反復関数集合によるフラクタル画像生成

Fractal Image Generation with Iterated Function Set

浅井　貴浩*
Takahiro　ASAI

要　　旨

反復関数集合を用いることにより，少ないデータから多彩なフラクタル画像を得る方法を開発した．従来のフラクタル画像生成方法，例えばジュリア・マンデルブロー集合を図示するものや自己相似関数を用いるHata-Hatchinsonの方法では，いずれも反復過程において常に一種類のパラメータしか用いないために生成される画像は単純なものであった．複数のパラメータからなる反復関数集合を作成し，反復の各段階において集合の要素であるパラメータを選択して適用することにより，少ないデータからでも，より多彩な画像を生成することを可能とした．

ABSTRACT

A new fractal image generation method using a generation function set is developed. The generation function set which has several different parameters is defined, one of which to be selected at each recursion. The usual methods use only one parameter, such as the Julia-Mandelbrot and the Hata-Huchinson methods. As a result, various kinds of unique class of non-linear fractal images from a few data are constructed.

* 研究開発本部　村山塾
Murayama Juku, Research and Development Group

1. 背景と目的

近年ポスター等のデザインや映画，テレビCM等において計算機システムにより作成されるCG(Computer Graphics)画像が多用されており，新しい種類の画像を生成するためのアルゴリズムへの期待が高まっている．またDTP(Desk Top Publishing)システムやInternetの普及に伴い，個人による文書やWebページの作成等に使用するデジタルコンテンツとしてのCG画像，それらを作成するアプリケーションへの需要が高まっている．

こうした背景の中でMandelbrotにより提唱されたフラクタル幾何学[1]はCG画像を生成するアルゴリズムの一つとして現在広く使われている．フラクタル画像生成の方法は大きく2種類に分けることができ，一方はfBm[2]等を利用して不確定要素を付加することによりフラクタル性を持つ形状を生成するもので，ランダムフラクタルあるいは非決定論的フラクタルと呼ばれる．これは主に地形の起伏や雲等の自然物の形状モデルを作成するために使われている．もう一方は数式により求められるフラクタル集合を画像として描くもので，不確定要素がないことから決定論的フラクタルと呼ばれる．反復関数集合による方法[3]，Hata-Hatchinsonの方法[4]，L-systemによる方法[5]やJulia，Mandelbrot集合を図示する方法[1]等がこれに当たる．決定論的フラクタルアルゴリズムにより作成された画像は，形状の独特な複雑さから計算機による画像を連想させ，主にポスターや広告等のデザインに使われることが多い．

決定論的フラクタルによる画像生成の場合，同じ数式を用いるならば一種類のパラメータにより一種類の画像が決まる．従ってパラメータを変化させながら生成画像のバリエーションを得ることになるが，同じ数式から生成される画像には一定の傾向が見られるため変化に乏しい．さらに生成される画像は自己相似集合[6][7]となるため，全体のミニチュアが各部に繰り返し現れるパターン画像となり形状が単調で面白みに欠ける．そこで反復回数等の条件付けによる色の変更，集合を図示する領域を変える等の処理で画像のバリエーションを得ているのが現状である．

本研究はこうした決定論的フラクタルによる画像生成の問題点に着目し，反復関数系を用いる決定論的フラクタル画像生成方法を拡張，複数のパラメータを反復段階において変更することで多彩なフラクタル画像を生成可能とする技術を開発することを目的とした．

2. 反復関数系によるフラクタル画像生成

2-1 反復関数系

基本となる反復関数系の定義および定理を示す．詳細は文献[8]にある．

定義2.1

(X,d)を完備な距離空間とする．このときハウスドルフ空間$H(X)$とは，空間内の点が空集合を除くコンパクトな部分集合となるような空間を意味する．

定義2.2

反復関数系は，完備な距離空間(X,d)と，$n=1,2,\ldots N$においてそれぞれ縮小係数s_nを持つ縮小写像$\omega_n: X\to X$の有限集合とから成る．'IFS'の略記は'iterated function system(反復関数系)'として使用される．

