JAPLA 研究会資料 2017/3/11

# 群論と幾何学模様の対称性－その２
# －Ｊグラフィックス壁紙群－

西川　利男

先の JAPLA 研究会の１月例会では、Ｊの壁紙群（格子群）の入り口だけだがトライしてみた［1, 2, 3］。 だがもう少し完成したものへと、いろいろ手直しを行った。
はじめにＪグラフィックスと幾何学とについて、私見を少々述べたい。

## １．Ｊによる描画（作図）のための幾何学

Ｊプログラミングシステムと数学とのかかわり合いについて、大きく２つの分野があると思う。これをＪの基本は動詞と名詞であるという哲学から比較してみる。

| | 動詞 | 名詞 |
|---|---|---|
| ・J Calculus … 計算するＪ | 関数 | 値 |
| ・J Graphics … 図形を扱うＪ | 幾何学操作 | 図形 |

ここで幾何学操作の内容は群論操作も含めて、次のものである。

- 合同　　＝ 移動して重ねてしらべる
- 相似　　＝ 拡大、縮小した後、移動して重ねてしらべる
- 回転
- 鏡映
- 滑り移動

これ以外にも、いろいろな幾何学操作が考えられよう。
幾何学の伝統的な考え方では、暗黙で手動の操作によりこれを行う、としている。しかしながら、今ではコンピュータ画面上の幾何学操作が実用上極めて重要である。
例えば、合同の操作は、画面上の図形をマウスでドラッグして、重ねて見ることができれば最も良いが、残念ながらＪではこれは不可能である。代わりにＪのウィンドウ・システムでは、ボタンを押してＪの動詞を実行させることでこの操作を行う。
ところで、ユーザプログラムレベルで、このようなことを行える環境はＪシステム以外にはないと私は思う。このような意味において、Ｊは現代の幾何学にとってかけがえのない強力なツールだといえよう。

## ２．Ｊ壁紙グラフィックス(前回版)の改良

前回の JAPLA(2017/1/21)版のＪグラフィックスシステムは、いろいろな点で私としては不満足であった。具体的には、壁紙のパターン図形を描画したり、回転や移動するルーチンが、渾然一体となってしまっている。これらをもっとすっきりとしたい。
前節で述べたように、壁紙のパターン図形は名詞とし、回転や移動などの群論操作は動詞として、連続操作などもできるようにしたい。もう１つ「J」なる文字図形についても不完全であり、見直しを行った。このようにしたのが今回の改訂版である。

［1］西川利男「群論と幾何学模様の対称性－その１」JAPLA 研究会, 2017/1/21
［2］伏見康治、安野光雅、中村義作「美の幾何学」早川書房(2010).
［3］M.A.Armstrong,”Group and Symmetry”,Springer(1988).

佐藤信哉訳「対称性からの群論」

## ３．gl2 の gllines と glpolygon の差異－「J」図形の作成についての改善

前回のの不自然さの原因は、gl コマンドの微妙な差異にあることが解った。ここで、gllines と glpolygon の差異についてあらためて検討した。

gllines は座標値を単純に、結んで図形を描くだけである。一方、glploygon は座標値を結んで図形を描き、その内部を指定した色で塗りつぶす。つまり、glploygon では最初と最後の座標同志を結んで閉じた図形にするが、gllines では必ずしも閉じることなく、開いた線の集まりである。したがって、n 角形を描くためには、glpolygon では引数として n 個の座標値でよいが、gllines では開始の座標値を最後に付け加えることが必要になる。

```
gllines X0, Y0, X1, Y1, ... , Xn, Yn, X0, Y0
glpolygon  X0, Y0, X1, Y1, ... , Xn, Yn
```

さらに、gllines では、一筆書きとして描くので、２つ以上の図形を合わせるときには注意が必要である。

これらを考慮して、改善した「J」の文字図形の座標値はつぎのように改めた。

```
JP1 =:  , _1 * |: > (cos th);(sin th)
JP2 =:  ,  |: 0.5 * (1, _1) *"(0 1) > (cos th);(sin th)
JP =: JP2, (_1, 0), JP1, (1, 2), (0.5, 2)
```

