Rhombus Substitution Tiling

Cye H. Waldman
cye@att.net

Abstract

Rhombic, or rhombus substitution, tiling is an outgrowth of triangle substitution tiling, but has not been identified as such to our knowledge. Many instances of such tilings can be found in the literature, but appear to be *ad hoc* tilings, without recognition that they are, at the core, the same tiling. With this new paradigm, we can recreate these tilings and infinitely many others with a single algorithm. We demonstrate how to create radial and spiral tilings with the Goldberg shift.

Introduction

Inasmuch as the words *rhombus* and *substitution* appear frequently in discussions of tiling, we will first clarify what *we* mean by rhombus substitution tiling.

It is well known that we can create radial and spiral tilings by starting with a deformed isosceles triangle. This has been described by Gardner (2001) and in greater detail by Grünbaum and Shephard (1986). Both attribute the idea to Goldberg (1955), who pointed out that if we take a radial tiling of isosceles triangles and slice it in half, the two half-planes can be shifted by one or more tiles to make a spiral form. More recently, Waldman (2014) has formalized the procedure and made a Matlab code freely available.

A rhombus tile is two triangle tiles that are concatenated at the base. Symmetric and antisymmetric concatenations are both possible, but only the latter will tessellate radially. The parent triangles may be symmetric or antisymmetric tiles (this refers to the congruent legs). Some examples are shown in Table 1.

Examples of rhombic Lightning and Lord & Wilson tilings can be found in Grünbaum and Shephard (p. 516). The latter can also be found in Dutch (1999) and Roskes (2014), along with straight rhombus tilings. The sigma tile was first shown by Waldman (2014) and is a prime example of rhombus tiling, as shown in Figure 1. As far as we know, these are all *ad hoc* tilings. With the present model these tilings and many others can be readily created. Moreover, most of these tiles are deformable (optional shapes for the same vertex angle) or parameterized (optional shapes for variable vertex angles) and these variations are available with little or no extra effort.

Finally, we present a complete Matlab program for rhombus substitution tiling with several built-in tiles. This program can be readily modified by the user in order to modify or add tiles.

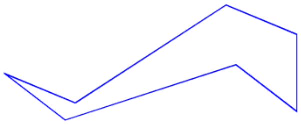
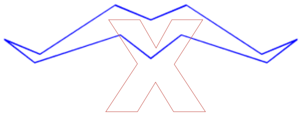
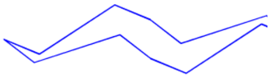
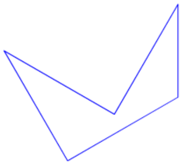
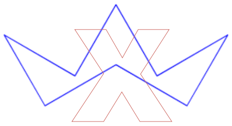
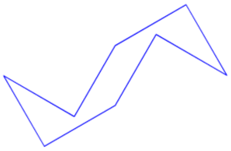
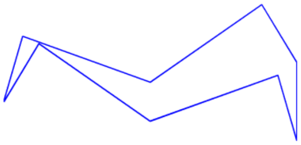
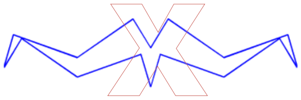
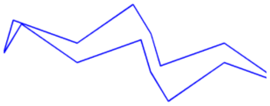
Tile	‘Triangle’	Symmetric ‘rhombus’	Antisymmetric ‘rhombus’
Lightning			
Lord & Wilson			
Sigma			

Table 1: Some examples of rhombic tiles.

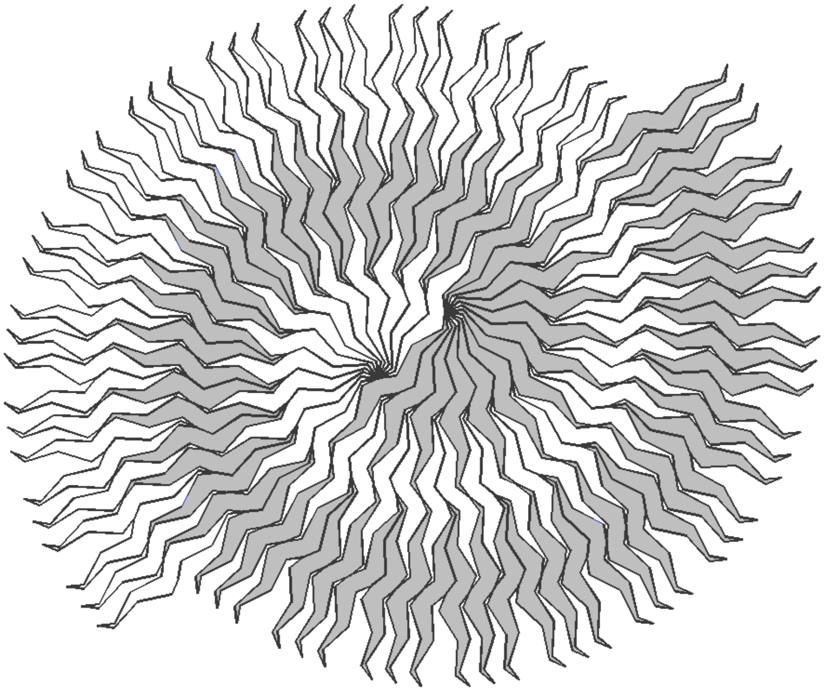


Figure 1: Rhombic tiling with Sigma tile.

Technical Discussion

RHOMBUS SUBSTITUTION TILING

Rhombus substitution tiling derives from the idea of Goldberg (1955), who pointed out that if we take a radial tiling of isosceles triangles and slice it in half, the two half-planes can be shifted by one or more tiles to make a spiral form. Here, however, we are using two isosceles triangles concatenated at the base. The general idea is shown in Figure 2. Moreover, the rhombuses can be distorted in infinite ways by replacing the legs with congruent lines, such as those in Table 1. In rhombus tiling the Goldberg shift is defined by two legs of the rhombus, that is, one-half the perimeter of the tile, or $s = 1/\sin(\alpha/2)$, where α is the vertex angle. As seen in Figure 2, the shift can take integer and half-integer values. And, of course, the shift can be positive or negative.