このIFSを$\{X;\omega_n, n=1,2,\ldots N\}$と表記し，またその縮小係数は$s=Max\{s_n; n=1,2,\ldots N\}$である．

定理2.1

$\{X;\omega_n, n=1,2,\ldots N\}$を縮小係数$s$をもつIFSとすると，全ての$B\in H(X)$において変換$W: H(X)\to H(X)$は

$$W(B)=\bigcup_{n=1}^{N}\omega_n(B)$$

で定義される．これは完備な距離空間$(H(X), h(d))$上での縮小係数sの縮小写像である．この縮小写像の一意の不動点$A\in H(X)$は以下の式を満たす．

$$A=W(A)=\bigcup_{n=1}^{N}\omega_n(A)$$

また任意の$B\in H(X)$について，$A=Lim_{n\to\infty}W^{on}(B)$で与えられる．

2-2 画像生成の例

IFSを利用するフラクタル画像生成方法について説明する．ここでは距離空間(X,d)をユークリッド平面R^2とユークリッド距離とし，R^2上の縮小写像$\omega_i: R^2\to R^2$をアフィン変換

$$\omega_i=\begin{pmatrix}x\\y\end{pmatrix}=\begin{pmatrix}a_i & b_i\\ c_i & d_i\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix}+\begin{pmatrix}e_i\\f_i\end{pmatrix} \qquad (1)$$

とする．

例えばKochの雪片曲線を作成するためのIFSは2つの縮小写像から成り，それぞれ式(1)に対応するパラメータa,b,c,d,e,fはTable1のようになる．

Table1 Parameters of generating the Koch curve

		a	b	c	d	e	f
a. W_1	ω_1	$\frac{1}{2}$	$\frac{\sqrt{3}}{6}$	$\frac{\sqrt{3}}{6}$	$-\frac{1}{2}$	0	0
	ω_2	$\frac{1}{2}$	$-\frac{\sqrt{3}}{6}$	$-\frac{\sqrt{3}}{6}$	$-\frac{1}{2}$	$\frac{1}{2}$	$\frac{\sqrt{3}}{6}$

初期集合K_0を適当に定め，K_0からn回反復後の集合

$$K_n=\omega_1(K_{n-1})\bigcup\omega_2(K_{n-1})=W^{on}(K_0)$$

を算出し，画像として描画する．

K_0を頂点$(0,0),(1,0),(1/2,\sqrt{3}/6)$とする三角形とし，

8回反復後のK_8を求めて画像生成していく様子をFig.1に示す.

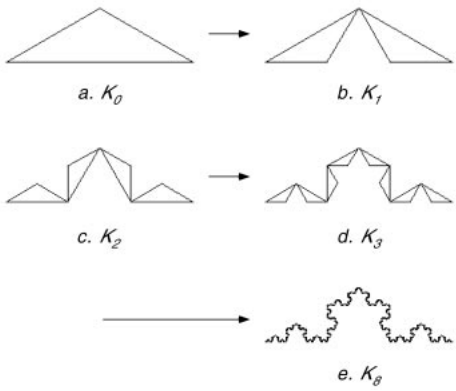

Fig.1 Example of generating the Koch curve under IFS(Iterated Function System)

算出された集合K_nを画像にする際，反復回数や軌跡による色の変更等の処理を行なうことでバリエーションが得られる．また反復回数nを十分大きくとるとき不動点は出力デバイス上で1ピクセルに収束するため初期集合K_0は任意であるが,生成する形状の興味という点からは少ない回数で反復を打ち切ることも可能であり，このときFig.1のK_1からK_3に見られるように細部にK_0の形状が現れる．文献[4]ではK_0を工夫することにより画像のバリエーションを増やしている.

IFS$\{\omega_1, \omega_2\}$のパラメータを変化させることで様々な形状を生成することが可能である．パラメータを変化させたときに同様の方法で生成した画像の例をFig.2 bからfに示す．各画像を生成するためのパラメータはAPPENDIX A-1に示した.

