The Effect of Multiple Rotations on Mobius Transformations in Generating IFS Fractals

T. Gangopadhyay
XLRI
C.H. Area (E), Jamshedpur,
India

ABSTRACT

Iterating two pairs of Mobius transformations as attractors generates fractals that are known as Mobius fractals. In the present paper one studies the effect of multiple rotations on Mobius transformations, that frequently lead to more well-rounded as well as original fractal designs.

General Terms

Fractal, Algorithm, Turbo C++, Program.

Keywords

Mobius, IFS, rotations,.

1. INTRODUCTION

A Mobius transformation of a complex number z is of the form $(az+b)/(cz+d)$, where $ad - bc \neq 0$ (Krantz [4],,Needham [7]). Iterating two pairs of Mobius transformations as attractors generates fractals that are known as Mobius fractals or Mobius maps(Mumford[6], Stevens[8]). In earlier papers one has studied the effect of related trigonometric coefficients on affine transformations in terms of the IFS fractals generated by them (Gangopadhyay[2]) as well as the effect of multiple rotations on these affine transformations (Gangopadhyay[3]). In the present paper one studies the effect of multiple rotations on Mobius transformations. Such rotations lend additional depth to the generated fractal as well as create new fractal designs. In a sense, the method used is a generalization of the process used for Pharaoh's breastplate (Mandelbrot and Novak [5]) and other Mobius fractals First a set of Mobius transformations are produced. Next multiple rotations are applied to produce more well-rounded and new fractal designs. These are primarily the distinctive features of this paper.

2. THE ALGORITHM

The algorithm takes two generic Mobius transformations, the first with coefficients a1, b1,c1 and d1 and the second with a2, b2, c2 and d2. Each is then rotated multiple times through angles that are multiples of a given angle. This given angle for a transformation has the value 360/m in degrees, where m is the number of rotations for that transformation.

In the next section we submit a programming code in Turbo C++ that captures the algorithm and generate some sample output.

3. THE CODE

The code uses a function tgmob which is declared first. The function has eleven parameters – the two sets of complex coefficients a1, b1, c1, d1 and a2, b2, c2, d2, an integer n which gives the total number of rotations and an integer m which gives the number of rotations per transformation. Both the tgmob function and the code are given below:

```
void tgmob(cmplx a1,cmplx b1,cmplx c1,
cmplx d1,cmplx a2,cmplx b2,cmplx c2,
cmplx d2, int n,int m)
{int j,k; double x=1,y=1,px; int scale=250;
cmplx z=cmplx(.1,.5); float ang;
for(long i=0;i<42700000;i++)
{  px=x;
j=random(999)%1000;
k=j%n;
if(k<m){ang=(k*360/m)*3.14/180.;
z=(a1*z+b1)/(c1*z+d1);
x=real(z),y=imag(z);px=x;
x=x*cos(ang)-y*sin(ang);
y=px*sin(ang)+y*cos(ang);
z=cmplx(x,y); }
else{ang=((k-m)*360/(n-m))*3.14/180.;
z=(a2*z+b2)/(c2*z+d2);
x=real(z),y=imag(z);px=x;
x=x*cos(ang)-y*sin(ang);
y=px*sin(ang)+y*cos(ang);
z=cmplx(x,y); }
x=real(z),y=imag(z);
int x1=scale*x+512,y1=scale*y+400;
if(x1>0&&x1<1024&&y1>0&&y1<800)
putpixel(x1,y1,getpixel(x1,y1)+1);
}}
void main()
{
cmplx ii=cmplx(0,1);
tgmob(-ii,0,2,ii,1,.5,0,1,4,2);getch();
closegraph();
}
```

The output of the   sample code is illustrated in Figure 1, a standard mobius fractal(see Mumford[6]).
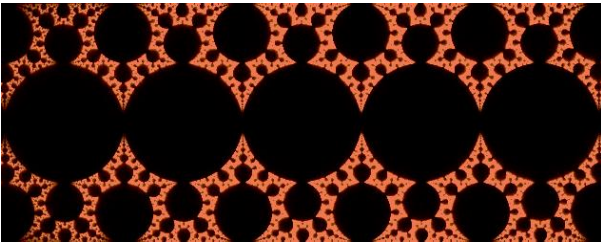
Fig 1 : Output of the sample code

