JAPLA 研究会資料 2019/8/2

J の gl2 における複素数グラフィックスーその１

西川　利男

Ｊでは複素数演算がすでに常備されていて、普通の数値計算と同様に行える。また、志村正人氏により何回も紹介されているが、plot グラフィックスつまり jd ライブラリにおいても複素数を用いた処理が可能である。

以前から私としては、Ｊの基本グラフィックスである gl2 においてもあらわな形で複素数計算で処理を行うグラフィックスを構築したいと考えていた。

実は、フラクタルやインディラ・グラフィックスのためには複素数グラフィックスは必須の欠かせないテクニックである。まだ極めて初期的なレベルであるが、その第一歩としてのツールを試みた。

１．複素数グラフィックスのための幾何学

以前､私が訳出した｢初めてのフラクタル｣[1]の p. 160 付録 A から複素数の幾何学の要点を、箇条書きとして示す。

［1］H. ラウヴェリエール、西川利男訳「初めてのフラクタル」丸善(1996)．

$z' = F(z)$

(1) $z' = z + c$　平行移動

(2) $z' = i\,z$　$\pi/2$の回転

(3) $z' = (\cos\alpha + i\sin\alpha)z$　角αの回転

(4) $z' = a\,z$　中心拡大

(5) $z' = \bar{z}$　反転　すなわち実軸に関する鏡像

(6) $z' = (\cos\alpha + i\ \sin\alpha)\bar{z}$　反転して、原点のまわりに$\alpha/2$の方向だけ回転

(7) (3)と(5)を組み合わせて、複素数の乗算

任意の点z_0を中心とした一般的相似変換は次のようになる。

$z' = z_0 + c(z - z_0)$

２．複素数グラフィックスのＪプログラム

先に報告したオリンピック・エンブレムのグラフィックス[2]のＪプログラムを元に手直しすることで、今回のＪ複素数グラフィックスを作成した。

[2]　西川利男「J のタイリング・グラフィックスによりオリンピック・エンブレムを描く」　JAPLA 研究会資料 2017/6/17
西川利男「J のタイリング・グラフィックスにより オリンピック・エンブレムを描く－続き　（完成版）」JAPLA 研究会資料 2017/8/5

今回はとりあえず、複素平面の上で、複素数の形で、点を入力して、三角形を作成する。これを元に前ページに示した移動、拡大縮小、回転をおこない、つまり複素平面の上で、相似変換を行う。このとき、すべてＪの複素数演算で計算処理を行い、グラフィックスとして表示する。

ここで、Ｊの以下の複素数演算プリミティブが、非常に効果的に使われる。

複素数の生成 x + i y　=> x　j.　y　=>　x j y

複素数から、実部、虚部を取り出す　+.　x j y => x , y

もちろん、複素数 x j y の形のままで、加減乗除の演算が可能である。

したがって、複素数の座標値に対して、前ページの移動、拡大縮小、回転の演算はすべて、複素数のままで行い、グラフィック表示の直前で、実部、虚部を取り出し、グラフィック処理を行うのである。

たとえば、具体的な数値で示してみると、つぎのようになる。

```
AA =: 4j3, BB =: 2j6, CC =: 2j3
```

と入力された複素数値に対し、以下の ABCA は三角形を形づくるための４点の複素数値の集まりである。

```
ABCA =: AA, BB, CC, AA
```

続いて、つぎの定義された関数 colorpolylines により、青い三角形が表示される。

```
(0, 0, 255) colorpolylines , +. ABCA
```

なお関数 colorpolylines は先のオリンピック・エンブレムで使用したものである。

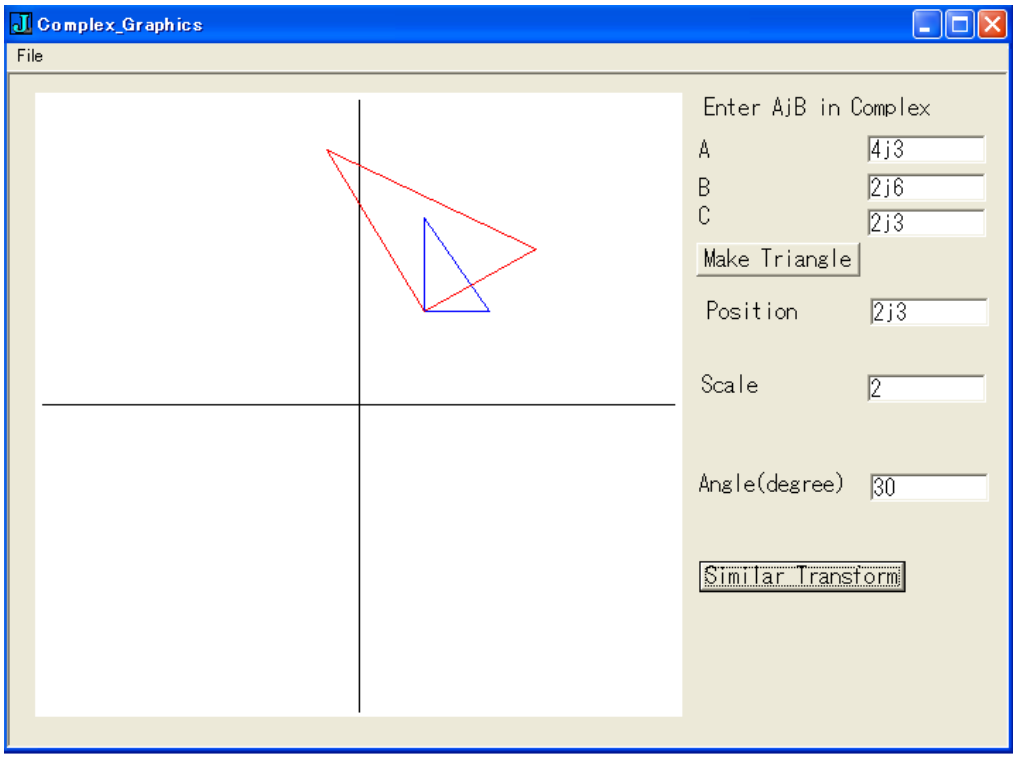
相似変換は以下の複素数の演算で行われる。

```
ABCA4 =:  POS + ((cosd ANG) j. (sind ANG)) * SS * (- POS) + ABCA
```