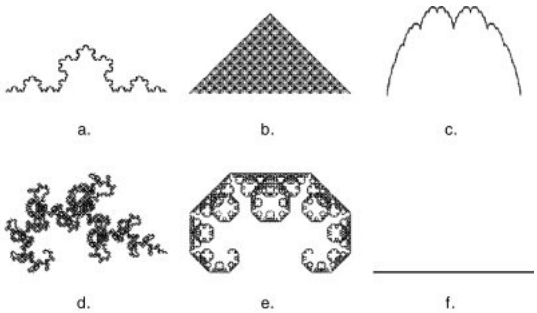

Fig.2 Examples of image generated by IFS

3．反復関数集合によるフラクタル画像生成

3-1 反復関数集合

反復関数系を拡張する．拡張の目的はより多彩なフラクタル画像を得ることにある．反復法における変換Wは反復過程において終始固定であるため，最終的に得られる集合は常に一定であった．そこで反復の各段階において変換Wを変更可能となるよう反復関数集合を定義する.

定義3.1(反復関数集合)

反復関数集合は，$m=1,2,\ldots,M$についてそれぞれ縮小係数s_mをもつ縮小写像$\omega_m: X \to X$からなる有限集合を$W=\{\omega_m, m=1,2,\ldots,M\}$としたとき，$N$個の$W$からなる有限集合$\{W_n, n=1,2,\ldots,N\}$を意味する.

反復の各段階で反復関数集合から変換を選択して適用することでパラメータを変化させることが可能となる.

定義3.2 (確率付き反復関数集合)

確率付き反復関数集合は，反復関数集合$\{W_n, n=1,2,\ldots,N\}$と，$n=1,2,\ldots,N$において

$$p_1+p_2+p_3+,\cdots,+p_n=1, p_n \geq 0$$

を満たすN個の有限集合$\{p_n, n=1,2,\ldots,N\}$とから成る．確率付き反復関数集合を$\{W_n; p_n, n=1,2,\ldots,N\}$と表記する.

p_nは，ある反復過程で反復関数集合のn番目の要素が変換として選択される確率を示す.

3-2 画像生成の例

Fig.2のa, bに示したKochの雪片曲線とPolya曲線の2つの変換からなる反復関数集合
$\{W_1, W_2\}=\{\{\omega_{11}, \omega_{12}\}, \{\omega_{21}, \omega_{22}\}\}$を例に，反復関数集合による画像生成方法について説明する.

まず各反復過程での変換W_1, W_2の適用順序を定める．ここでは反復回数nが奇数のときW_1，偶数のときW_2を適用するとすると，変換を4回反復した後の集合K_4は

$$K_4=W_2(W_1(W_2(W_1(K_0))))=W_2 o W_1 o W_2 o W_1(K_0)$$

となる．反復過程において形状が変化していく様子をFig.3に示す．変換の適用順序が異なると必ず形状が異なる画像を生成できるため，少ないパラメータから多彩なフラクタル画像を得ることが可能である.

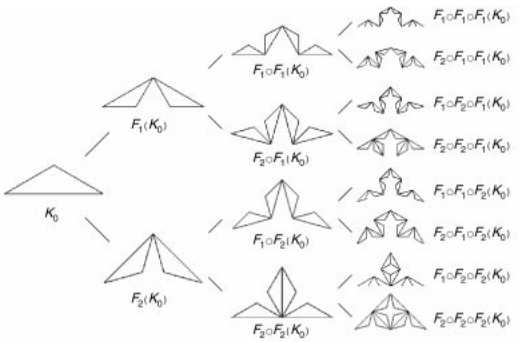

Fig.3 Example of generating images by IFS with a function set

3-3 反復関数集合によるフラクタル形状制御

反復関数集合を利用することでフラクタル形状の制御を行なうことが可能となる．一般に，フラクタル画像を生成する際に未知のパラメータから算出される不動点をあらかじめ予測するのは非常に困難であり，実際に計算機を使用して画像生成した後に形状を確認しているのが現状である．そこであらかじめ不動点のわかっている既知のパラメータのみを用いて反復関数集合を作成し，それを利用して画像生成を行なうことである程度最終的な形状を制御できる．

