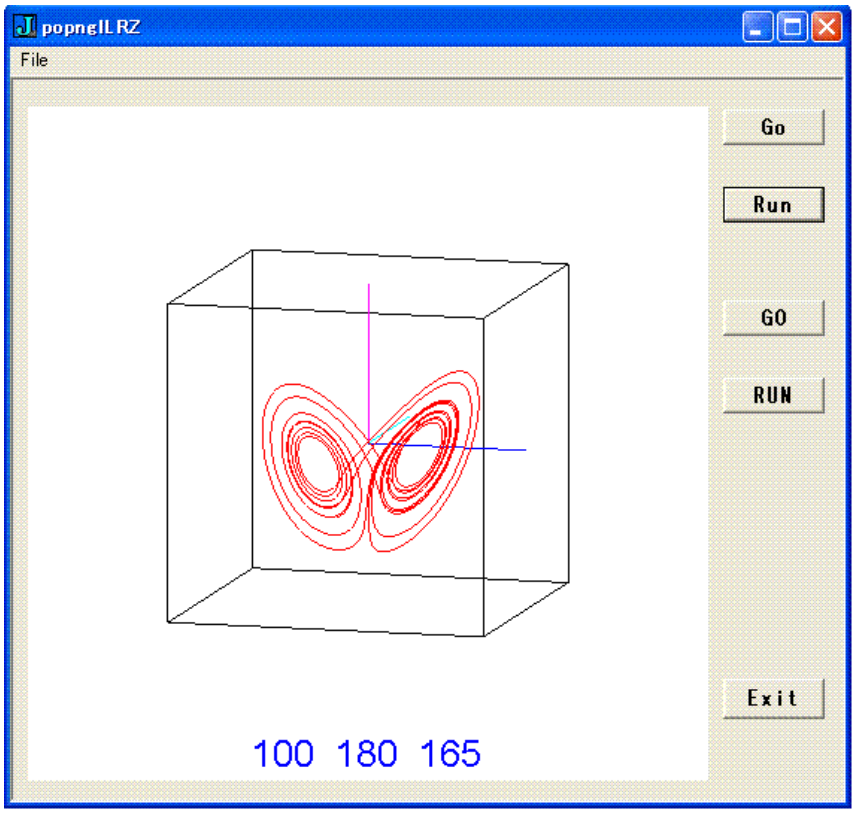
先にも述べたように、OpenGL のグラフィックスの大部分は OOP のクラス・プログラムとしたので、カオス式と初期値の設定の以下の OOP のインスタンス・プログラムで行う。

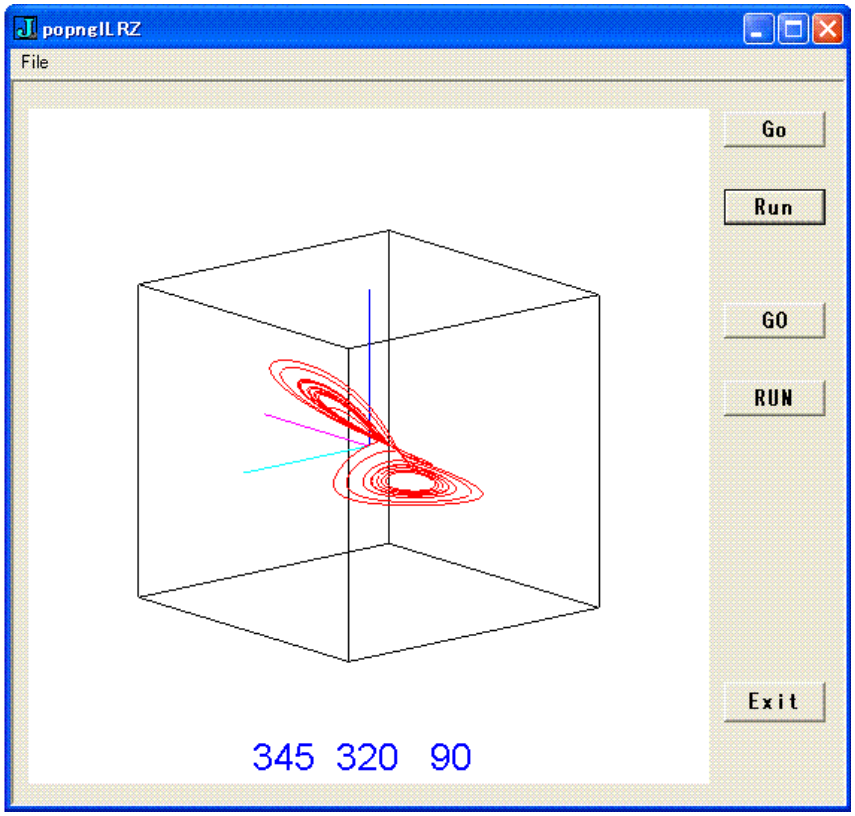
```
init  =:  8,  7, 28

dt =: 0.005
s =: 10
r =: 28
R =: 23
b =: 8 % 3

lz =: 3 : 0
('x';'y';'z') =. y.
X =. x + dt*s*(y-x)
NB. Y =. y + dt*((r*x) - (y + x*z))
Y =. y + dt*((R*x) - (y + x*z))
Z =. z + dt*((x*y) - b*z)
X, Y, Z
)
```