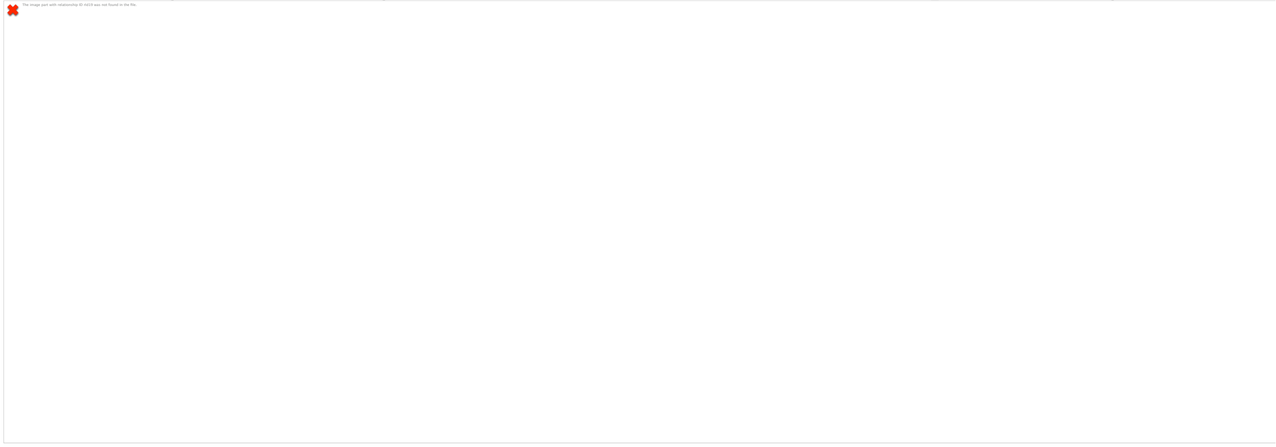
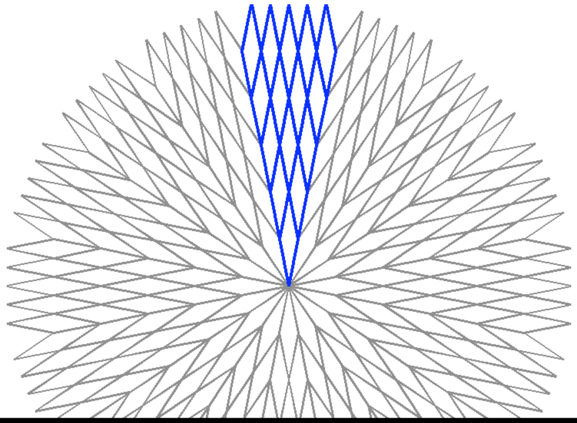

The image part with relationship ID rId19 was not found in the file.

Figure 2: Creating spirals from radial rhombic tilings (shifts of 0, ½, and 1).

We developed a simple and (almost) foolproof method for creating these tilings that we call R tiling. R is simply the tile resting on its vertex in a vertical orientation. All rhombic tiles have four points in common: the four corners of the parent rhombus , where, is the height of the parent rhombus.

Radial tiling can be constructed by corona or by sector. The latter is simpler and is adopted here. Referring to Figure 3, we see that the core consists of a single R-tile for each sector. Within the sector, the first corona consists of two tiles, touching at the waist, the second corona has three tiles, also touching at the waist, and so on. In all cases, the waist of the rhombus is unity. The n^{th} corona consists of n+1 R-tiles. Thus, for the n^{th} corona,

(1)



[redacted]

Figure 3: Rhombic tiling.

To set up for spiral tiling, it is convenient to rotate the sector Z counterclockwise so that it rests on the negative x-axis. Assuming the origin is at the vertex of the sector, we can say

$$Z = Ze^{i\pi/2}e^{-i\alpha/2} \qquad (2)$$

Radial tiling is then created by rotating this Z though all the sectors. Spirals are created as follows:

$$Z_{spiral} = \left[Z_{top} \quad Z_{bot} + g\Delta x \right] \qquad (3)$$

where Z_{top} and Z_{bot} are the sectors in the top and bottom halves of the plane, respectively, g is the number of the Goldberg shift, and s is half-perimeter of the reference rhombus, as defined above. This is amazingly simple, but there is one caveat that must be noted. First, the vertex angle of the reference triangles must be *mod* 360 and the number of sectors must be even. Or, let's just say that the vertex angle must be *mod* 180. Second, while the 'rhombus' itself is antisymmetric, there can be a problem tessellating radially because the coronas overlap. Specifically, if the parent 'triangle,' or one leg of the rhombus, is symmetric, then odd-numbered coronas are the negative conjugates of the values given by Eq. (1), i.e. $Z_{bot} \rightarrow -Z^{*}_{bot}$. That's all.

SOME SAMPLE TILES

Figure 4 shows the nine rhombus substitution tiles that are included with the enclosed program. All tiles nominally have a waist of unity; here they have been normalized to a unit height. The Voderberg tile is conspicuously absent from this collection. Those flexed feet seem to preclude tessellation in the rhombic mode. We leave this enigma as an exercise for the reader. Nevertheless, the tile in included with the enclosed program for your amusement.