またコラージュ定理[8]によれば，パラメータの値に小さな変更を与えたときには不動点すなわち形状も小さな変更を受けることがいえる．従ってパラメータに小さな変更を加えて新しいパラメータを作成し，それらを要素とする反復関数集合を使用して画像生成を行なうことにより元の形状を基本的に崩さずに細かな変更を与えることができる．Fig.4では基本となる変換W_1のパラメータの値を少しずつ変更してW_2からW_5を作成し，それらからなる反復関数集合を使用して画像生成を行なうことで最終形状にわずかな変更を与えてみた．bからeのどの画像も基本的な形状は元のaと大差ないが，細部はいずれも異なっている．

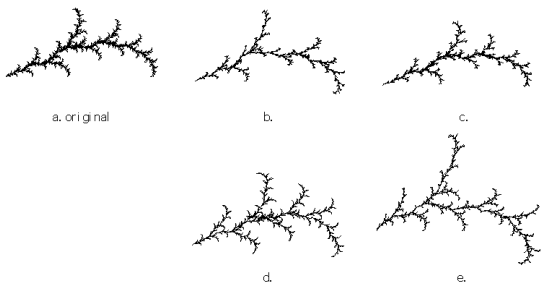

Fig.4 Controling fractal images by IFS with a function set

確率付き反復関数集合を利用することで選択される変換に重み付けを行ない，さらに細かい制御を行なうことも可能である．

3-4 3Dモデリング

距離空間を(R^3, d)とし，R^2上の写像式(1)をR^3上の写像式(2)にすることで，2次元の場合と同じアルゴリズムで反復関数集合による3次元フラクタルモデルを生成することが可能である．

$$\omega_i\begin{pmatrix}x\\y\\z\end{pmatrix}=\begin{pmatrix}a_{11}&a_{12}&a_{13}\\a_{21}&a_{22}&a_{23}\\a_{31}&a_{32}&a_{33}\end{pmatrix}\begin{pmatrix}x\\y\\z\end{pmatrix}+\begin{pmatrix}b_{11}\\b_{21}\\b_{31}\end{pmatrix} \qquad (2)$$

例として3Dシェルピンスキーガスケットと3D Levyからなる反復関数集合を使用してモデルを生成し，適当なレンダリングを施して表示させた例をFig.5に示す．初期集合K_0はR^3上の5点(0,0,0)，(1,0,0)，(1/2, 0, √3/6)，(1/2, 0,－√3/6)，(1/2, √3/6, 0)を頂点とする四角錐とした．

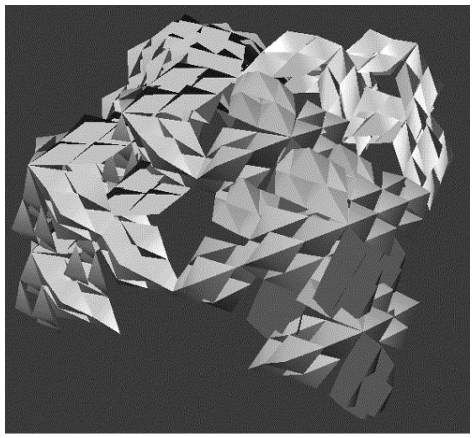

Fig.5 Example of 3D Fractal Model generated by IFS with a function set

4．応用

4-1 パターン画像の作成

反復関数集合によるフラクタル画像の一つのアプリケーションとしてパターン画像作成ツールを開発した．パターン画像とは タイル と呼ばれる基本画像を平面上に幾何的に複数配置することで得られる繰り返しを持つ画像のことを指す．