By changing the value of b2 to 1, scale to 450 and n to 6 and other parameters unchanged in the function tgmob we can generate a new fractal design. Here the first transformation is rotated twice while the second is rotated four times. The result is shown in Figure 2.
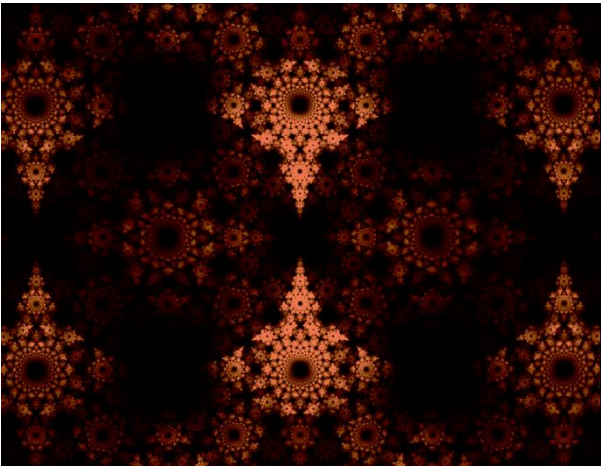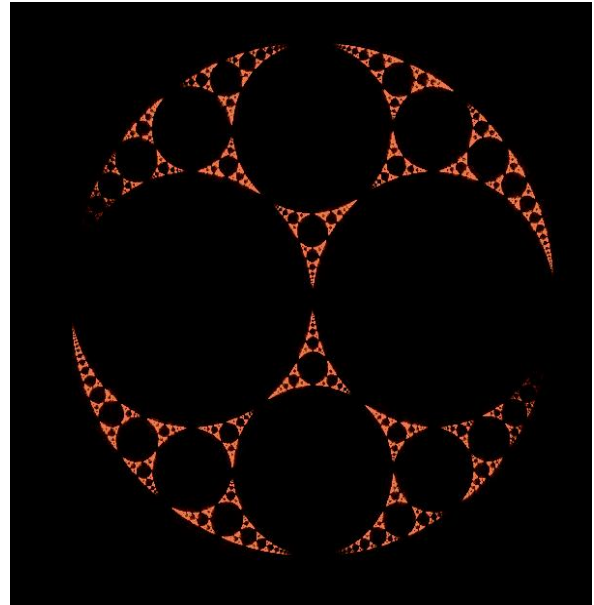

Fig. 2 : Output of sample code with b2=1 and n=6.

If, in addition m is also changed to 4, the first transformation is rotated four times and the second only twice. The result is shown in Figure 3 .
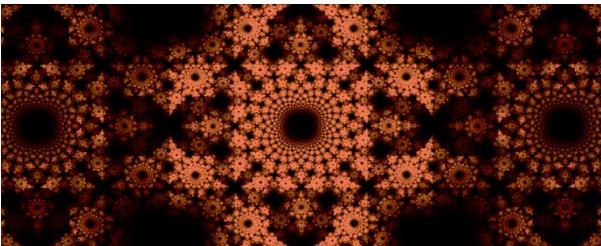

Fig. 3 : Output of sample code with b2=1, n=6 and m=4.