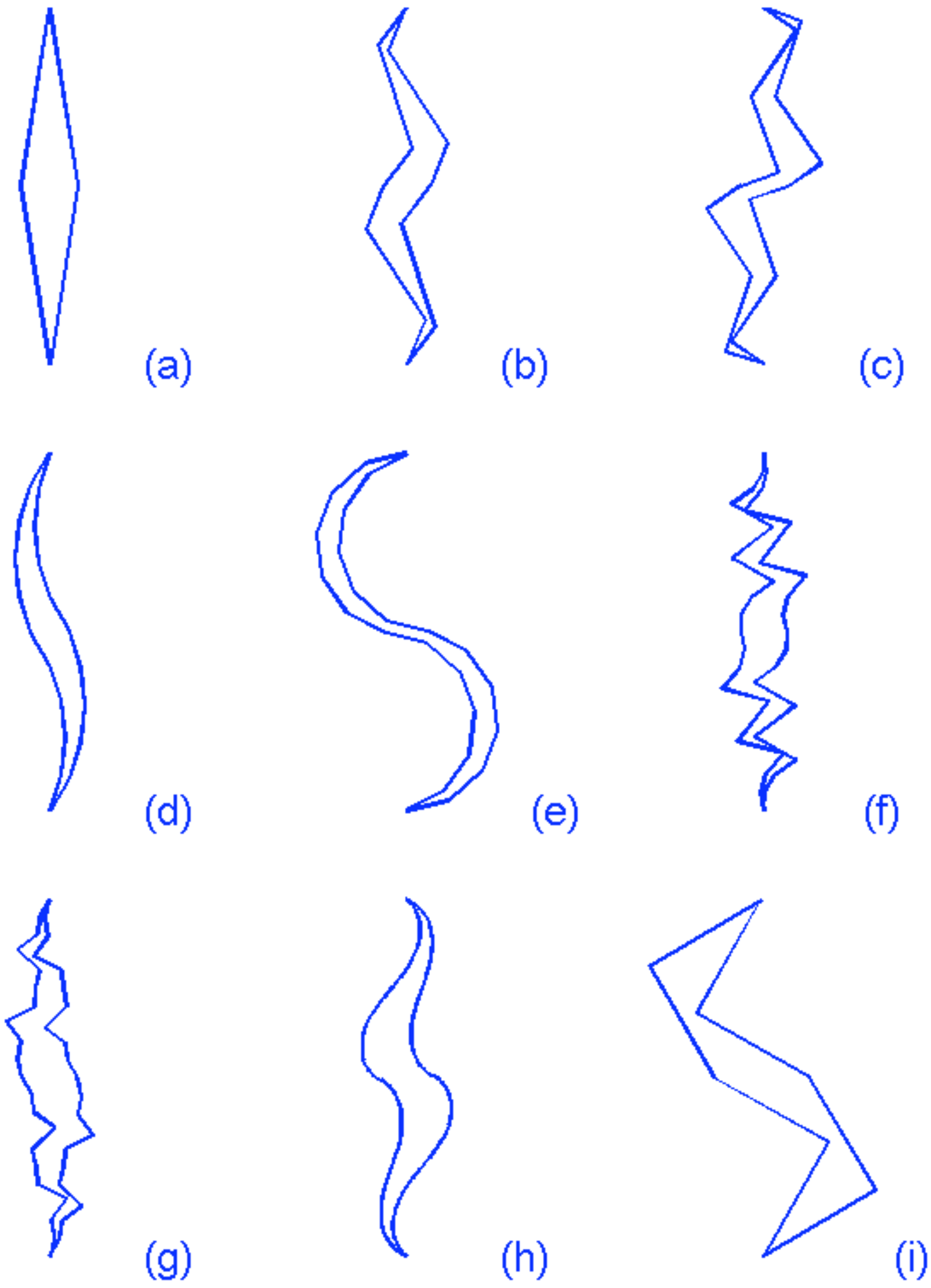


Figure 4: Built-in tiles for the enclosed rhombus substitution tiling program.

Table 2 shows the properties of these tiles. The ‘Generating tile’ is the reference ‘triangle’, i.e., the parent of the rhombic tile; ‘Description’ is just that, and ‘Figure’ refers to its position in Figure 4. Now, the columns ‘Deformable or free parameter’ is a user-selected value to customize the tile. ‘Range’ is the recommended ranges of that parameter. The Lightning and Sigma tiles permit a continuous variation in the tile shapes and thus allow animated tilings. These parameters are used in the program input.

Generating tile	**Description**	**Figure**	**Deformable or free parameter**	**Range**	**Asymmetry**

Triangle	Parent rhombus tile	(a)	α , vertex angle	*mod* 180	TRUE
Lightning	Bent heptagon	(b)	ℓ, a length	ℓ ⍰1.75	TRUE
Sigma	Bent enneagon	(c)	ℓ, a length	$\ell \sim$ 0.72-2.65	FALSE
Versatile	Variable sided non-convex odd-gon	(d)	*K*, no. of sides on one limb of the 'rhombus'	$K \geq 1$	FALSE
Reflexed odd polygon	Variable sided non-convex odd-gon	(e)	*K*, no. of sides on one limb of the 'rhombus'	$K \geq 1$	FALSE
Random antisymmetric	A random anit-sym icosihenagon	(f)	α , vertex angle	*mod* 180	TRUE
Random symmetric	A random symmetric icosihenagon	(g)	α , vertex angle	*mod* 180	FALSE
Cornu spiral	1-2 spiral segments	(h)	Number of spiral segments	1 or *not* 1	TRUE
Lord & Wilson	Variable vertex non-convex pentagon	(i)	α , vertex angle	*mod* 180	FALSE

Table 2: Properties of the built-in tiles.

The differentiation between antisymmetric and symmetric tiles (these refer to the generating tile) is that they tessellate differently, and this will impact the radial tiling. In plain English, antisymmetric tiles tessellate with themselves and symmetric tiles tessellate with their mirror (conjugate) selves.

The procedure for creating a 'rhombus' tile from a 'triangle' tile is quite simple. First, let's designate the triangle tile as V, with the vertex at the origin and a flat top. Now, the first step is to place an antisymmetric tile on the flat top. Thus we desinate $\mathrm{A} = -\mathrm{V} + i \cdot h$, where *h* is the height of the rhombus, . Now, since the 'triangle' must have $2K+1$ sides, the 'rhombus' tile is given very simply by the range of points from V and A shown here:

$$\mathrm{R}=\left[\mathrm{V}\left(1\text{:K+1}\right);\mathrm{A}\left(\text{K+3:end}\right);\mathrm{A}\left(1\text{:K+1}\right);\mathrm{V}\left(\text{K+3:end}\right)\right] \tag{4}$$

Tile coordinates are ordered counterclockwise starting from the vertex.

Figure 5 (a) shows a typical spiral, in this case, a lightning tile with two coronas and a shift of minus one-half. Now, the really cool thing about the program is that it doesn't care if R is input as concatenated rhombus substitution or as a pair of tessellated tiles, e.g., $\mathrm{R}=\left[\mathrm{V}\ \mathrm{A}\right]$. In this case, you recover a triangle substitution, as seen in Figure 5 (b). The striking stegosaurus fins are the residual tiles from the 'rhombus.' They can be optionally omitted in the program to fully reproduce a true triangle substitution tiling, as seen in Figure 5 (c),

