On Transforming Popcorn Fractals with Spherical and Other Functions

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

ABSTRACT

Popcorn fractals are instances of Integrated Fractal Systems involving trigonometric functions. In this paper, we study the effect of spherical, swirl and pseudo-horseshoe functions on popcorn fractals to produce talismanic and tantric designs.

General Terms

Fractal, Algorithm, Turbo C++, Program.

Keywords

Popcorn, IFS, spherical, swirl, bailout.

1. INTRODUCTION

In an earlier paper (Gangopadhyay[5]) we have studied the effect of the average of two special transformations on standard escape-time fractals. In this paper we study their effect on popcorn fractals. Popcorn fractals were introduced first by Clifford Pickover ([7]). These are a type of IFS generated fractals. Usually in IFS generated fractals an arbitrary point is transformed repeatedly through multiple affine functions to produce fractal shapes. In popcorn fractals, however, there are two distinguishing features. Instead of affine functions trigonometric functions are used. Also, every kth pixel(k>=2) is iterated a finite number of times. The coloring scheme uses the number of hits. This paper's distinctive features consist of applying a second (sometimes a third) transformation before coloring the pixel. The transformations are of the type introduced by Scott Draves [4] in his flame fractal. The final image generated resembles sometimes a talisman, sometimes Tantric art. The effect is quite distinct from that of Pickover popcorn fractals[ ], and also from usual IFS fractals such as fern (Barnsley [1]), maple leaf (Barnsley [1]), dragon curves (Davis and Knuth [3]), Lissajous figures (Brill [2]) or those obtained through strange attractors (Hofstadter [6], Ruelle [8]).

2. THE ALGORITHM

In iterated function systems Michael Barnsley [1] used affine transformations repeatedly on a starting point to produce fractal shapes. Draves [4] extended the scope of these transformations further. He introduced seven functional variations of the original point. These are described below:

Let (x, y) be the coordinates of the original point. Let $r = \sqrt{x^2 + y^2}$, $s = r^2$ and $t = \arctan(y/x)$.

a) linear : $f(x, y) = (x, y)$.

b) sinusoidal : $f(x, y) = (\sin x, \sin y)$.

c) spherical : $f(x, y) = (x/s, y/s)$.

d) horseshoe : $f(x, y) = (r\cos(t + r), r\sin(t + r))$.

e) swirl : $f(x, y) = (r\cos(2t), r\sin(2t))$.

f) polar : $f(x, y) = (t/\pi, r - 1)$

g) bent : $f(x, y) = (g(x), h(y))$

where

$$g(x) = x \quad \text{if} \quad x \geq 0$$
$$= c * x \text{ otherwise}$$

and

$$h(y) = y \quad \text{if} \quad y \geq 0$$
$$= y/c \text{ otherwise.}$$

In the present paper, we take standard Pickover popcorn fractals as our starting point. For each complex variable z that is iterated, we first separate real(z) and imag(z). We apply Pickover transformations two these parts separately. The standard transformations use sin and tan functions. We have varied these for some of the more startling effects. We next apply some of Draves transformations. In the code given below, we use the spherical transformation in the second stage. In our notation:

z= (real(z), imag(z)) first by

z=(x-h*real(g1(y+g2(a*y)))-h*imag(g1(x+g2(a*x))),

y-h*real(g1(px+g2(a*px)))-h*imag(g1(y+g2(a*y)))) ,

and then by

z=(x/norm(z),y/norm(z))

which is the spherical transformation. Finally we rotate the image by 45 degree.

Using standard forms of bailout we colour each escaping pixel by its current hit number..

In the next section we submit a programme in Turbo C++.