4. OTHER TRANSFORMATIONS AND THEIR ROTATIONS

Now, the second transformation is changed completely by setting a2=1-ii,b2=1,c2=1 and d2=1+ii. For n=4 and m=2, the output is an apollonian gasket(Bourke[1]) shown in Figure 4.
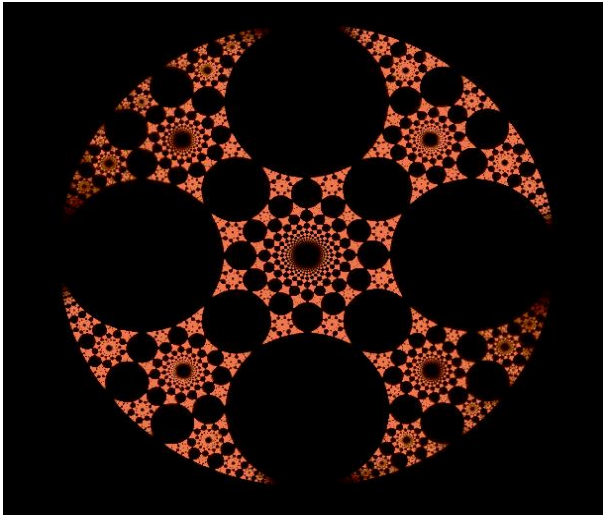

Fig. 4 : Output of sample code with the second transformation changed.

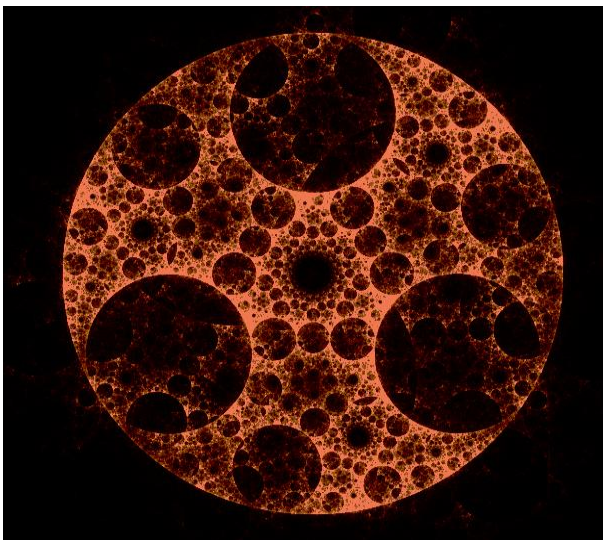Changing n to 6 in Figure 4 results in a more well-rounded and symmetric output which is depicted Figure 5.
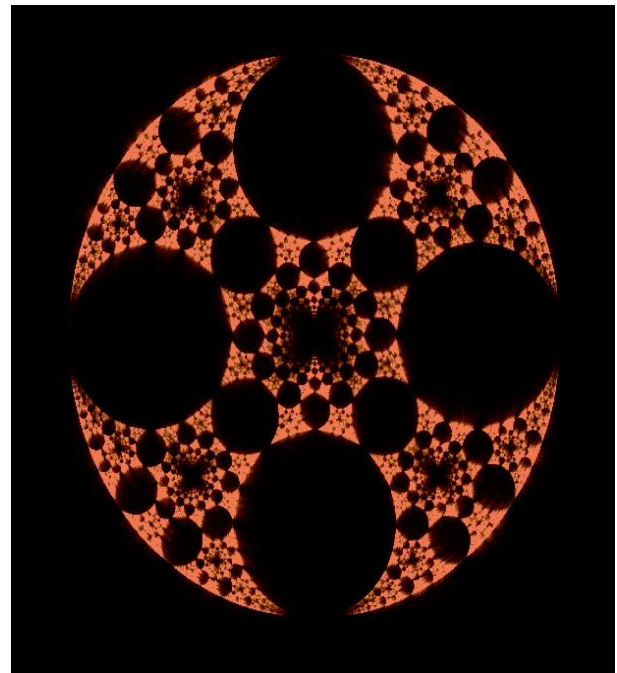
Fig. 5 : The effect of changing n in Figure 4

If, in addition to changing n, m is also changed to 4, one gest an interesting variant of Figure 5, which is depicted in Figure 6.