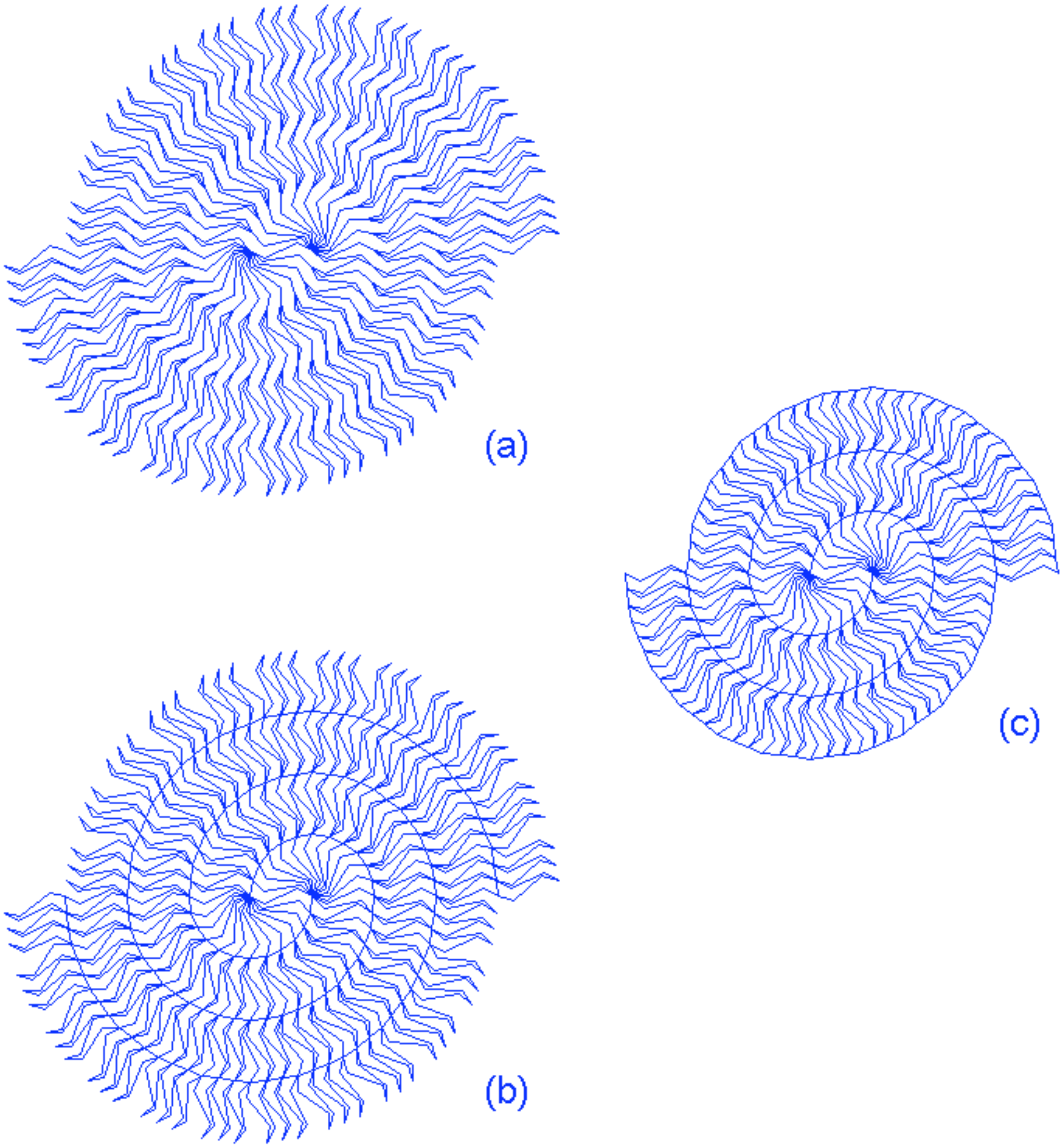


Figure 5: Examples of rhombus substitution tiling program output: (a) ‘rhombus’ configuration; (b) ‘triangle’ configuration (includes residual spikes); (c) ‘triangle’ configuration with optional removal of spikes (same as triangle substitution tiling).

ABOUT THE PROGRAM

A complete Matlab program, `RhombicTiling4NCB.m` is available in the Appendix. The input and output variables are described here. All calculations are carried out in `function RhombicRadialSpiral4NCB` by the radial sector method described above. The program plots the results of the specified tiling. It can be easily modified to alter or add to the built-in tiles.

The program is run from the command line as follows:

```
Z=RhombicTiling4NCB(tile,param,coronas,shift,rhombus,trim);
```

The output, `Z` is a complex matrix of tiles. The columns are the tiles and the rows are their complex coordinates. The input parameters are described in Table 3.

Parameter	Description	Remarks

`tile`	Built-in tiles:
	`'Triangle','Lightning',`
	`'Sigma','Versatile',`
	`'ReflexedPolygon',`
	`'Random','RandomSym',`
	`'Cornu','Lord&Wilson',`
	`'Voderberg' (exptl)`
		A string parameter; case insensitive
		Refer to Table 2

`param`	This is the parameter for the deformable tiles and the vertex angle for the triangle and random tiles	All angles are input in degrees All tiles require a parameter Refer to Table 2
`coronas`	The number of annuli about the central core tiles	
`shift`	The number of tiles to shift in order to create spirals by Goldberg's method	Positive or negative integers or half-integers; normally **`|shift| ≤ coronas`** Refer to Table 2
`rhombic`	Indicates whether to create rhombic (= 1) or triangle (*not* 1) tiling	
`trim`	Indicates whether to trim (=1) or not (*not* 1) to trim the stegosaurus fins (spikes)	Refer to Figure 5 and the preceding paragraph

Table 3: Program input parameters.

For example, the following code will produce the rhombic sigma spiral tiling of Figure 1 in 0.16 seconds:

```
tile=sigma;
param=1;
coronas=2;
shift=-1/2;
rhombic=1;
```

```
trim=0;
Z=RhombicTiling4NCB(tile,param,coronas,shift,rhombic,trim);
```

The program contains a complete set Versatiles [Grünbaum and Shephard (1979)] and reflexed odd-polygons; both are expressed parametrically in terms of the number of segments on each leg, *K* and the total number of sides is 2*K*+1. These are symmetric tiles.

The nominal vertex angle of the Cornu tiles is 20°; this can be changed in the program.

Voderberg tiling works properly only for radial tiling, i.e., `rhombic=0`, and not otherwise.

A gallery of randomly created images is shown below. The program for generating these, as well as testing the program, is embedded in the header section of the program. The title in each figure gives the name of the tile followed by the numerical input parameters: `param,coronas,shift`. Caution is advised when working with the random tiles; they *are* random and the legs may overlap.

There are two animations associated with this paper [Waldman (2015), this paper]. The first shows the buildup of a sector, then radial and spiral tilings. The second shows how a single parent tile can lead to three distinct tilings.

Summary

We have developed a new paradigm for radial and spiral tiling as an extension of triangle substitution tiling [Waldman 2014)]. We called it rhombic, or rhombus substitution, tiling because the prototiles are concatenated triangle substitution tiles. In so doing, we have brought under one roof, a collection of spirals that were previously thought to be *ad hoc* tilings. Moreover, we have extended the range of coronas and spirals (through the Goldberg shift) indefinitely. In addition, we have introduced a number of tiles that are compatible with rhombic tiling. Users can create new compatible tiles. The program is the engine to turn those into radial and spiral tilings.

We go a step further. Every tile built into the program is continuously or discretely variable and we've included two randomly generated tiles to boot. And finally, the program will also produce traditional and enhanced triangle substitution tilings. To say that there are an infinite number of tilings possible with this program is an understatement.

Now, we have nothing against *ad hoc* tiling, *per se*. Figure 1 started out that way, with a program that was described as 'too ugly for public consumption.' The question is, 'what are you going to do after that?' And that is what led us to where we are today.