3. THE CODE

We use the variable names standardized in Stevens [9].

```
void fractal(double,double,double,double,int,int);


void main()

{

double xmax=1.7,xmin=-1.7,ymax=1.7,ymin=-
1.7,deltap,deltaq;

int max_iterations=21;int max_size= 1124.0;

fractal(xmax,xmin,ymax,ymin,max_iterations,max_size);

getch();

closegraph();

}


complex g1(complex x)

{return (sin(x));}

complex g2(complex x)

{return (tan(x));}

double cabs(complex z)

{return sqrt(norm(z));}

complex flip(complex c)

{return complex(imag(c),real(c)); }


void fractal(double xmax,double xmin,double ymax,double
ymin,int max_iterations,int max_size)

{complex c,z;    complex x,y,px,a=complex(3,0);

int color;

float col=0,row;float dist;

double deltap,deltaq;float h=.05;

deltap=(xmax-xmin)/480;

deltaq=(ymax-ymin)/480;

while(!kbhit()&&col<480){ col+=1;

{for(row=0;row<480;row+=1)

{z=c=complex(xmin+col*deltap,ymax-row*deltaq);

color=0;

while((color<max_iterations)&&(norm(z)<max_size))

{x=real(z),y=imag(z); px=x;

x=x-h*real(g1(y+g2(a*y)))-h*imag(g1(x+g2(a*x)));

y=y-h*real(g1(px+g2(a*px)))-h*imag(g1(y+g2(a*y)));

z=(x)+flip((y));

x=x/complex(norm((z)),0.0),y=y/complex(norm((z)),0.0);

z=(x)+flip((y));

color++;

float angle=45.*3.14/180.;

int x0=(real(z)*cos(angle)-imag(z)*sin(angle)-xmin)/deltap,

y0=(ymax-real(z)*sin(angle)-imag(z)*cos(angle))/deltaq;

if(getpixel(x0+50,y0)<224)putpixel(x0+50,y0,getpixel(x0+50
,y0)+1);

}}}}}
```

The output of this sample code is illustrated in Figure 1a.

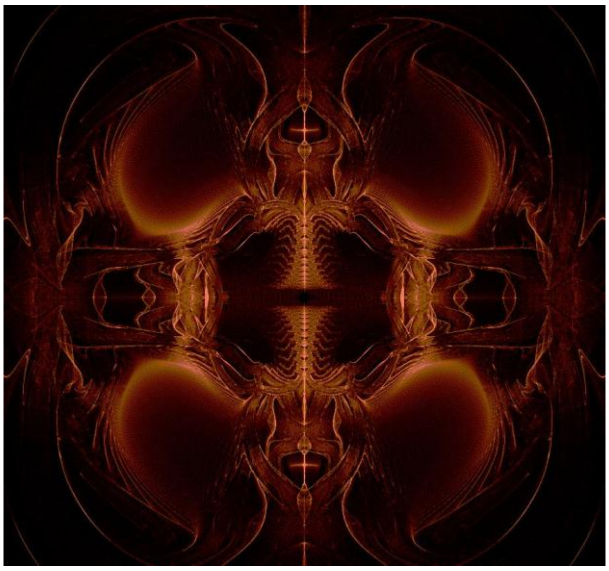

Fig 1a : Output of the sample code

The standard popcorn fractal for the same parameters is diplayed belowin Figure 1b for comparison

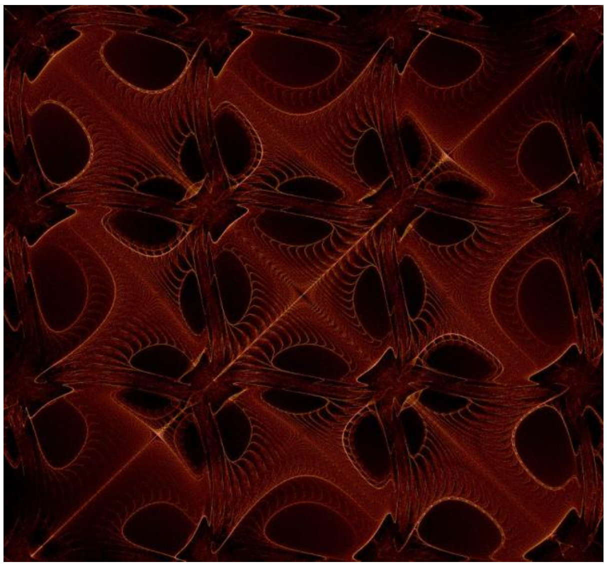

Fig 1b : Standard popcorn fractal

4. VARIATIONS ON THE SAME THEME

In the sample code we use two function: g1(x) = sin(x) and g2(x) = cos((x)+tan(x)). By changing these we generate a variety of images.