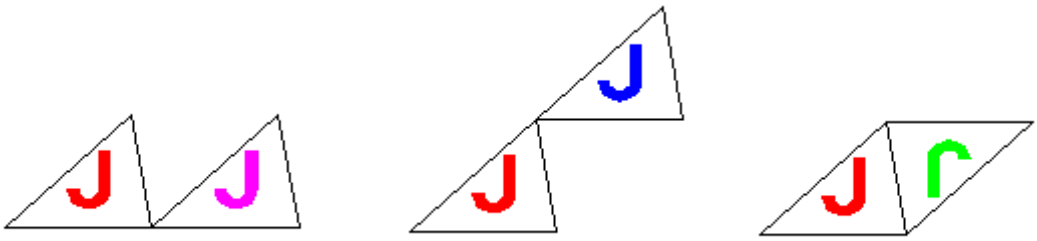
## ４．Ｊによる壁紙格子群の操作

前回の報告では、対称性 p2 のうち、２回軸回転操作 p2 を行っただけであった。今回、滑り移動操作 pm まで入れて、壁紙格子群として完結させる。これにより平面敷き詰め格子を生成することができる。

基本となるパターンは、前回と同様に、以下のような不等辺３角形の中に上の「J」なる文字図形を入れた図形とした。

```
PP =: _2, _1
QQ =: 1.5, _1
RR =: 1, 2
FIGT =: PP, QQ, RR, PP
```

この基本図を元につぎのような３通りの滑りコピー操作で、平面敷き詰めを生成するようにした。



なお、ここで J の図形回転のルーチン rotate, rot 及び図形移動のルーチン shift を使用したが、これらのプログラム定義の詳細については、別報告で述べる[4]。