ここで、回転の角度は ANG、拡大縮小は SS、処理の位置は POS である。これは前ページの最後に示した複素数による相似変換の式に相当する。

Ｊのプログラムコーディングは最後に示した。

３．複素数グラフィックスの実際



J のプログラムリスト

```
NB. Create Triangle ========================================
complexgraphics_EntA_button=: 3 : 0
AA =: ". EntA
)
complexgraphics_EntB_button=: 3 : 0
BB =: ". EntB
)
complexgraphics_EntC_button=: 3 : 0
CC =: ". EntC
)
complexgraphics_RunComp_button=: 3 : 0
ABCA =: AA, BB, CC, AA
 (0, 0, 255) colorpolylines , +. ABCA
glshow ''
)
```

```
NB. Position ========================================
complexgraphics_EntPOS_button=: 3 : 0
POS =: ". EntPOS
)
complexgraphics_RunPOS_button=: 3 : 0
ABCA0 =: (- POS) + ABCA
(0, 0, 255) colorpolylines , +. (- POS) + ABCA
glshow ''
)
NB. Scaling =========================================
complexgraphics_EntSC_button=: 3 : 0
SS =: ". EntSC
)
complexgraphics_RunScale_button=: 3 : 0
ABCA1 =: SS * ABCA0
(0, 255, 0) colorpolylines , +. ABCA1
glshow ''
)
complexgraphics_SimTrans_button=: 3 : 0
NB. Similar Transform Calc. / ABCA to ABCA4 in complex ======================
ABCA4 =:  POS + ((cosd ANG) j. (sind ANG)) * SS * (- POS) + ABCA
if. RUN = 0 do. (255, 0, 0) colorpolylines , +. ABCA4 end.
if. RUN = 1 do. (0, 255, 0) colorpolylines , +. ABCA4 end.
if. RUN = 2 do. (0, 255, 255) colorpolylines , +. ABCA4 end.
RUN =: RUN + 1
glshow ''
)

colorpolylines =: 3 : 0
:
glrgb x.
glbrush ''
glpen 1 0
gllines , sz adj0 y.
)
```

```
NB. ================================================
adj0 =:  3 : 0    NB. centered
1 adj0 y.
:
NB. 500 + 100 * x. * y.
(x0, y0) + "(1) > cut2 100 * x. * y.  NB. y-axis down
NB. (500, 400) + "(1) > cut2 100 * x. * y.  NB. y-axis down
)

adj =:  3 : 0     NB. start left down
1 adj y.
:
NB. (500&+)@(100&*)
200 + 40 * x. * y.
)

NB.  cut2 i.10
NB. +---+---+---+---+---+
NB. |0 1|2 3|4 5|6 7|8 9|
NB. +---+---+---+---+---+
cut2 =: 3 : 0
y =. ((-:#y.),2)$y.
<"(1) y
)


NB. eg. 30 rotate figure(x0, y0, x1, y1, .. ) around (0, 0)
rot =: 3 : 0
:
'x y' =. y.
t =. x.
((x*cosd t) - (y*sind t)), ((x*sind t) + (y*cosd t))
)

rotate =: 3 : 0
:
```

```
ANG =. x.
RXY =. cut2 , > y.
RX =. {."(1) > ANG rot L:0 RXY
RY =. {:"(1) > ANG rot L:0 RXY
, |: RX ,: RY
)

NB. eg. (1, 2) shift x, y
shift =: 3 : 0
:
'Xs Ys' =. x.
, (Xs, Ys) +"(1) > cut2 y.
)

colorpolygon =: 3 : 0
:
glrgb x.
glbrush ''
glpen 1 0
glpolygon , sz adj0 y.
)

colorpolylines =: 3 : 0
:
glrgb x.
glbrush ''
glpen 1 0
gllines , sz adj0 y.
)
```