References

Dutch, S. (1999). "Radial and Spiral Tilings,"
https://www.uwgb.edu/dutchs/symmetry/rad-spir.htm .
Gardner, M. (2001). *The Colossal Book of Mathematics*, Norton. (The chapter including triangle substitution tiling originally appeared in a 1977 *Scientific American* column.)
Goldberg, M. (1955). "Central Tessellations," *Scripta Mathematica*, **21**, pp 253-260.
Grünbaum, B. and Shephard G.C. (1979) "Spiral Tilings and Versatiles," *Mathematics Teaching*, No.88, Sept.1979, pp.50-1.
Grünbaum, B. and Shephard, G.C. (1986). *Tilings and Patterns*, Freeman. (Hardback edition)
Roskes , B. (2014). *Rotational and Spiral Tiling*, (GeomeTricks Aperiodic Patterns, Book 1), 3DVinci. See the complete set of GeomeTricks printed and e-books at
http://www.3dvinci.net/ccp0-catshow/GM.html.
These books are highly recommended, even if you don't use the SketchUp program, for the wealth of ideas on and examples of tiling within them. Nicely organized and profusely illustrated.
Waldman, C.H. (2014). "Voderberg Deconstructed & Triangle Substitution Tiling,"
http://curvebank.calstatela.edu/waldman9/waldman9.htm.
Waldman, C.H. (2015). "Rhombus Substitution Tiling,"
http://curvebank.calstatela.edu/waldman14/waldman14.htm.

Gallery

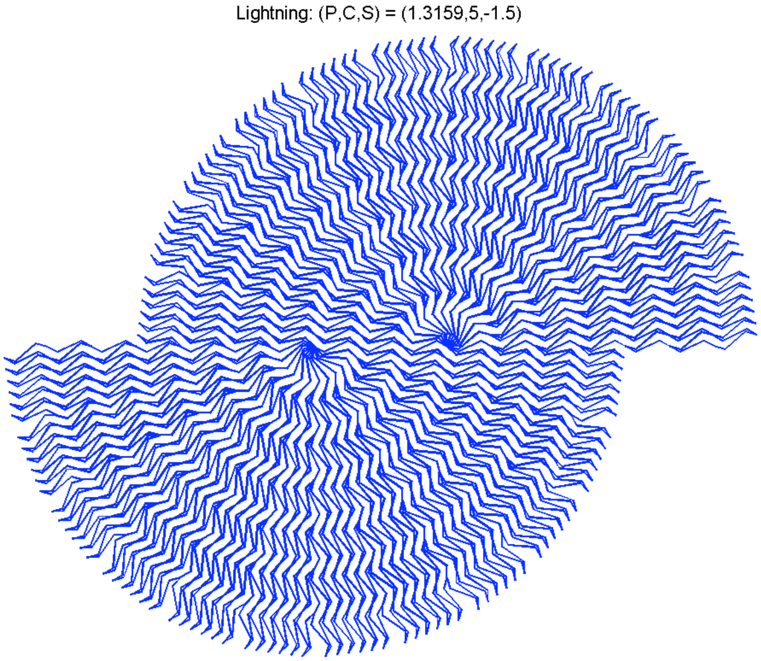


Figure 6: Lightning rhombic spiral tiling.

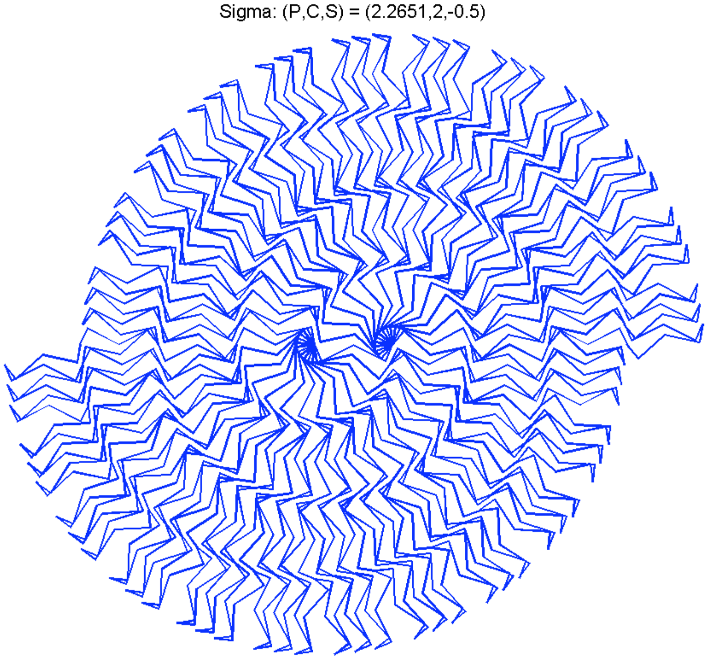


Figure 7: Sigma rhombic spiral tiling.

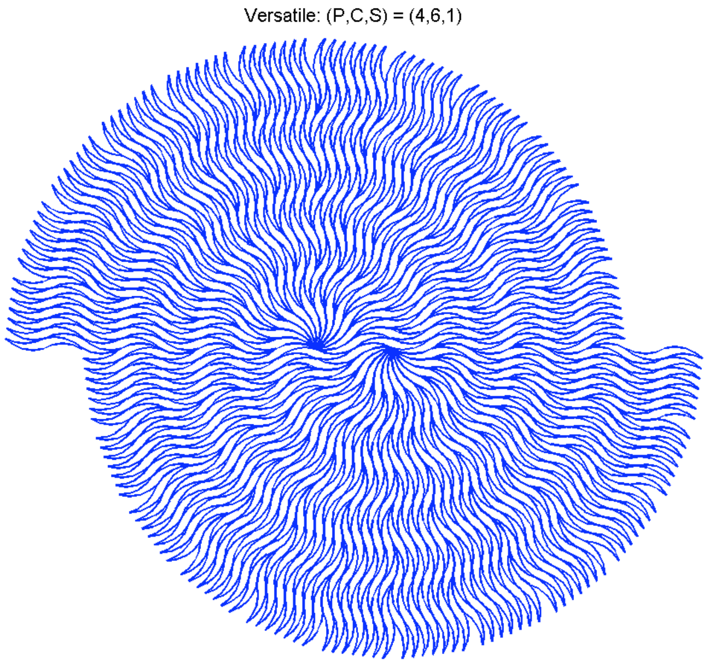


Figure 8: Versatile rhombic spiral tiling.

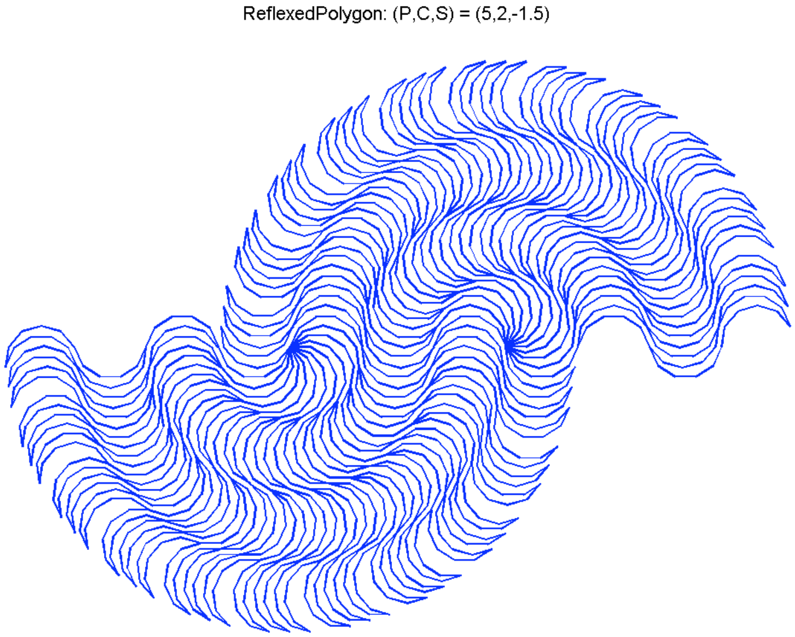


Figure 9: Reflexed odd-polygon rhombic spiral tiling.

