ヒルベルトとシェルピンスキーのフラクタル曲線

SHIMURA Masato
jcd02773@nifty.ne.jp

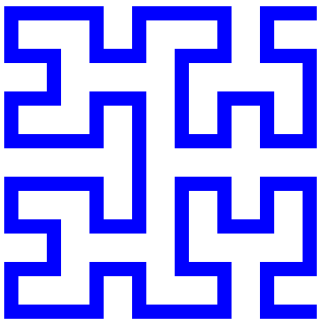
2019 年 12 月 13 日

目次

1 ヒルベルト曲線

ヒルベルト（1862-1943）の尊名は聞いていても数学の門外漢には 20 世紀の始まりに 23 の課題を提示したこと以外にさほどなじみがないが、高木貞二の 1932 年 10 月のゲッチンゲンでのヒルベルト訪問記がネットに上がっていたので糸口としたい。

J のの Studio/Demos のグラフィックスにヒルベルト曲線が入っており、Script の解読を試みた。



ヒルベルト曲線は 1891 年のまだケニッヒスベルグ時代に提唱したものでペアノの曲線の翌年である。空間充填曲線は後にフラクタルに含まれることとなった。

このスクリプトはビューアーの viewmat で見ることができる 0/1 のマトリクスの数字の羅列である。

ここの達人の作成した J のスクリプトの中心部分は超難解な一行で書かれている。

1. 初期値=土台 HP (名詞で y として用いる)

```
   HP
0 0 0
0 1 0
0 0 0
```

2. ここをベースにして反復して変形していく

```
     (,.|:)HP
0 0 0 0 0 0
0 1 0 0 1 0
0 0 0 0 0 0
```

3. 線を描くのではなく、0/1 で正方形のボックスを現し、2 色で表示する。1 が白抜き。
4. Program (hp)

```
hp=: 3 : '(|.,]) 1 (0 _2 _2 ,&.> _2 _1 0 + #y) } (,.|:) y'
```

5. Usage:

```
 NB.   WB viewmat hp ^:7 HP
```

6. ループの仕組み
   - 一回

```
      hp HP
0 0 0 0 0 0
0 1 1 1 1 0
0 1 0 0 0 0
0 1 0 0 0 0
0 1 1 1 1 0
0 0 0 0 0 0
```

   - 2 回

```
      hp ^:2 HP
0 0 0 0 0 0  0 0 0 0 0 0
0 1 1 1 1 1  1 1 0 0 1 0
0 1 0 0 0 0  0 1 0 0 1 0
0 1 0 0 0 0  0 1 0 0 1 0
0 1 1 1 1 0  0 1 1 1 1 0
0 0 0 0 1 0  0 0 0 0 0 0

0 0 0 0 1 0  0 0 0 0 0 0
0 1 1 1 1 0  0 1 1 1 1 0
```

```
0 1 0 0 0 0  0 1 0 0 1 0
0 1 0 0 0 0  0 1 0 0 1 0
0 1 1 1 1 1  1 1 0 0 1 0
0 0 0 0 0 0  0 0 0 0 0 0
```