[4] 西川利男「J によるペンローズ・タイルのグラフィックス」JAPLA 研究会, 2017/3/11

## ５．J壁紙格子群p2グラフィックスの実際

改良したJ壁紙グラフィックスの実行のようすは次のようになる。

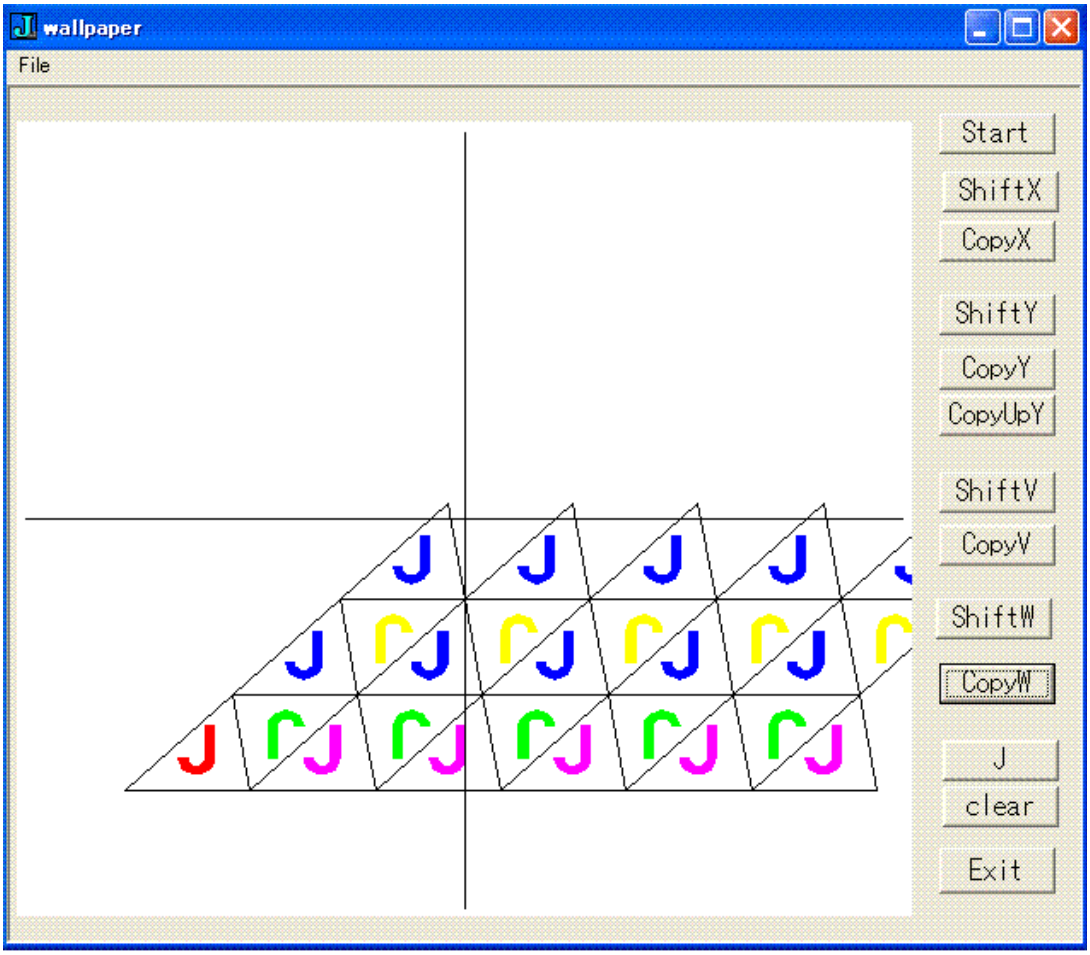
(1) Startボタンを押すと、アカで基本のスタート図を表示する。

(2) ShiftXボタンを押すと、ムラサキで右方向の滑り移動を行い、CopyXボタンで次々と右方向に滑り移動コピーを続けて行く。

(3) ShiftYボタンを押すと、アオで一段上に滑り移動を行い、CopyYボタンで右方向に滑り移動コピーを続けて行く。

(4) ShiftVボタンを押すと、ミドリでp2回転を行い、CopyVボタンでは、鏡映した図を元に右方向に滑り移動コピーを続けて行く。

(5) ShiftWボタンを押すと、キイロで一段上で、p2回転を行い、CopyWボタンでは、この段で右方向に滑り移動コピーを続けて行く。




以上のようにして、壁紙の貼り付けがおこなわれる。なお、このプログラムでは、壁紙格子群操作のなかで、あえて対称性p2というゆるい対称性で行うようにした。

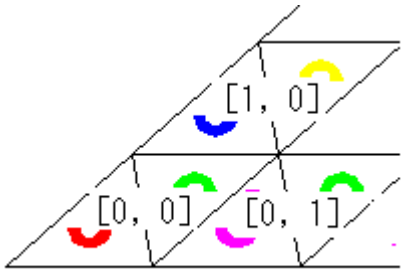
対称性がもっと高い、たとえばスタートの基本図形が長方形で、正方格子の場合などでは、ずっと容易に対応できる。

## ６．壁紙格子群とペンローズ・タイルー私のトライアル・コメント

おわりに、壁紙格子群とペンローズ・タイル（別稿）のＪプログラミングを通して得た、私の一つの見方を述べたい。

**・格子群とは**

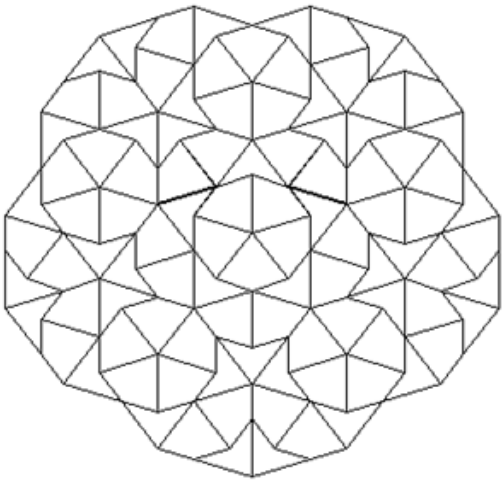
あるパターンを元に平面上を、左右、縦横に広がり繰り返されたものである。このとき、人間の目には、図形の大きさは意識せず、並び方（＝対称性）を全体として見るだけである。




実際にグラフィック画面に図形として描くときには
直交座標として、原点(0, 0)から正負に広がる値の組である。
つまり
「格子群とは、タテヨコの直交ないし斜交形式でありながら
一方では、中心から広がっていく極形式である」
という ambivalent な(２重性をもった)考え方といえる。