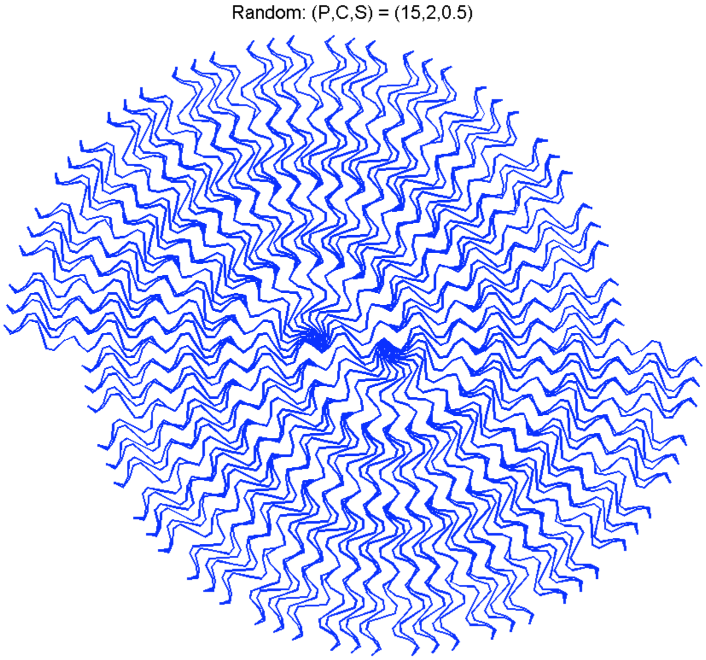


Figure 10: Random antisymmetric icosihenagon rhombic spiral tiling.

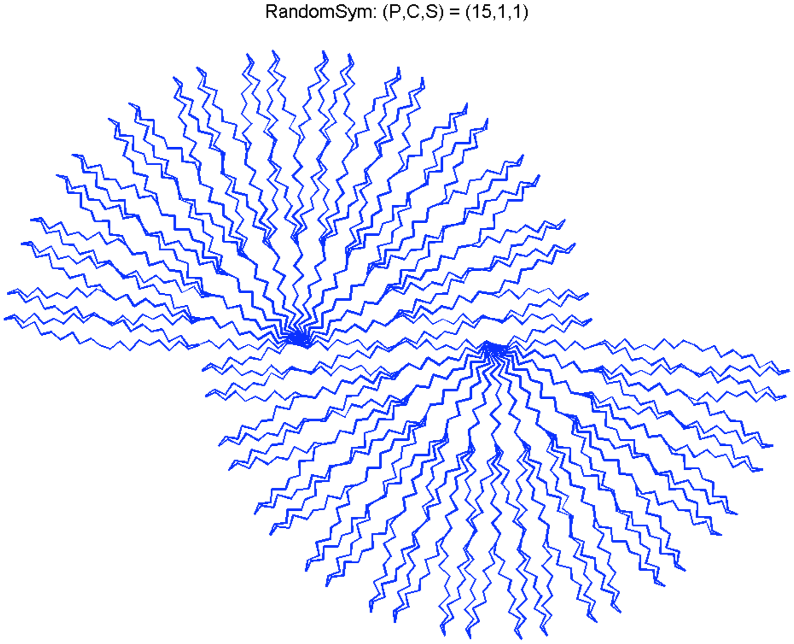


Figure 11: Random symmetric icosihenagon rhombic spiral tiling.

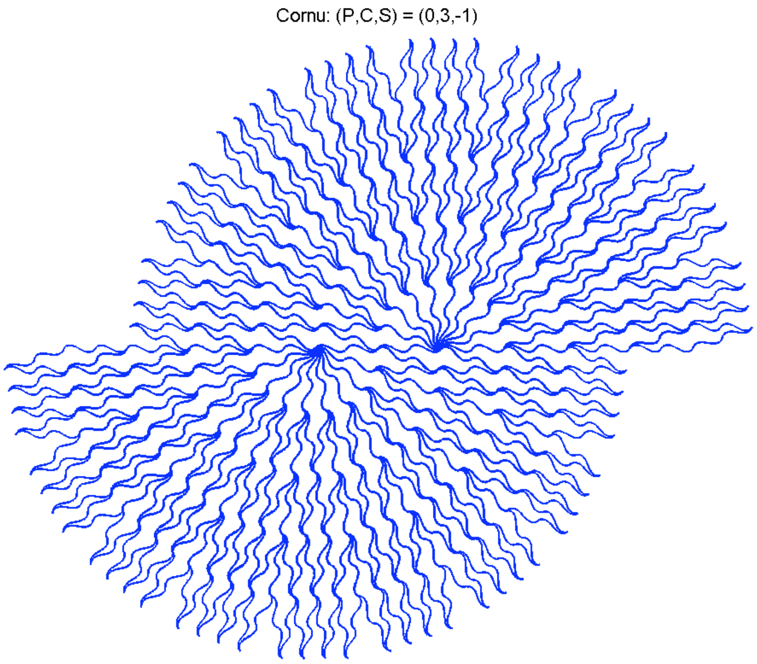


Figure 12: Cornu rhombic spiral tiling.

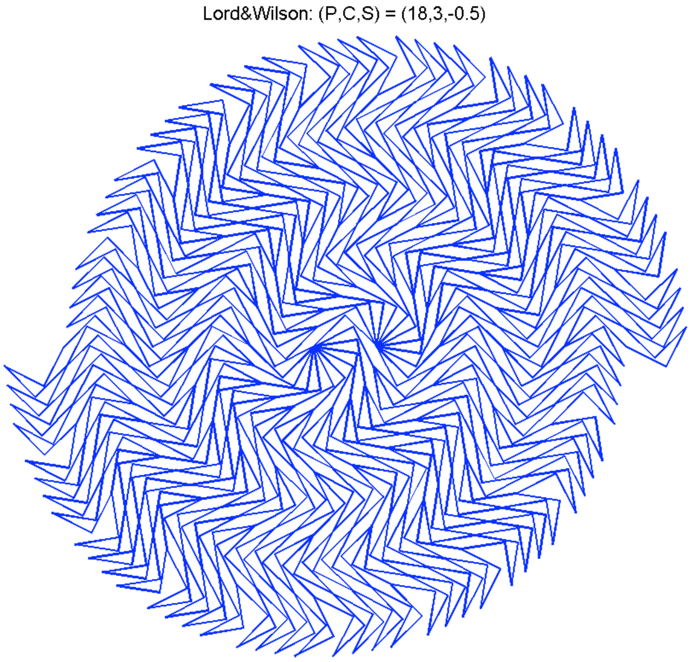


Figure 13: Lord & Wilson rhombic spiral tiling.