- 3 回

```
     hp ^:3 HP
0 0 0 0 0 0  0 0 0 0 0 0  0 0 0 0 0 0  0 0 0 0 0 0
0 1 1 1 1 1  1 1 0 0 1 1  1 1 1 1 1 0  0 1 1 1 1 0
0 1 0 0 0 0  0 1 0 0 1 0  0 0 0 0 1 0  0 1 0 0 0 0
0 1 0 0 0 0  0 1 0 0 1 0  0 0 0 0 1 0  0 1 0 0 0 0
0 1 1 1 1 0  0 1 1 1 1 0  0 1 1 1 1 0  0 1 1 1 1 0
0 0 0 0 1 0  0 0 0 0 0 0  0 1 0 0 0 0  0 0 0 0 1 0

0 0 0 0 1 0  0 0 0 0 0 0  0 1 0 0 0 0  0 0 0 0 1 0
0 1 1 1 1 0  0 1 1 1 1 0  0 1 0 0 1 1  1 1 0 0 1 0
0 1 0 0 0 0  0 1 0 0 1 0  0 1 0 0 1 0  0 1 0 0 1 0
0 1 0 0 0 0  0 1 0 0 1 0  0 1 0 0 1 0  0 1 0 0 1 0
0 1 1 1 1 1  1 1 0 0 1 0  0 1 1 1 1 0  0 1 1 1 1 0
0 0 0 0 0 0  0 0 0 0 1 0  0 0 0 0 0 0  0 0 0 0 0 0

0 0 0 0 0 0  0 0 0 0 1 0  0 0 0 0 0 0  0 0 0 0 0 0
0 1 1 1 1 1  1 1 0 0 1 0  0 1 1 1 1 0  0 1 1 1 1 0
0 1 0 0 0 0  0 1 0 0 1 0  0 1 0 0 1 0  0 1 0 0 1 0
0 1 0 0 0 0  0 1 0 0 1 0  0 1 0 0 1 0  0 1 0 0 1 0
0 1 1 1 1 0  0 1 1 1 1 0  0 1 0 0 1 1  1 1 0 0 1 0
0 0 0 0 1 0  0 0 0 0 0 0  0 1 0 0 0 0  0 0 0 0 1 0

0 0 0 0 1 0  0 0 0 0 0 0  0 1 0 0 0 0  0 0 0 0 1 0
0 1 1 1 1 0  0 1 1 1 1 0  0 1 1 1 1 0  0 1 1 1 1 0
0 1 0 0 0 0  0 1 0 0 1 0  0 0 0 0 1 0  0 1 0 0 0 0
0 1 0 0 0 0  0 1 0 0 1 0  0 0 0 0 1 0  0 1 0 0 0 0
0 1 1 1 1 1  1 1 0 0 1 1  1 1 1 1 1 0  0 1 1 1 1 0
0 0 0 0 0 0  0 0 0 0 0 0  0 0 0 0 0 0  0 0 0 0 0 0
```

7. 中心部

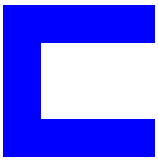
```
0 _2 _2 ,&. > _2 _1 0 +   # HP
```

```
+---+----+----+
|0 1|_2 2|_2 3|
+---+----+----+
```

8. ユニットの半分

```
      1  (0 _2 _2 ,&. > _2 _1 0 +   # HP)}(,. |:)HP
0 1 0 0 0 0
0 1 1 1 1 0
0 0 0 0 0 0
```

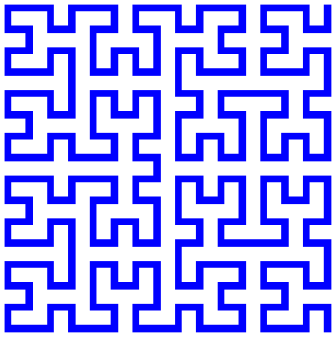
9. ユニットの作成



```
   (|. ,])
0 0 0 0 0 0
0 1 1 1 1 0
0 1 0 0 0 0
0 1 0 0 0 0
0 1 1 1 1 0
0 0 0 0 0 0
```

10. このユニットを一部変形しながら重ねている。
11. 名人の技は観賞するのみである。

```
WB viewmat  hp ^:4 HP
```



1.1　色彩の部分

WB=: 255,:0 0 255 NB. white/blue color triples

```
   WB
255 255 255   NB. white
  0   0 255   NB. blue
```

```
   WR=: 256 #. 255,:255 0 0 NB. whitered color RGB integers/
```

2　シェルピンスキーのフラクタル曲線

シェルピンスキー (1882-1969) はポーランド人で集合や数論に功績がありポーランド学派の樹立に貢献した。第一次大戦ではロシアの捕虜になったが、ロシアの数学者がモスクワ近くの収容所に移送を手助けし、研究がつづけられた。第二次大戦ではナチスの捕虜になったが間もなく連合軍に解放された。

マイケル・ブラッドリー「数学を現代化した預言者達」には略伝と共に次のフラクタル図形が載っていた。

これは 0/1 のグラフィックスが描けそうだ。

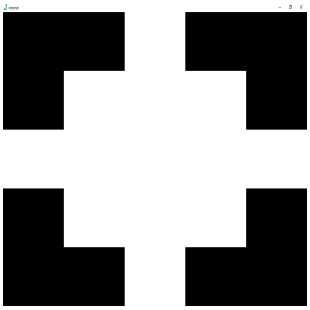
1. アドレス。これがアメンド（修正）時のマトリクスのアドレスになる

```
   Dadress=: 0 2;1 1;1 2;1 3;2 0;2 1;2 2;2 3;2 4;3 1;3 2;3 3;4 2
      Dadress
+---+---+---+---+---+---+---+---+---+---+---+---+---+
|0 2|1 1|1 2|1 3|2 0|2 1|2 2|2 3|2 4|3 1|3 2|3 3|4 2|
+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

2. 最初のピース。グラフィックスではなくマトリクスの座標なので左上が (0,0) で raw,column で数える。