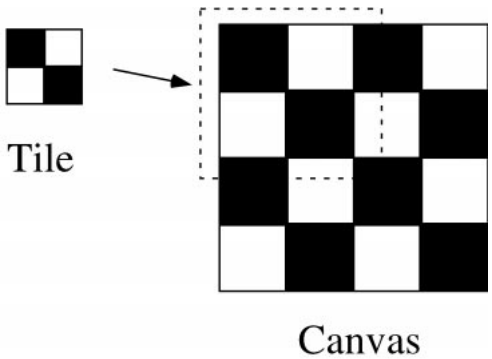

Fig.6 Generating a pattern by a tile image

まず基本となる画像を反復関数集合を用いるフラクタルアルゴリズムで自動生成し，生成された画像をそのまま，あるいは全体を左右/上下対称となるようにアフィン変換を施したものとの重ね合わせを行うことで タイル を作成した．

Fig.2で示された6種類の反復関数系を要素として持つ反復関数集合から作成可能なタイルの例をFig.7に示す．各タイルを作成するためのパラメータをAPPENDIX A-4に示す．

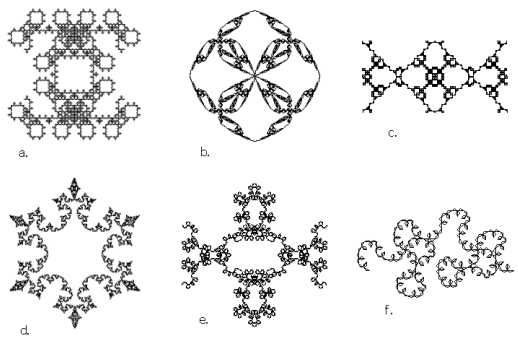

Fig.7 Examples of tiles generated by IFS with a function set

このような操作で得られたタイルをツールを用いてビットマップ上に幾何的に繰り返し配置し，最終的に葉書作成アプリケーションで使用する背景用テンプレート用画像（文様または地紋）とした．反復関数集合により作成された文様，地紋の例をFig.8 a, bに示す．

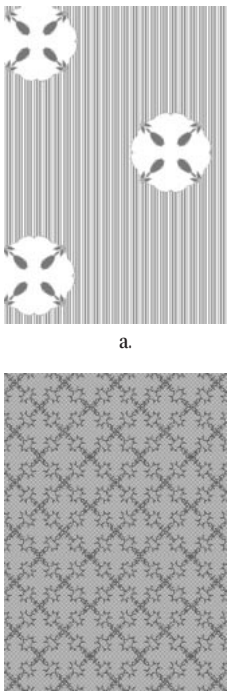

Fig.8 Examples of patterns by fractal images

5．結論

反復関数集合によるフラクタル画像生成は従来の方法と比較して以下の点で有効である．

・バリエーションの飛躍的な増加

異なる変換を異なる順序で適用することにより生成可能な形状を飛躍的に増加させることが可能である．反復関数集合の要素数m，反復回数nのときに得られる画像のバリエーションの総数はm^n種類であり，例えばFig.2に示した6種類のパラメータを利用して反復回数10回の画像を生成する場合に生成し得るバリエーションの総数は$6^{10}=60,466,176$種類となる．

・形状制御

既知のパラメータを組み合わせて使用する，あるいは一つのパラメータの値から新しいパラメータを作成し，それらを組み合わせることで形状を制御することができる．パラメータの値をランダムに変更する方法と比較して 最終形状に連結性を保持したい 等の細かな形状制御をする必要がある場合に，特に有効である．さらに適用する変換の選択される確率を設定することにより，より細かな制御を行なうことも可能である．

・データサイズの圧縮

多数のフラクタル画像をデータ化する際に，反復関数集合の要素を複数の形状間で共用することによりデータサイズを少なく抑えることが可能である．反復関数集合の要素数mを，反復回数nをとすると，色などの付加データを除く形状のみのデータは変換の適用順序だけで済むため，データサイズの理論値は1つの形状につき$n\times\log_2 m$ビットとなる．