Appendix: Rhombus Substitution Tiling Program

```
function Z=RhombicTiling4NCB(tile,param,coronas,shift,rhombic,trim)
% this program was created for the National Curve Bank to accompany the
% submission "Rhombus Substitution Tiling,"
% see http://curvebank.calstatela.edu/waldman11/waldman11.htm for details
%--------------------------------------------
% HERE IS A SHORT PROGRAM FOR RANDOM TESTING
% function TestRhombicTiling
% % title says it all; average execution time <2 seconds
% close all
% mod360=[12 15 18 30 36 45 60]';
% tiles={'Triangle','Lightning','Sigma','Versatile','ReflexedPolygon',...
%     'Random','RandomSym','Cornu','Lord&Wilson','Voderberg'};
%
% for k=1:length(tiles)
%     params=[mod360(floor(1+7*rand)),1.75*rand,0.72+(2.65-0.72)*rand,...
%         floor(1+9*rand),floor(1+12*rand),mod360(floor(1+3*rand)),...
%         mod360(floor(1+3*rand)),round(rand),mod360(floor(1+3*rand)),...
%         111+(153-111)*rand,1];
%     coronas=floor(1+6*rand);
%     shift=(-6+floor(12*rand))/2;
%     rhombic=1;
%     trim=0;
%     Z=RhombicTiling4NCB(tiles{k},...
%         params(k),coronas,shift,rhombic,trim);
%     title([tiles{k} ': (P,C,S) = (' num2str(params(k)) ','...
%         num2str(coronas) ',' num2str(shift) ')'])
%     assignin('caller',['Z' num2str(k)],Z)
% end
%
% return
%--------------------------------------------
% Copyright 2014, Cye H. Waldman
% Let me know how this works out for you at cye@att.net

tile=lower(tile);
switch tile

    case 'triangle'
        alef=param*pi/180;
        x=1;
        h=1/(2*sin(alef/2));
        S=[x;h;h];
        phi=[0;pi/2+alef/2;3*pi/2-alef/2];
        z=cumsum(S.*exp(i*phi));
        z=[0;z];
        V=conj(z)-1/2+i*max(abs(imag(z)));
        A=z;
        sectors=2*pi/alef;
        if mod(sectors,2)>0
            error('Need an even integer number of sectors for spirals')
        end
        antisym=1;
        V=[V(3:4);V(2:3)];
    case 'lightning'
        alef=pi/12;
        sectors=2*pi/alef;
        s=1/(2*sin(alef/2));
        l=1;
        theta=pi/3;
        z=l*exp(i*theta);
        z=[0;z];
        z=[z;z-z(2)+i*s];
        z=[z;flipud(z*exp(i*alef))];
        V=z*exp(-i*alef/2);
        A=(-z+i*s)*exp(-i*alef/2);
        antisym=1;

    case 'sigma'
        alef=pi/12;
        sectors=2*pi/alef;
        s=1/(2*sin(alef/2));
        lratio=param;
        f=@(x) cos(x)./cos(pi/2+x)+lratio;
        theta=fzero(f,.5);
        phi=acos(-cos(theta)/2);
        l1=s/2*sin(phi-pi/2)/sin(pi-phi+theta);
        l2=2*l1;
        z=[0;l1*exp(i*theta);l1*exp(i*theta)+l2*exp(i*phi)];
        z=[z;flipud(conj(z(1:end-1))+i*s)];
        z=z*exp(-i*alef/2);
        z=[z;flipud(z*exp(i*alef))];
        V=z;
        A=conj(z)+i*s*exp(-i*alef/2);
        V=-conj(V);
        A=-conj(A)+1;
        antisym=0;

    case 'versatile'
        K=param;
        v=Versatile(K);
        v=-conj(v);
        % figure;plot(V);axis equal
        % hold on;plot(v,'r')
        V=v;
        n=6*K;
        alef=2*pi/n;
        sectors=n;
        antisym=0;

    case 'reflexedpolygon'
        K=param;
        v=ReflexedPolygon(K);
        v=-conj(v);
        V=v;
        n=2*K+1;
        alef=pi/n;
        sectors=2*pi/alef;
        antisym=0;
    case 'random'
        antisym=1;
        a=param;
        alef=a*pi/180;
        h=1/(2*tan(alef/2));
        s=1/(2*sin(alef/2));
        % let's make a 5-legged line,
        y=linspace(0,s/2,6)';
        x=s/3*(-.5+1*rand(4,1));
        % get the correct endpoints
        x=[0;x;0];
        z1=complex(x,y);
        if antisym
            z=[flipud(-z1);z1(2:end)];
        else
            z=[flipud(conj(z1));z1(2:end)];     % for symmteric case
        end
        z=z-z(1);
        % make the tile
        z=[z*exp(-i*alef/2);flipud(z*exp(i*alef/2))];
        V=z;
        sectors=2*pi/alef;
        if mod(sectors,2)>0
            error('Need an even integer number of sectors for spirals')
        end

    case 'randomsym'
        antisym=0;
        a=param;
        alef=a*pi/180;
        h=1/(2*tan(alef/2));
        s=1/(2*sin(alef/2));
        % let's make a 5-legged line,
        y=linspace(0,s/2,6)';
        x=s/3*(-.5+1*rand(4,1));
        % get the correct endpoints
        x=[0;x;0];
        z1=complex(x,y);
        if antisym
            z=[flipud(-z1);z1(2:end)];
        else
            z=[flipud(conj(z1));z1(2:end)];     % for symmteric case
        end
        z=z-z(1);
        % make the tile
        z=[z*exp(-i*alef/2);flipud(z*exp(i*alef/2))];
        V=z;
        if antisym
            A=-V+1/2+i*h;
        else
            A=conj(V)+1/2+i*h;     % for symmteric case
        end
        sectors=2*pi/alef;
        if mod(sectors,2)>0
            error('Need an even integer number of sectors for spirals')
        end
```

```
case 'cornu'
    integrand=inline('exp(i*(a(1)+s.*(a(2)+s.*(a(3)))))','s','a');
    singlet=param;
    alef=20*pi/180;
    smax=.731;
    npts=101;
    s=linspace(-smax,smax,npts)';