Fig. 6 : The effect of changing m in Figure 5
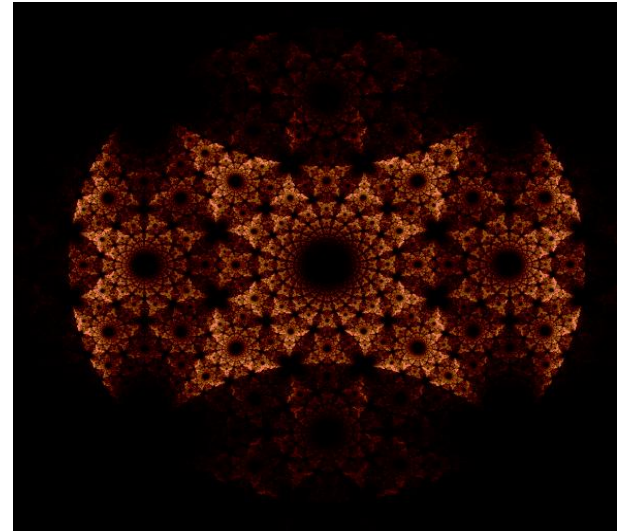
Finally, by changing m to 3 one obtains the following:
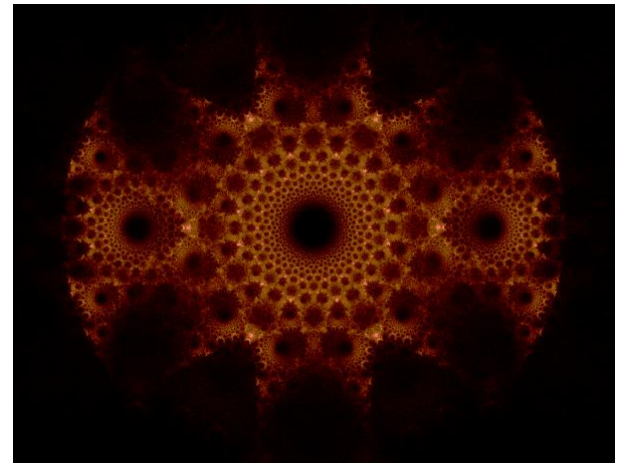

Fig. 7 : The effect of m = 3 in Figure 6

Next, the first transformation is also changed, by setting c1 to 1+ii. Then for n=4 and m=2, n=6 and m=4 and n=8, m=6 one obtains the output in Figures 8,9 and 10 respectively. Figures 9 and 10 show the effect of multiple rotations on the second transformation.
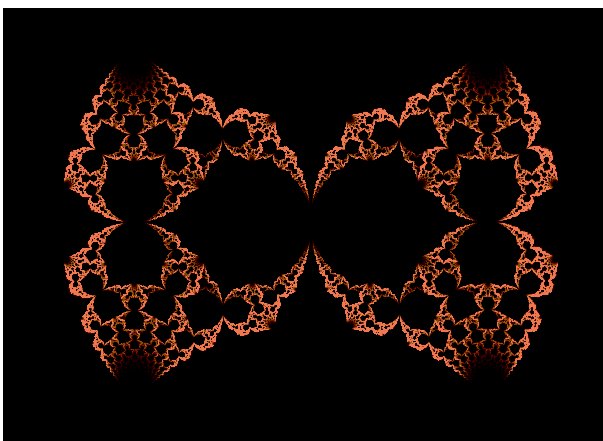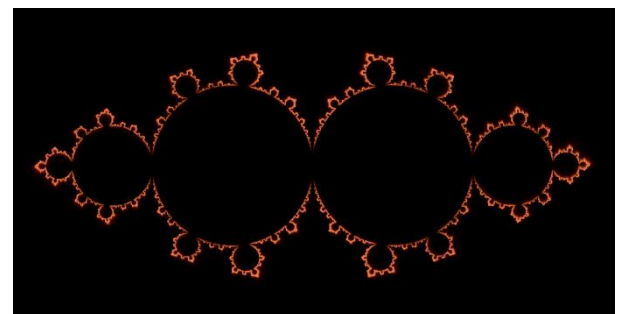

Fig. 8 : c1=1+ii, n=4 , m=2

Fig. 9 : c1=1+ii, n=6 , m=4

Fig. 10 : c1=1+ii, n=8 , m=6

5. USING IRRATIONAL COEFFICIENTS IN TRANSFORMATIONS

Next keeping the first transformation as in the sample code and changing the second by setting a2=sqrt(1.5), b2=sqrt(.5), c2=sqrt(.5) and d2=sqrt(1.5), one obtains the output resembling a double mandelbrot. The same is depicted in Figure 11.