```
   1    ( Dadress)}5 5 $0
0 0 1 0 0
0 1 1 1 0
1 1 1 1 1
0 1 1 1 0
0 0 1 0 0
```

クロスの木を反復させてた。余白の L を 4 回回転させる方法の方が本来の手法かもしれない。



3. 13×13 で 4 個のピースが収まる。13×13 のマトリクス上に描く。13 = 5×2+3

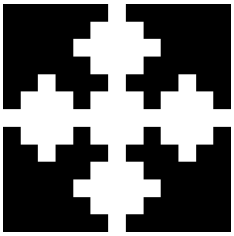
```
   1    ( 0 4 + L:0 Dadress)}13 13 $0
0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 1 1 1 0 0 0 0 0
0 0 0 0 1 1 1 1 1 0 0 0 0
0 0 0 0 0 1 1 1 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0
```

4. 取りあえずの Script を作る

```
calc_sier10=: 3 : 0
NB. Usage:(WB) viewmat (13 13 $ 0) calc_sier0 0 6  // raw column / origin is 0
```

```
mat=.  13 13 $ 0
'N E S W'=: y + (L:0) Posdiff
mat=. 1 (N + L:0 Dadress)} mat  NB. N
mat=. 1 (E + L:0 Dadress)} mat  NB. E
mat=. 1 (S + L:0 Dadress)} mat  NB. S
mat=. 1 (W + L:0 Dadress)} mat  NB. W
mat=. 1 (Cadress+ (L:0) y)}mat
)
```

```
 viewmat (13 13 $ 0) calc_sier10 0 6
```



5. 次は 29 × 29 で、13 × 2 + 3 となる。センターは 14
   ツリー 4 個づつのお花の剣山を中心に置く積み重ねとなる。反復は一回ずつの手書きである。スマートなポジションの値を求める手法は求められず、tacit 型には纏められていない。
6. Script
   （a）アドレス

```
   Dadress=: 0 2;1 1;1 2;1 3;2 0;2 1;2 2;2 3;2 4;3 1;3 2;3 3;4 2  NB. Cross Tre
   Cadress=: 5 _1;5 0;5 1;6 _1;6 0;6 1;7 _1;7 0;7 1               NB.Center Box
```

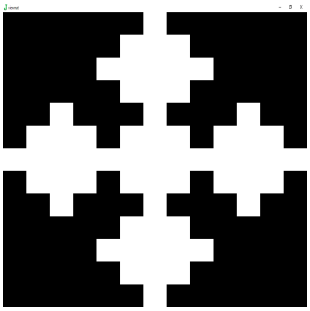
   （b）最初のピース

```
   calc_sier0=: 3 : '1 Dadress } 5 5 $ 0'
```

   （c）4 個のピース

```
   Posdiff=: 0 _2;4 2;8 _2;4 _6  NB. N E S W y の値と最初の一トの差分
   viewmat  (13 13 $ 0) calc_sier1 0 6
```

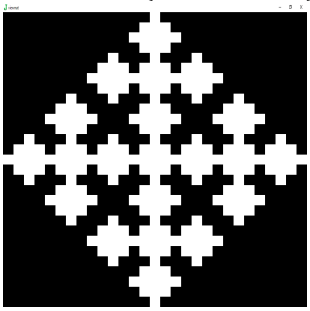
   次々と再利用するとき、マトリクスのサイズと中心点を変数として利用するため両項で定義し x でマトリクスのサイズ、y で頂点座標を与える



```
calc_sier1=: 4 : 0
NB.  viewmat (13 13 $ 0) calc_sier1 0 6//OK
mat=. x NB. 13 13 $ 0  NB. ought to use x for expand
AD1=.  y + (L:0) Posdiff NB. 0 _2;4 2;8 _2;4 _6  NB. N E S W
for_ctr. i. 4 do.
 mat=. 1 ((ctr{AD1) + L:0 Dadress)} mat  NB. N
end.
mat=. 1 (Cadress+ L:0 y)}mat
)
```

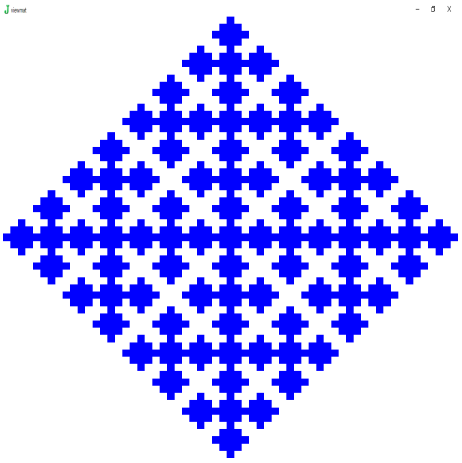
（d）16 個のピース。4 倍に拡大し、センターボックス一個は別途描く