    dels=s(2)-s(1);
    a=[0 0 3]; % classic Cornu spiral --> pi/3
    z=cumtrapz(integrand(s,a))*dels;
    theta=angle(z(end)-z(1));
    z1=z*exp(i*(pi/2-theta));
    if singlet
        z3=flipud(z1*exp(i*alef));
        l=abs(z(end)-z(1));
        x=2*l*sin(alef/2);
        z2=flipud((z-z(end))*x/l*exp(i*0*alef/2));
        theta1=angle(z3(1)-z1(end));
        theta2=angle(z2(end)-z1(1));
        z2=(z2-z1(1))*exp(i*(theta1-theta2))+l*i;
        z=[z1;z2(2:end);z3(2:end)];
        z=z*exp(-i*alef/2);
        % and finally, slace to unit top
        z=z/x;
        v=-z+i*2*imag(z(101));
        w=[z(1:101);v(202:end);v(2:101);z(202:end)];
        h=imag(w(101));
    else
        z1=[z1;z1(2:end)+i*max(abs(z1))];
        z3=flipud(z1*exp(i*alef));
        l=abs(z1(end)-z1(1));
        x=4*l*sin(alef/2);
        z2=flipud((z-z(end))*x/l*exp(i*0*alef/2));
        theta1=angle(z3(1)-z1(end));
        theta2=angle(z2(end)-z1(1));
        z2=(z2-z1(1))*exp(i*(theta1-theta2))+l*i;
        z=[z1;z2(2:end);z3(2:end)];
        z=z*exp(-i*alef/2);
        % and finally, slace to unit top
        z=2*z/x;
        v=-z+i*2*imag(z(201));
        w=[z(1:201);v(302:end);v(2:201);z(302:end)];
        h=imag(w(201));
    end
    sectors=2*pi/alef;
    antisym=1;

case 'lord&wilson'
    a=param;
    alef=a*pi/180;
    n=2*pi/alef;
    theta=[a;90;180-a;270;a;90;180-a;270]*pi/180;
    phi=cumsum(pi-theta);
    z=cumsum(exp(i*phi));
    z=[0;z];
```

```
        beth=angle(z(5));
        z=z*exp(i*(beth+alef));
        delx=real(z(3)-z(7));
        % PENTAGON OPTION
        za=[z(1:3);z(7:9)];
        zb=[z(3:7);z(3)];
        % create a unit-top' tile
        V=za/(abs(za(3)-za(4)));
        sectors=2*pi/alef;
        antisym=0;

    case 'voderberg'
        alef=12*pi/180;
        b=param;
        beth=b*pi/180;
        L=2*sin((pi-alef)/2)/cos(beth-pi/2);
        x=(csc(alef/2)/2-L*cos(pi-beth))/2-sin(alef/2);
        S=[1;x;L;x;1;x;L;x;1];
        theta=[alef;(3*pi-alef)/2;(2*pi-beth);beth;(pi+alef)/2;
          (pi-3*alef)/2;(2*pi-beth);beth;(pi+alef)/2];
        phi=cumsum(pi-theta);
        z1=cumsum(S.*exp(i*phi));
        z1=[0;z1];
        z3=-z1+1/2+i*max(imag(z1));
        V=z1;
        A=z3;
        sectors=2*pi/alef;
        antisym=1;

    otherwise
        error('Please select from the available tiles')
end

% avoid potential numerical problems
if rem(sectors,1)>0 sectors=round(sectors); end;
if strcmpi(tile,'cornu')
    R=w;
    V=z;
    A=-V+i*2*h;
else
    K=length(V)/2-1;
    h=max(imag(V));
    A=-V+i*2*h;
    R=[V(1:K+1);A(K+3:end);A(2:K+1);V(K+3:end)];
    % for those who wish to look at the symmetric rhombic tile
    B=-conj(A);
    S=[V(1:K+1);B(K:-1:1);B(end-1:-1:K+2);V(K+3:end)];
end
if ~rhombic R=[V A]; end;

Z=RhombicRadialSpiral4NCB(R,sectors,coronas,shift,~antisym,rhombic,trim);
figure;plot(Z,'b','LineWidth',1);axis equal;axis off;set(gcf,'Color','w')
v=axis;axis(1.01*v)

return
```

```
function Z=RhombicRadialSpiral4NCB(R,sectors,coronas,shift,issym,rhombic,trim)
% this function produces a normal radial tiling of unit antisymmetrical
% Rhombus D, with origin at the center, in sectors

isodd=inline('rem(number,2) == 1;','number');

alef=2*pi/sectors;
dely=1/(2*tan(alef/2));     % corona height

z=R;
for r=1:coronas
    this=[];
    % calculate the rows
    for k=1:r+1
        this=[this R+k];
    end
    this=this-1-r/2;
    if issym && isodd(r); this=-conj(this); end
    if ~rhombic
        if r==coronas && trim; this=this(:,1:2:end); end
    end
    z=[z this+i*dely*r];
end

% rotate to lie on negative x-axis
z=z*exp(i*pi/2)*exp(-i*pi/sectors);

Z=z;
for k=1:sectors-1
    Z=[Z z*exp(-i*k*alef)];
end
% and finally the spiral if shift~=0
if abs(shift)>0
    Ztop=Z(:,1:end/2);
    Zbot=Z(:,end/2+1:end);
    if issym && isodd(abs(shift*2))>0; Zbot=-conj(Zbot); end
    delx=1/(sin(alef/2));
    Z=[Ztop Zbot+shift*delx];
end

return
```

```
function z=Versatile(K)
% generates the Versatile on the basisof the number of segments on the
% congruent legs, K >= 1

% basic required angles and sides
n=6*K;
theta=2*pi/n;
alef=pi/3;
beth=2*pi*(n-3)/(3*n);
phi=pi*(n-2)/n;

% internal angles, ccw from vertex
THETA=theta;
for k=1:K-1
    THETA=[THETA;2*pi-phi];
end
THETA=[THETA;alef;beth];
for k=1:K-1
    THETA=[THETA;phi];
end

PHI=cumsum(pi-THETA);
z=cumsum(exp(i*PHI));
z=[0;z];
upright=angle(z(K+2)-z(K+1))-pi;
% upright=angle(z(end-1)-z(end));
z=z*exp(-i*upright);

return


function z=ReflexedPolygon(K)
% generates the reflexed odd-gon on the basisof the number of segments on
% the congruent legs, K >= 1

% create an n-gon
n=2*K+1;
theta=(n-2)*pi/n*ones(n,1);
phi=cumsum(pi-theta);
z=cumsum(exp(i*phi));
z=[0;z];

zeta=angle(z(K+1));
w=z*exp(-i*zeta);
w=[conj(w(1:K+1));w(K+2:end)];
z=w*exp(i*zeta)*exp(-i*phi(1)/2);

return
```

Please notify us at cye@att.net if there are any problems. And, by all means, let us know of any new tiles or tilings you develop for the program.