Fig. 11 : Double Mandelbrot(n=4, m=2)

Figures 12 and 13 respectively depict the effect of multiple rotations on Figure 11 for n=6, m=2 and n=6,m=4. In Figure 13, c1 is also changed to 3.
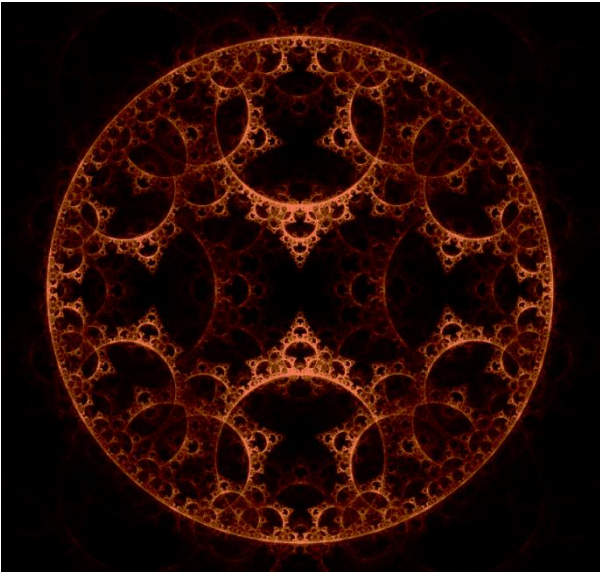
Fig. 12 : Double Mandelbrot(n=6, m=2)

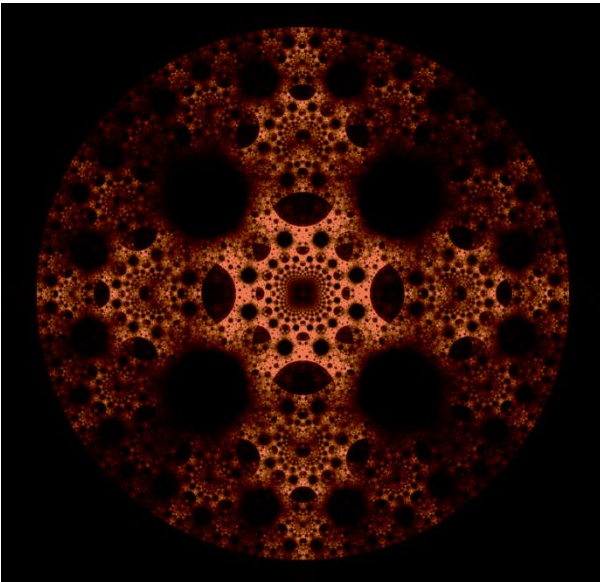**Fig. 13 : Double Mandelbrot(n=6, m=4, c1=3)**

Finally, if one multiplies a2,b2,c2 by ii and d2 by –ii, then the one obtains the output shown in Figure 14.:
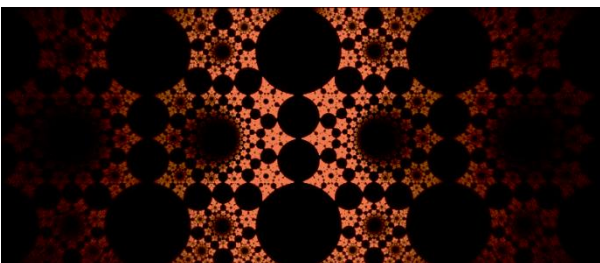