VARIATION 1

Let g1(x) = sin(x)+cos(x) and g2(x) = x+tan(x).

The output is illustrated in Figure 2.

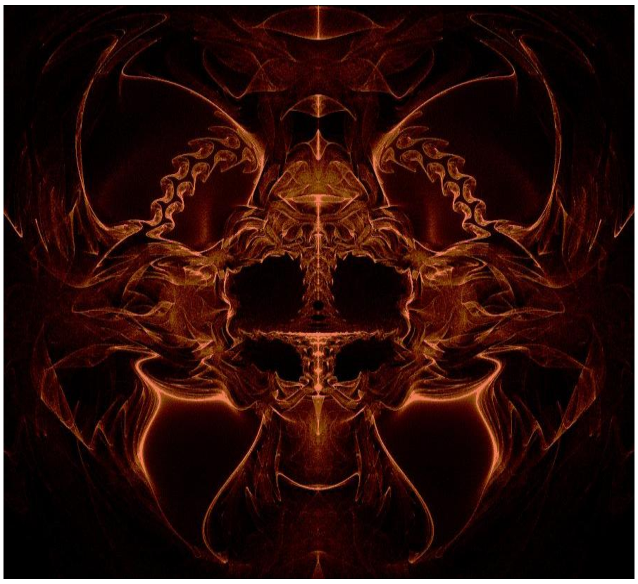

Fig. 2 : Output of Variation 1

VARIATION 2

Let g1(x) = sinh(x+sin(x)) and g2(x) = cos((x)+tan(x)).

The output is illustrated in Figure 3.

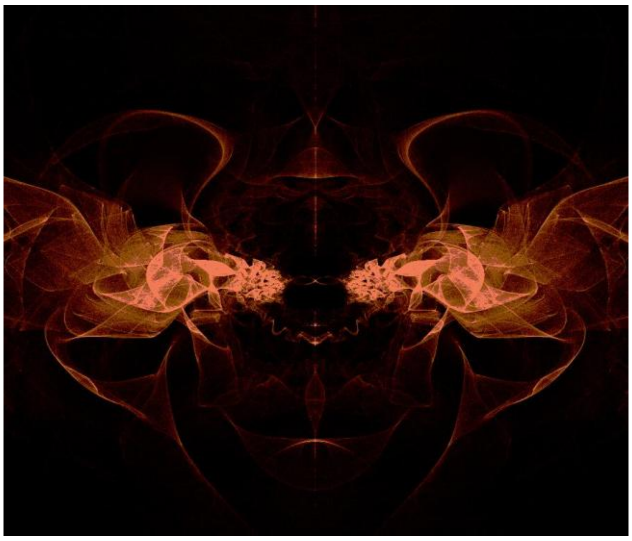

Fig. 3 : Output of Variation 2

VARIATION 3

Let g1(x) = sinh(x) and g2(x) = sin((x)+tan(x)).

The output is illustrated in Figure 4.