５．ローレンツのカオス

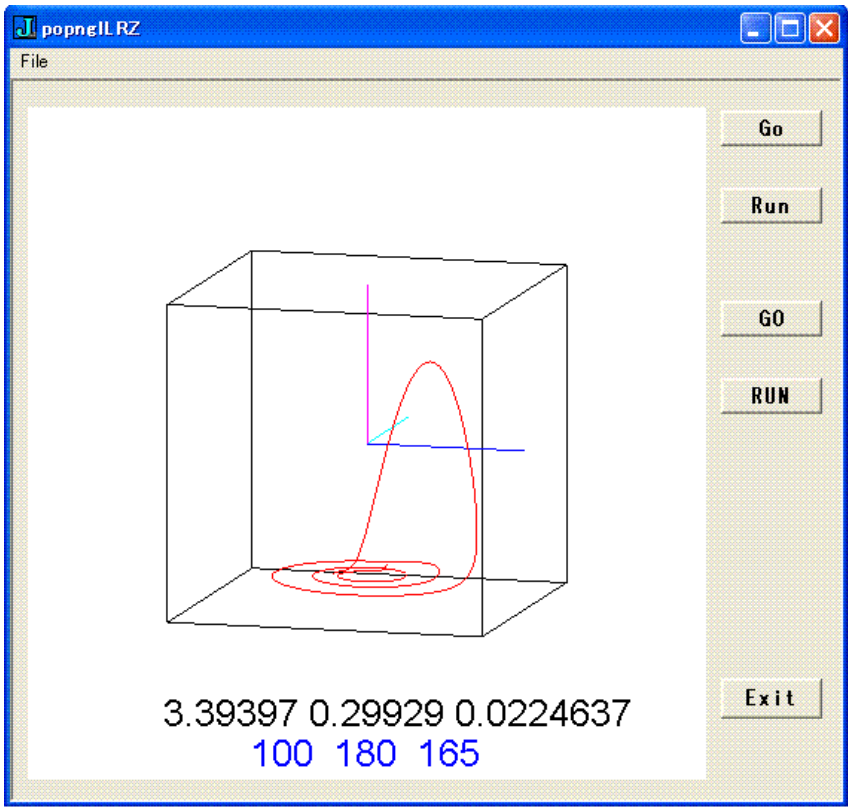

popnglLRZ
File
Go
Run
GO
RUN
Exit
9.54229 6.19552 27.9428
100 180 165

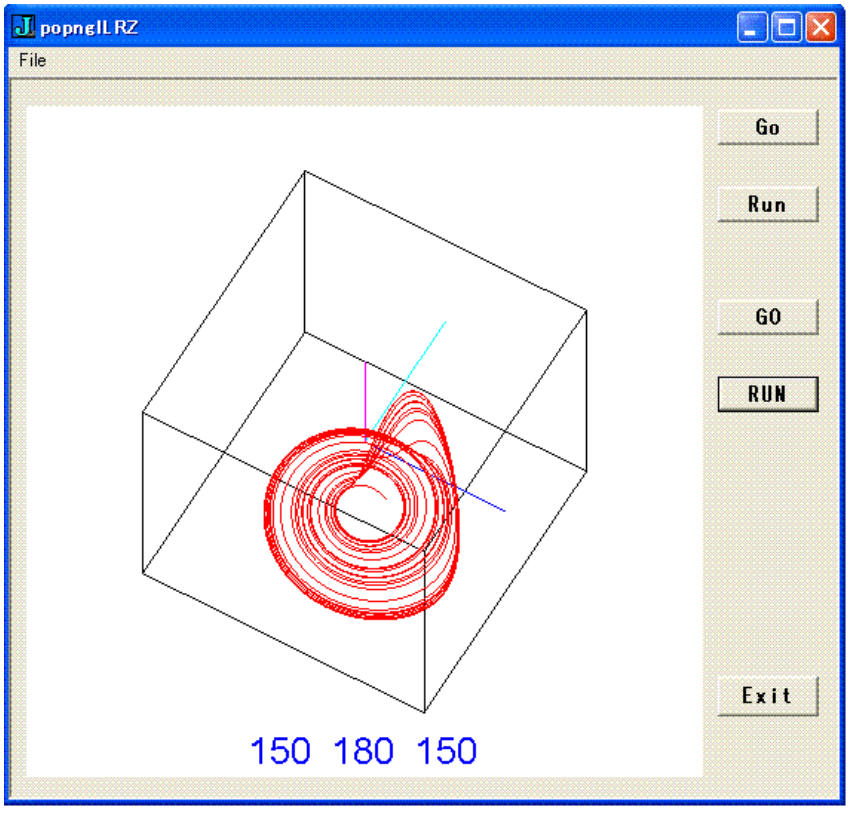

popnglLRZ
File
Go
Run
GO
RUN
Exit
11.3651 1.29696 35.3187
100 180 165


popnglLRZ
File
Go
Run
GO
RUN
Exit
100 180 165


popnglLRZ
File
Go
Run
GO
RUN
Exit
345 320 90

６．レスターのカオス


popnglLRZ
File
Go
Run
GO
RUN
Exit
3.39397 0.29929 0.0224637
100 180 165


popnglLRZ
File
Go
Run
GO
RUN
Exit
150 180 150

```
NB. OpGLN_Chaos.ijs
NB.  using 'user¥classes¥popnglLRZ.ijs' as OOP class file
NB. run init, inita, run initb
NB. run 1 2 3

NB. Lorenz Chaos System ================================
dt =: 0.005
s =: 10
r =: 28
R =: 23
b =: 8 % 3

init000 =: 0, 0, 0
init100 =: 1, 0, 0
init010 =: 0, 1, 0
init001 =: 0, 0, 1
init0 =: 8.5, 8.5, 27.3  NB. fixed point
init  =:  8,  7, 28
inita =:  0, _8, 25
initb =:  8, _8, 25
initc =:  8, _16, 25
initp0 =: (6*%:2), (6*%:2), 27
initp1 =: (6*%:2), (6*%:2), 28
initq0 =: (_6*%:2), (_6*%:2), 27
initq1 =: (_6*%:2), (_6*%:2), 26

lz =: 3 : 0
('x';'y';'z') =. y.
X =. x + dt*s*(y-x)
NB. Y =. y + dt*((r*x) - (y + x*z))
Y =. y + dt*((R*x) - (y + x*z))
Z =. z + dt*((x*y) - b*z)
X, Y, Z
)

NB. Roessler Chaos System ==============================
```

```
initros =: 1.62 1.36 1

ra =: 0.2
rb =: 0.2
rc =: 5.7

ros =: 3 : 0
('x';'y';'z') =. y.
X =. x - dt*(y + z)
Y =. y + dt*(x + ra*y)
Z =. z + dt*(rb + z * (x-rc))
X,Y,Z
)

Path =: 1!:40 ''

run =: 3 : 0
NB. left arg. = function
NB. right arg. = initial values
'lz' run y.
:
 load Path, 'user¥classes¥popnglLRZ.ijs'
 FUNC =: x.  NB. function
 DA =: y.    NB. initial x, y, z values
 d1 =: 'DA' conew 'popnglLRZ'
)
```

```
NB. Class Module called from OpGLN_Chaos.ijs
NB.   2014/7/23 by T.Nishikawa
coclass 'popnglLRZ'

require 'gl3'

POPNGLLRZ=: 0 : 0
pc popnglLRZ;
menupop "File";
menu new "&New" "" "" "";
menu open "&Open" "" "" "";
menusep ;
menu exit "&Exit" "" "" "";
menupopz;
xywh 238 179 34 12;cc cancel button;cn "Exit";