6．今後の展開

今後の展開として3Dモデリングへの応用を進めていく予定である．本文中で述べたフラクタルモデリングに加えて景観シミュレーションやテクスチャ画像生成などへの応用，また植物形状を表現するために広く用いられているL-systemにおける反復段階でのパラメータ変更方法の検討も行なっていく．

参考文献

1) Mandelbrot,B.B.: *Fractal Geometry of Nature*, W.H.Freeman and Co., New York, (1982)
2) Mandelbrot,B.B. and Ness, J.W.van: Fractional Brownian Motion, fractional noises and applications, SIAM Review 10,4 (1968), p.422-437
3) M.Barnsley and S. Demko: *Iterated Function Systems and the Global Construction of Fractals*, Proceedings of the Royal Society of London, A399 (1985), p243-275
4) Day, N. and Murayama, N.: *Fractal Image Generation*, IP-94-33, IEEJ Chaos Workshop, (1994)
5) Przemyslaw Prusinkiewicz: *Graphical Applications of L-system*, Graphics Interface, (1986), p247-253
6) Hata, M.: *On the structure of self-similar sets*, Japan Journal of Applied Mathematics 2,2 (1985), p.381-414
7) Hutchinson,J.: *Fractals and self-similarity*, Indiana University Journal of Mathematics 30 (1981), p.713-747
8) M.Barnsley: *Fractals Everywhere*, Academic Press, Inc. (1988), p96-97

APPENDIX

A. Parameters of generating fractals

A-1 Fig.2

Table 2. Mappings

		a	b	c	d	e	f
b. W_2	ω_1	1/2	1/2	1/2	−1/2	0	0
	ω_2	1/2	−1/2	−1/2	−1/2	1/2	1/2
c. W_3	ω_1	1/2	0	1/2	1/2	0	0
	ω_2	1/2	0	−1/2	1/2	1/2	1/2
d. W_4	ω_1	1/2	−1/2	1/2	1/2	0	0
	ω_2	1/2	−1/2	−1/2	−1/2	1/2	1/2
e. W_5	ω_1	1/2	−1/2	1/2	1/2	0	0
	ω_2	1/2	1/2	−1/2	1/2	1/2	1/2
f. W_6	ω_1	1/2	0	0	−1/2	0	0
	ω_2	1/2	0	0	−1/2	1/2	0

A-2 Fig.4

Table 3a. Mappings

		a	b	c	d	e	f
W_1	ω_1	.461	.461	.461	-.461	.000	.000
	ω_2	.622	.196	-.196	.622	.378	.196
W_2	ω_1	.511	.375	.431	-.375	.000	.000
	ω_2	.622	.196	-.196	.622	.378	.196
W_3	ω_1	.511	.375	.431	-.375	.000	.000
	ω_2	.622	.196	-.196	500	.378	.196
W_4	ω_1	.511	.375	.431	-.375	.000	.000
	ω_2	.622	.196	-.250	.500	.378	.196
W_5	ω_1	.500	.250	.450	-.500	.000	.000
	ω_2	.625	.375	-.250	-.500	.378	.196
W_6	ω_1	.500	.289	.289	-.500	.000	.000
	ω_2	.625	.125	-.250	.625	.375	.125

Table 3b. Orders

a. (original)	$W_1^{\infty}(K_0)$
b.	$W_2 \circ W_4 \circ W_5 \circ W_2 \circ W_4 \circ W_5 \circ W_2 \circ W_4 \circ W_5(K_0)$
c.	$W_2 \circ W_6 \circ W_5 \circ W_5 \circ W_3 \circ W_4 \circ W_2 \circ W_1 \circ W_3(K_0)$
d.	$W_6 \circ W_6 \circ W_4 \circ W_2 \circ W_6 \circ W_6 \circ W_1 \circ W_1 \circ W_5(K_0)$
e.	$W_5 \circ W_6 \circ W_5 \circ W_6 \circ W_2 \circ W_6 \circ W_5 \circ W_6 \circ W_5(K_0)$

