バーンスレーの IFS
点描の世界

M.Shimura
JCD02773@nifty.ne.jp

2007 年 6 月 15 日

目次

1 図形の拡大・縮小と移動

■2D での画像変換の基礎

$$translation\ add\ -0.5, 2 \quad rotation\ 90^o \quad Scaling\ 0.3$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} Scale \rightarrow \begin{bmatrix} 0.3 & 0 & 0 \\ 0 & 0.3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$Rotate \rightarrow \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.3 & 0 & 0 \\ 0 & 0.3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$Translate \rightarrow \begin{bmatrix} 1 & 0 & -0.5 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.3 & 0 & 0 \\ 0 & 0.3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & -0.5 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.3 & 0 & 0 \\ 0 & 0.3 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & -0.3 & -0.5 \\ 0.3 & 0 & 2 \\ 0 & 0 & 1 \end{bmatrix}$$

$translation\ add -0.5, 2 \quad rotation\ 90^o \quad Scaling\ 0.3$

	transrate and rotate	Refrection	Scale
$\begin{bmatrix} A & 0 \\ 0 & 1 \end{bmatrix}$	$\begin{bmatrix} cos\phi & -sin\phi & 0 \\ sin\phi & cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} s & 0 & 0 \\ 0 & t & 0 \\ 0 & 0 & 1 \end{bmatrix}$
	Counterclockwise rotation angle ϕ	Refrection in $y = x$	Scale x by s and y by t

2 IFS

IFS *Iteration Function Syatem*

2.1 C.Reiter

C.Reiter の Lab に入っている Fractal Visualization より。

```
   rt=: t0‘t1‘t2@.(?@3:)

   rt 0.9 0.9 1 NB. initial
0.95 0.45 1

   rt^:(i.5) 0.9 0.9 1
    0.9     0.9 1
   0.45    0.45 1
  0.725   0.225 1
 0.3625  0.6125 1
0.18125 0.80625 1
```

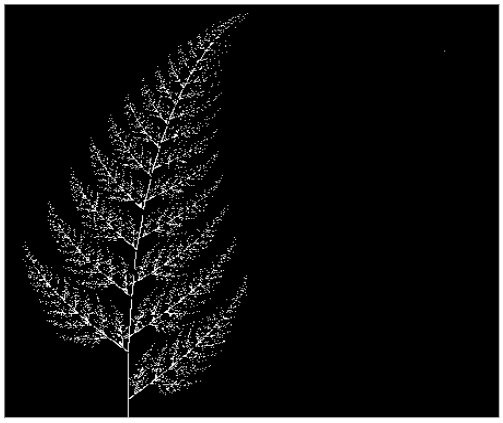

図 1 fern

■計算

1. 変換マトリクス

mk_gm は　ポジションを指定すればマトリクスに変形する my_utility (8 は 1 に固定）

```
0 1 2
3 4 5
6 7 8
```

```
NB. my formura
 F0=: 4 6 mk_gm  0.16 0.25
 F1=:  0 1 3 4 6 7 mk_gm 0.85 _0.04  0.04 0.85  0.0375 0.17
 F2=: 0 1 3 4 6 7 mk_gm  0.2 0.23 _0.26 0.22 0.2 0.1025
 F3=: 0 1 3 4 6 7 mk_gm _0.15 0.26 0.28 0.24  0.2875 _0.021

   F0;F1;F2;F3
+-----------+-------------+-------------+---------------+
|   0    0 0|  0.85 _0.04 0|  0.2   0.23 0| _0.15   0.26 0|
|   0 0.16 0|  0.04  0.85 0|_0.26   0.22 0|  0.28   0.24 0|
|0.25    0 1|0.0375  0.17 1|  0.2 0.1025 1|0.2875 _0.021 1|
+-----------+-------------+-------------+---------------+
```

2. 計算式 T0,T1,T2,T3

```
T0=: (+/ . *)&F0

   T0 0.9 0.9 1
0.25 0.144 1
   T1 0.9 0.9 1
0.8385 0.899 1
   T2 0.9 0.9 1
0.146 0.5075 1
   T3 0.9 0.9 1
0.4045 0.429 1
```

3. 計算式を確率で選び、20000 回ほど繰り返す。
4. tacit のループ。初期値は 0.9 0.9 1

```
TR=: T0‘T1‘T2‘T3 @.(?@9: { 0 1 1 1 1 1 1 2 3"_)
TR ^:(i. x) 0.9 0.9 1
)
```

■viewmat で描画

1. 3 列の値のうち最後の列を落とす。
2. 1，2 列を整数化して、500 倍し、1 を打つアドレスとする。

3. 500×500 の 0 の行列に先のアドレスで 0 を 1 に変換する。
4. 方向を整えて、viewmat で見る。

```
a=: }:"1  rt^:(i.125) 0.9 0.9 1
viewmat |.|: 1 (;/roundint 499 * a) } 500 500$0
```

A:　viewmat はどれくらいの大きさまでいけるのか。
P:　さあ
A:　色は
P:　0/1 は白黒　カラーも OK

2.2　J の本格的なグラフィックスで

■glpixel で描画　天地逆だがご愛敬

repeat　10 万回。　キャンバス　1000×1000

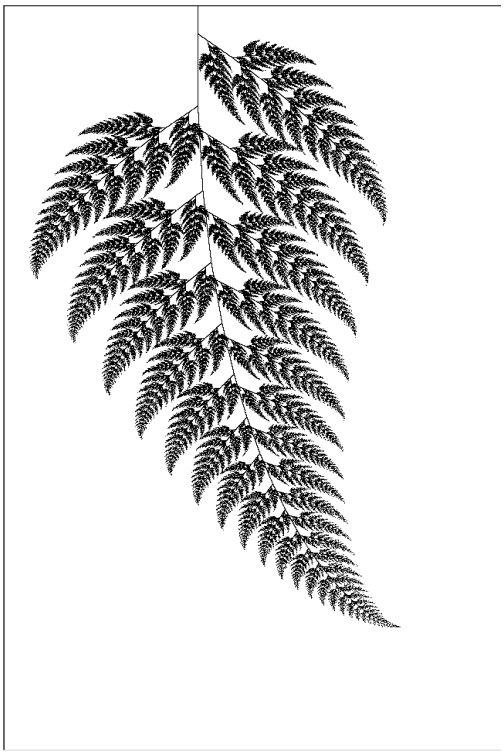

図 2　fern

```
run_ifs0=: 3 : 0
NB. usage: u F0;F1;F2;F3
DAT=:  1000 *  1000000 mk_ifs0_sub y
gopen ''
```

```
gclear''
glpixel  DAT
)
```