xywh 238 9 34 11;cc Go button;
xywh 238 32 34 11;cc Run button;
xywh 238 66 34 11;cc GO button;
xywh 238 89 34 11;cc RUN button;
xywh 5 8 228 201;cc g isigraph ws_clipchildren ws_clipsiblings;
pas 6 6;pcenter;
rem form end;
)

create=: 3 : 0
NameDA =: y.   NB. left argument of conew
NameFN =: FUNC_base_
smoutput NameFN
wd POPNGLLRZ
formhwnd=: wd'qhwndp'
NB. initialize form here
glaRC ''
R =: 100 180 165
SC =: 2
LS =: 0
Hid =: 0
```

```
RN =: 0
glaFont 'arial 30'
glaUseFontBitmaps 0 32 26 32
wd 'pshow;ptop'
)

destroy=: 3 : 0
wd'pclose'
codestroy''
)

popnglLRZ_cancel=:popnglLRZ_cancel_button=:popnglLRZ_close=:destroy

formselect=: 3 : 'wd''psel '',formhwnd'


NB. display the model picture ======================================
popnglLRZ_g_paint =: verb define
glClearColor 1 1 1 0
glClear GL_COLOR_BUFFER_BIT
draw ''        NB. draw box
drawcurve ''   NB. draw curve
drawtext''
glaSwapBuffers ''
)

connect =: 3 : 0
:
PP =. x.
C =. y.
i =. 0
while. i < #C
  do.
    glVertex (i{C) { PP
    i =. i + 1
end.
```

```
)

PBox =: ] ;._2 (0 : 0)
       1  1  1
      _1  1  1
      _1 _1  1
       1 _1  1
       1  1 _1
      _1  1 _1
      _1 _1 _1
       1 _1 _1
)

PCord =: ] ;._2 (0 : 0)
       1  0  0
       0  0  0
       0  1  0
       0  0  0
       0  0  1
       0  0  0
)

PBox =: 0.6 * ". PBox

PCord =: 0.6 * ". PCord

NB. Draw Vertex
draw =: verb define
glMatrixMode GL_MODELVIEW
glLoadIdentity ''
glTranslate 0 0 0
glRotate R ,. 3 3 $ 1 0 0 0
glScale 3#SC
if. LS = 0
  do.
    glPolygonMode GL_FRONT, GL_LINE   NB. Paint line
```

```
  else.
    glPolygonMode GL_FRONT_AND_BACK, GL_FILL   NB. Paint full
end.
glPolygonMode GL_BACK, Hid{GL_LINE, GL_POINT  NB. Hidden


NB. Draw Coordinate Box ================================
glBegin GL_LINE_STRIP
  glColor 0 0 0 0
  PBox connect 0, 1, 2, 3, 0, 4, 5, 6, 7, 4
glEnd ''
glBegin GL_LINE_STRIP
  glColor 0 0 0 0
  PBox connect 3, 7, 6, 2
glEnd ''
glBegin GL_LINE_STRIP
  glColor 0 0 0 0