A-3 Fig.5

Table 4a. 3D Levy

W_1	ω_1	a_{11}	a_{12}	a_{13}	b_{11}
3D Levy		.500	-.500	.000	.000
		a_{21}	a_{22}	a_{23}	b_{21}
		.500	.500	.000	.000
		a_{31}	a_{32}	a_{33}	b_{31}
		.000	.000	.707	.000
	ω_2	a_{11}	a_{12}	a_{13}	b_{11}
		-.500	.500	.000	1.00
		a_{21}	a_{22}	a_{23}	b_{21}
		.500	.500	.000	.000
		a_{31}	a_{32}	a_{33}	b_{31}
		.000	.000	.707	.000
	ω_3	a_{11}	a_{12}	a_{13}	b_{11}
		.000	.000	-.707	.500
		a_{21}	a_{22}	a_{23}	b_{21}
		.500	.500	.000	.000
		a_{31}	a_{32}	a_{33}	b_{31}
		.500	-.500	.000	-.500
	ω_4	a_{11}	a_{12}	a_{13}	b_{11}
		.000	.000	-.707	.500
		a_{21}	a_{22}	a_{23}	b_{21}
		-.500	.500	.000	.500
		a_{31}	a_{32}	a_{33}	b_{31}
		.500	.500	.000	.000

Table 4b. 3D Sierpinski gasket

W_2	ω_1	a_{11}	a_{12}	a_{13}	b_{11}
3D Sierpinski Gasket		.500	.000	.000	.250
		a_{21}	a_{22}	a_{23}	b_{21}
		.000	.500	.000	.144
		a_{31}	a_{32}	a_{33}	b_{31}
		.000	.000	.500	.000
	ω_2	a_{11}	a_{12}	a_{13}	b_{11}
		.500	.000	.000	.000
		a_{21}	a_{22}	a_{23}	b_{21}
		.000	.500	.000	.000
		a_{31}	a_{32}	a_{33}	b_{31}
		.000	.000	.500	.000
	ω_3	a_{11}	a_{12}	a_{13}	b_{11}
		.500	.000	.000	.500
		a_{21}	a_{22}	a_{23}	b_{21}
		.000	.500	.000	.000
		a_{31}	a_{32}	a_{33}	b_{31}
		.000	.000	.500	.000
	ω_4	a_{11}	a_{12}	a_{13}	b_{11}
		.500	.000	.000	.250
		a_{21}	a_{22}	a_{23}	b_{21}
		.000	.500	.000	.000
		a_{31}	a_{32}	a_{33}	b_{31}
		.000	.000	.500	.144
	ω_5	a_{11}	a_{12}	a_{13}	b_{11}
		.500	.000	.000	.250
		a_{21}	a_{22}	a_{23}	b_{21}
		.000	.500	.000	.000
		a_{31}	a_{32}	a_{33}	b_{31}
		.000	.000	.500	-.144

b. Order

A-4 Fig.7

See the Table 1 and Table 2 for the mapping data

Table 5. Orders

a.	$W_2 \circ W_5 \circ W_5 \circ W_5 \circ W_2 \circ W_5 \circ W_5 \circ W_5 \circ W_2(K_0)$
b.	$W_2 \circ W_3 \circ W_2 \circ W_2 \circ W_3 \circ W_2 \circ W_2 \circ W_3 \circ W_2 \circ W_2(K_0)$
c.	$W_2 \circ W_5 \circ W_4 \circ W_2 \circ W_5 \circ W_4 \circ W_2 \circ W_6 \circ W_4(K_0)$
d.	$W_2 \circ W_1 \circ W_2 \circ W_1 \circ W_2 \circ W_1 \circ W_2 \circ W_1(K_0)$
e.	$W_5 \circ W_5 \circ W_1 \circ W_5 \circ W_5 \circ W_1 \circ W_5 \circ W_5 \circ W_1(K_0)$
f.	$W_5 \circ W_2 \circ W_1 \circ W_5 \circ W_4 \circ W_5 \circ W_2 \circ W_1(K_0)$