**・ペンローズ・タイルとは**

対称性をもっとゆるくして、
「極形式で広がる、拡張されたより一般的な壁紙格子群」
と見ることができる。



プログラム・リスト

```
NB. 2017/2/7, revised 2017/2/11 ======================
NB. Figure Triangle
PP =: _2, _1
QQ =: 1.5, _1
RR =: 1, 2
FIGT =: PP, QQ, RR, PP

NB. J-charcter xy-values
JP1 =:  , _1 * |: > (cos th);(sin th)
JP2 =:  ,  |: 0.5 * (1, _1) *"(0 1) > (cos th);(sin th)
JP =: JP2, (_1, 0), JP1, (1, 2), (0.5, 2)
JCHT =: 0.5 * JP

wallpaper_Start_button=: 3 : 0
(0 0 0) colorpolylines  FIGT
(255 0 0) colorpolygon JCHT
draw_xy ''
glshow ''
Xfig =: FIGT
Xjch =: JCHT
)

wallpaper_ShiftX_button=: 3 : 0
XS =. QQ - PP

FIGX =. ,  XS +"(1) > cut2 FIGT
(0 0 0) colorpolylines FIGX

JCHTX =. , XS +"(1) > cut2 JCHT
(255 0 255) colorpolygon JCHTX
draw_xy ''
glshow ''
)

wallpaper_CopyX_button=: 3 : 0
Xshift =. (QQ - PP)

Xfig =: , Xshift +"(1 1) > cut2 Xfig
(0 0 0) colorpolylines Xfig

Xjch =: , Xshift +"(1 1) > cut2 Xjch
(255 0 255) colorpolygon Xjch
glshow ''
)
```

```
wallpaper_ShiftY_button=: 3 : 0
YS =. RR - PP

FIGY =: ,  YS +"(1) >  cut2 FIGT
(0 0 0) colorpolylines FIGY

JCHTY =. , YS +"(1) > cut2 JCHT
(0 0 255) colorpolygon JCHTY
draw_xy ''
glshow ''
Yfig =: FIGY
Yjch =: JCHTY
)

wallpaper_CopyY_button=: 3 : 0
Yshift =. (QQ - PP)

Yfig =: , Yshift +"(1 1) > cut2 Yfig
(0 0 0) colorpolylines Yfig

Yjch =: , Yshift +"(1 1) > cut2 Yjch
(0 0 255) colorpolygon Yjch
glshow ''
)

wallpaper_CopyUpY_button=: 3 : 0
Yshift =. (RR - PP)

Yfig =: , Yshift +"(1 1) > cut2 Yfig
(0 0 0) colorpolylines Yfig

Yjch =: , Yshift +"(1 1) > cut2 Yjch
(0 0 255) colorpolygon Yjch
glshow ''
)


wallpaper_ShiftV_button=: 3 : 0
QR =. QQ - RR
QR2 =. (QQ + RR) % 2

FIGV0 =.   180 rotate , > FIGT
FIGV =. , (2*QR2) +"(1) > cut2 FIGV0
(0 0 0) colorpolylines FIGV
```

```
JCHTV0 =. 180 rotate , > JCHT
JCHTV  =. , (2*QR2) +"(1) > cut2 JCHTV0
(0 255 0) colorpolygon JCHTV

draw_xy ''
glshow ''
XYfig =: FIGV
XYjch =: JCHTV
)

wallpaper_CopyV_button=: 3 : 0
XYshift =. (QQ - PP)

XYfig =: , XYshift +"(1 1) > cut2 XYfig
(0 0 0) colorpolylines XYfig

XYjch =: , XYshift +"(1 1) > cut2 XYjch
(0 255 0) colorpolygon XYjch
glshow ''
)

wallpaper_ShiftW_button=: 3 : 0
SS =: (0, 1){FIGY     NB. revised 2/11
TT =: (2, 3){FIGY     NB. revised 2/11
ST =: SS + TT         NB. revised 2/11

FIGW0 =.   180 rotate , > FIGT
FIGW =. , (ST) +"(1) > cut2 FIGW0
(0 0 0) colorpolylines FIGW
JCHTW0 =. 180 rotate , > JCHT
JCHTW  =. , (ST) +"(1) > cut2 JCHTW0
(255 255 0) colorpolygon JCHTW
draw_xy ''
glshow ''
STfig =: FIGW
STjch =: JCHTW
)

wallpaper_CopyW_button=: 3 : 0
STshift =. (QQ - PP)
STfig =: , STshift +"(1 1) > cut2 STfig
(0 0 0) colorpolylines STfig

STjch =: , STshift +"(1 1) > cut2 STjch
```

```
(255 255 0) colorpolygon STjch
glshow ''
)
```