  PBox connect 1, 5
glEnd ''

NB. Draw Coordinate Line ===============================
glBegin GL_LINE_STRIP
  glColor 0 0 1 0        NB. X-axis
  PCord connect 0, 1
glEnd ''
glBegin GL_LINE_STRIP
  glColor 0 1 1 0        NB. Y-axis
  PCord connect 2, 3
glEnd ''
glBegin GL_LINE_STRIP
  glColor 1 0 1 0        NB. Z-axis
  PCord connect 4, 5
glEnd ''
)

NB. project the picture on the screen ==================
```

```
popnglLRZ_g_size =: verb define
wh =. glqwh ''
glViewport 0 0, wh
glMatrixMode GL_PROJECTION
glLoadIdentity ''
glOrtho _2.5 2.5 _2.5 2.5 _2.5 2.5
NB. gluPerspective 60, (%/wh), 1 30
)

NB. Key-in Command ====================================
NB. x, y, z, X, Y, Z for rotation along axis
popnglLRZ_g_char =: verb define
k =. 0 { sysdata
R =: 360 | R + 5 * 'xyz' = 0 { sysdata
R =: 360 | R - 5 * 'XYZ' = 0 { sysdata
NB. l(larger), s(smaller) for scaling
SC =: SC * 1 + 0.25 * 'l' = 0 { sysdata  NB. l => larger
SC =: SC * 1 - 0.25 * 's' = 0 { sysdata  NB. s => smaller
NB. g for run stepwise
RN =: RN + 10 * 'g' = 0 { sysdata
RN =: RN - 10 * 'G' = 0 { sysdata
LS =: ('s' = k) { LS, -. LS
Hid =: ('h' = k) { Hid, -. Hid
glpaintx''
)

NB. indicate rotated angle values x, y, z in degree ============
drawtext =: verb define
glMatrixMode GL_MODELVIEW
glLoadIdentity ''
glColor 0 0 0 0
glRasterPos _1.5 _2.1 0
glCallLists ": LZZ
glColor 0 0 1 0
glRasterPos _1 _2.4 0
glCallLists 5 ": R
```

```
)

popnglLRZ_help_button =: verb define
wd 'mb OpenGL *Press keys, x/X, y/Y, z/Z rotate, s: line or solid, h: line
hidden toggle.'
wd 'setfocus g'
)

drawcurve =: 3 : 0
glBegin GL_LINE_STRIP
  glLineWidth 8.0
  glColor 1 0 0 0
  PD connect CD
glEnd ''
glpaintx''
)

NB. Lorenz System ========================================


popnglLRZ_test_button=: 3 : 0
smoutput BDATA =: ". NameDA,'_base_'
smoutput (": BDATA), ' = ', (": # ": BDATA), ' characters'
smoutput 'Sqrt = ', ": *: BDATA

smoutput lz_base_ BDATA
NB. == exec(".) should take string form, verb and noun ====