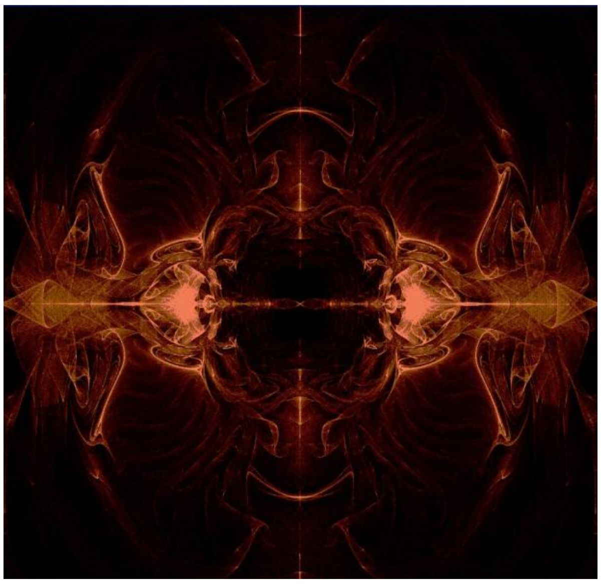

Fig. 4 : Output of Variation 3

VARIATION 4

Let g1(x) = sin(x) and g2(x) = sinh((x)+tanh(x)).

The output is illustrated in Figure 5.

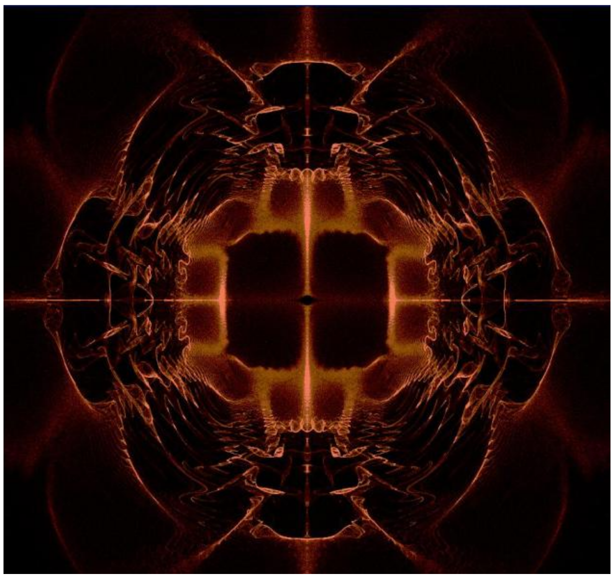

Fig. 5 : Output of Variation 4

VARIATION 5

Let g1(x) = sin(x)+cos(x) and g2(x) = sin((x)+tan(x)).

Let h=.25.

The output is illustrated in Figure 6.

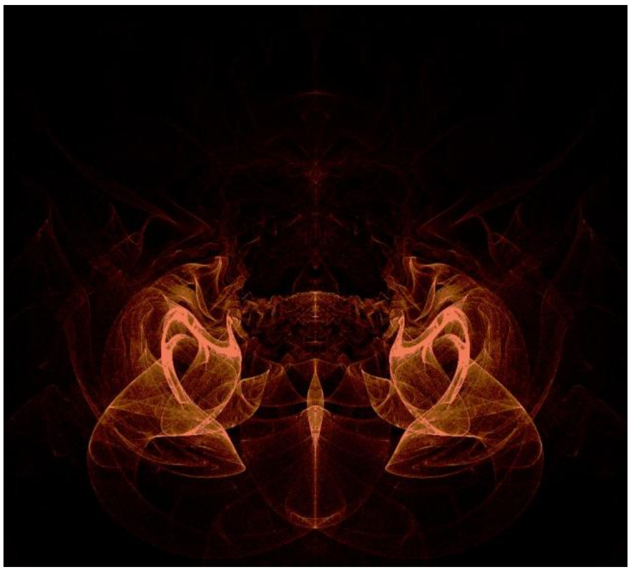

Fig. 6 : Output of Variation 5

4. THE EFFECT OF ADDING SWIRL

A nice floral effect is created if we add the swirl transformation to the spherical. For this x = real(z) and y = imag(z) have to be declared as floating point variables rather than as complex ones. Also we take simpler versions of Pickover transformations, viz,

x=x-h* (g1(y+g2(a*y)))

y=y-h*(g1(px+g2(a*px))).

Where px=x.