Fig.14. Using imaginary coefficient in the second transformation

Again, if one sets c1 in the first transformation to 3 and changes the second transformation by setting a2=sqrt(5), b2=2 c2=-2 and d2=sqrt(5), the output produced is the twin Pharaoh's breastplate shown in Figure 15.
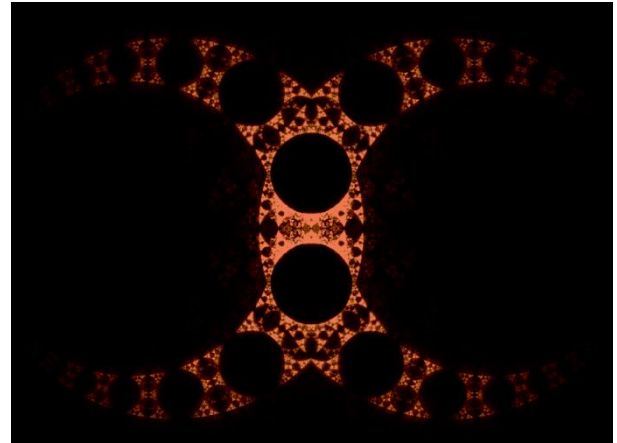

Fig. 15 : Twin Pharaoh's breastplate

6. ADDING A THIRD TRANSFORMAT-ION

If one uses an additional transformation further interesting output can be retrieved. An example is shown in Figure 16. Here the coefficients for the three transformations are a1,b1,c1,d1 for the first, a2,b2,c2,d2 for the second and a1,b1,-c1 and d1 for the third, where a1=1,b1=0,c1=-2.5*ii,d1=1 and a2=.5, b2=.25,c2=.25,d2=.5;
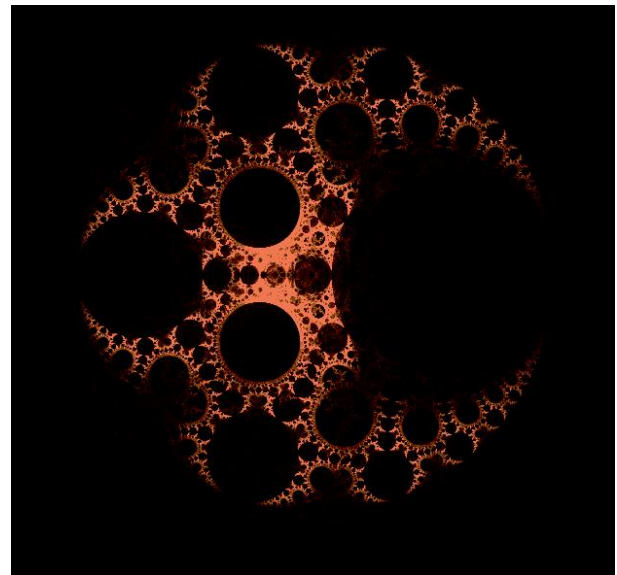

Fig. 16 : Adding a transformation

7. CONCLUSION

This paper presents the effect of applying rotations of Mobius transformations in generating IFS fractals. In subsequent studies one will further explore the effect having three or more Mobius transformations. One will also study the effect of multiple rotations on single Mobius transformation. These are the aspects that would be explored in future work.

8. ACKNOWLEDGMENTS

The author wishes to acknowledge his debt to the referee(s) for their constructive suggestions and encouragement

9. REFERENCES

[1] Bourke, P.: An Introduction to the Apollony Fractal. Computers and Graphics, 30(2006) 134–136.

[2] Gangopadhyay, T. IFS Fractals generated by Affine Transformation with Trigonometric Coefficients and their Transformations, International journal of Computer Applications 53(2012) 29-32..

[3] Gangopadhyay, T. The Effect of Multiple Rotations on a Unified System of Affine Transformations with Related Trigonometric Coefficients, International journal of Computer Applications April(2016)(in press)..

[4] Krantz, S. G. "Möbius Transformations." §6.2.2 in Handbook of Complex Variables. Boston, MA: Birkhäuser, p. 81, 1999..

[5] Benoit B. Mandelbrot, Miroslav Michal Novak, Thinking in Patterns: Fractals and Related Phenomena in Nature

[6] Mumford , D et al. Indra's Pearls, Cambridge University Press, 2002

[7] Needham, T. "Möbius Transformations and Inversion." Ch. 3 in Visual Complex Analysis. New York: Clarendon Press, pp. 122-188, 2000.html.

[8] Stevens, R. Creating Fractals, Charles River media, Inc. 2005