smoutput 'exec func value in base'
smoutput ". NameFN,'_base_', ' ',  (": BDATA)
)

popnglLRZ_GoL_button=: 3 : 0
if. RN = 0
 do.
     LZ =: lz_base_ ^: (i.20) ". NameDA,'_base_'
     NB. LZ =: lz_base_ ^: (i.20) init_base_
```

```
     DA =: (0, 0, _0.5) +"(1) 0.02 * LZ
     LZZ =: {: LZ
 else.
     LZ =: LZ, lz_base_ ^: (i.20) LZZ
     DA =: (0, 0, _0.5) +"(1) 0.02 * LZ
     LZZ =: {: LZ
 end.
NB. smoutput RN
NB. smoutput $LZZ
NB. smoutput LZZ
NB. smoutput '----------'
PD =: DA
CD =: i. (RN+1)*20
NB. drawcurve '' is not needed
glpaintx ''
wd'setfocus g'
RN =: RN + 1
)

popnglLRZ_RunL_button=: 3 : 0
LZ =. lz_base_ ^: (i.4000) init_base_
DA =: (0, 0, _0.5) +"(1) 0.02 * LZ
PD =: DA
CD =: i.4000
NB. drawcurve '' is not needed
glpaintx ''
wd'setfocus g'
)

popnglLRZ_Go_button=: 3 : 0
 BDATA =: ". NameDA, '_base_'
 Func =: NameFN, '_base_'
if. RN = 0
  do.
   LZ =: ". Func, '^:(i.30)', (":BDATA)
   DA =: (0,0,_0.5) +"(1) 0.02*LZ
```

```
    LZZ =: {: LZ
  else.
    LZ =: LZ, (''.Func, '^:(i.30)', (':LZZ) )
    DA =: (0,0,_0.5) +"(1) 0.02*LZ
    LZZ =: {: LZ
end.
PD =: DA
CD =: i.(RN+1)*30
glpaintx ''
wd 'setfocus g'
RN =: RN + 1
)


popnglLRZ_Run_button=: 3 : 0
 BDATA =: ''. NameDA, '_base_'
 Func =: NameFN, '_base_'
 LZ =: ''. Func, '^:(i.3000)', (':BDATA)
   DA =: (0,0,_0.5) +"(1) 0.02*LZ
 PD =: DA
 CD =: i.3000
glpaintx ''
wd 'setfocus g'
)

popnglLRZ_GO_button=: 3 : 0
 BDATA =: ''. NameDA, '_base_'
 Func =: NameFN, '_base_'
if. RN = 0
  do.
   LZ =: ''. Func, '^:(i.1000)', (':BDATA)
   DA =: (0,0,_0.5) +"(1) 0.04*LZ
   LZZ =: {: LZ
  else.
   LZ =: LZ, (''.Func, '^:(i.1000)', (':LZZ) )
   DA =: (0,0,_0.5) +"(1) 0.04*LZ
```

```
   LZZ =: {: LZ
end.
PD =: DA
CD =: i.(RN+1)*1000
glpaintx ''
wd 'setfocus g'
RN =: RN + 1
)

popnglLRZ_RUN_button=: 3 : 0
 BDATA =: ". NameDA, '_base_'
 Func =: NameFN, '_base_'
 LZ =: ". Func, '^:(i.30000)', (":BDATA)
   DA =: (0,0,_0.5) +"(1) 0.04*LZ
 PD =: DA
 CD =: i.30000
glpaintx ''
wd 'setfocus g'
)
```