With these changes the relevant section of the sample code reads as follows:

```
x=x-(h*g1(y+g2(a*y)));
y=y-(h*g1(px+g2(a*px)));
z=cmplx(x,y);
x=x/norm(z),y=y/norm(z);
float r=sqrt(x*x+y*y);
float t=atan(y/(x));
x=r*cos(2*t),y=r*sin(2*t);
z=cmplx(x,y);
```

There is now no need of the rotation by 45 degrees.

With these changes the output of the sample code is displayed in Figure 7. Here h=.5,a=13, g1(x)= sin(x+tan(x)), g2(x) = tanh(x+tanh(x)).

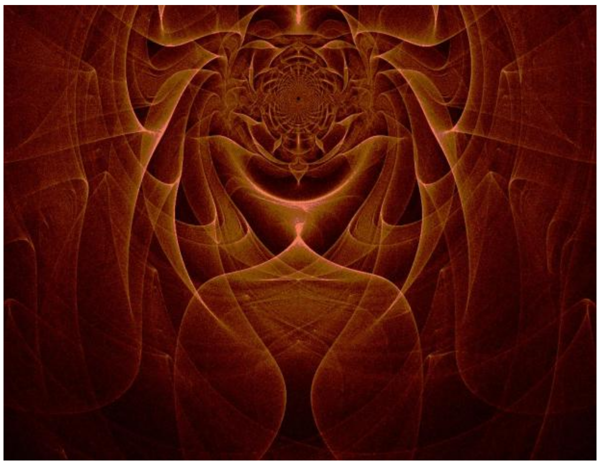

Fig. 7 : Output of modified code

Another example is displayed in figure 8. Here g1(x)= sin(x-tan(x)), g2(x)= tanh(x+tanh(x)).

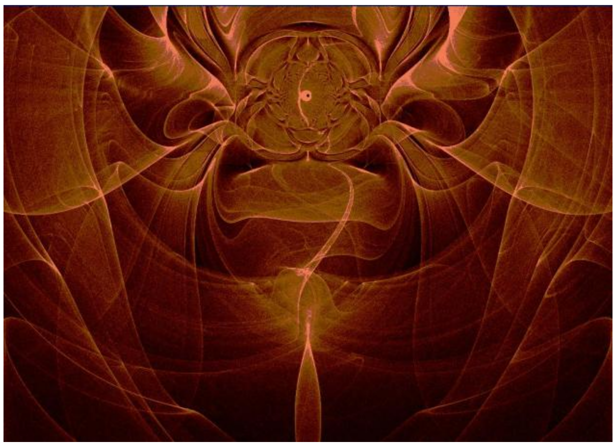

Fig. 8 : A variation of modified code

5. CONCLUSION

This paper presents the effect of applying a particular sequence of transformations on popcorn fractals. This sequence could be altered, or new transformations could be introduced to the sequence. Similarly transformations could also be used successfully on Chebychev fractals. These and other modifications would be explored in future work.

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

7. REFERENCES
[1] Barnsley, M. 1983 Fractals Everywhere, Academic Press.

[2] Brill, R. 1995 Embellished Lissajous Figures, The Pattern Book(ed. Pickover, C.).

[3] Davis, C. and Knuth, D.E. 1970 Number representations and dragon curves,Journal of Recreational Mathematics 3(1970) 66-81 and 133-149.

[4] Draves, S. 1992 The Fractal Flame Algorithm, flame3.com/flame-draves.pdf.

[5] Gangopadhyay, T. 2012 On generating skyscapes through escape-time fractals, International journal of Computer Applications 43(2012)17-19.

[6] Hofstadter, D.R. 1982 Strange attractors: Mathematical patterns delicately poised between order and chaos, Scientific American 245(May 1982)16-29.

[7] Pickover C. quoted in Fractint formula documentation,www.nahee.com/spanky/www/fractint/popcorn_type.html.

[8] Ruell,D. 1980 Strange attractors, Math Intelligencer 2(1980)126-137.

[9] Stevens, R. 1989 Fractal Programming in C, M&T Books.