```
viewmat (29 29 $ 0) calc_sier2 0 14
```



```
calc_sier2=: 4 : 0
NB. Usage: viewmat  (29 29 $0) calc_sier2 0 14 // OK
mat =. x NB. 29 29 $ 0
AD2=.  y + (L:0) 0 0; 8 8; 16 0;;8 _8  NB. N E S W
for_ctr. i. 4 do.
mat =. mat  calc_sier1 ctr{AD2
end.
Cpos=. ({.AD2) + (L:0) 8 0   NB. 8 0 OK , but why ??
1 (Cadress+ (L:0) Cpos )}mat
)
```

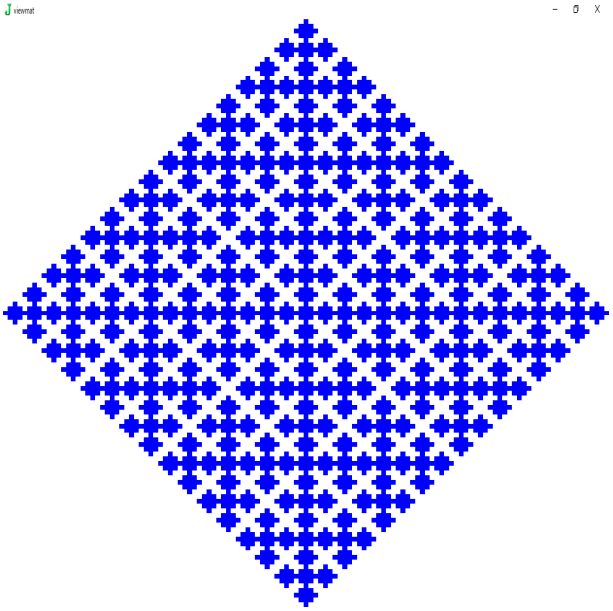
（e）64 個のピース。更に 4 倍に拡大。センターボックス 1 個は個別に



```
calc_sier3=: 4 : 0
NB. Usage:viewmat -. (61 61 $ 0) calc_sier3 0 30
mat =. x
AD3=. y + (L:0) 0 0; 16 16;32 0; 16 _16  NB. N E S W
for_ctr. i. 4 do.
 mat =. mat calc_sier2 ctr{AD3
end.
Cpos=. ({.AD3) + (L:0) 24 0   NB. 24 0 OK , but why ??
1 (Cadress+ (L:0) Cpos )}mat
)
```

(f) 256 個のピース。パラメーターの規則性も見えてきた。

```
viewmat -. (125 125 $ 0) calc_sier4 0 62
```



```
calc_sier4=: 4 : 0
```

```
NB. Usage:viewmat -. (125 125 $ 0) calc_sier4 0 62
mat =. x
AD4=. y + (L:0) 0 0; 32 32;64 0; 32 _32  NB. N E S W
for_ctr. i. 4 do.
 mat =. mat calc_sier3 ctr{AD4
end.
Cpos=. ({.AD4) + (L:0) 56 0   NB. 24 0 OK , but why ??
1 (Cadress+ (L:0) Cpos )}mat
)
```

7. パラメーター一覧。なんとなく規則が見えてきた。ロシアの人形・マトリョウシカのような入れ子にして一本のプログラムにすることもできるが返って複雑になりそうだ。

No	le	ft	ri	ght	N		E		S		W		Cen	ter
1	13	13	0	6									y	y
2	29	29	0	14	0	0	8	8	16	0	8	−8	8	0
3	61	61	0	30	0	0	16	16	32	0	16	−16	24	0
4	125	125	0	62	0	0	32	32	64	0	32	−32	56	0

References

マイケル・J・ブラッドリー/松浦俊輔訳「数学を開いた人々 4　数学を現代化した預言者達」青土社 2009