NBS
PUBLICATIONS
A11102 616247
NAT'L INST OF STANDARDS & TECH R.I.C.
A11102616247
Ressler, Sanford/Tiletool : a graphical
QC100 .U56 NO.86-3488 1986 V19 C.1 NBS-P

3488

Tiletool: A Graphical Interface for the Exploration of Generalized Penrose Tilings

Sanford Ressler

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Engineering Laboratory
Center for Manufacturing Engineering
Factory Automation Division
Machine Intelligence Group
Gaithersburg, MD 20899

November 1986

QC
100
U56
#86-3488
1986
c.2

sored by
eau of Engraving and Printing

NBS
RESEARCH
INFORMATION
CENTER

NBSIR 86-3488

TILETOOL: A GRAPHICAL INTERFACE FOR THE EXPLORATION OF GENERALIZED PENROSE TILINGS

Sanford Ressler

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Engineering Laboratory
Center for Manufacturing Engineering
Factory Automation Division
Machine Intelligence Group
Gaithersburg, MD 20899

November 1986

Sponsored by
Bureau of Engraving and Printing

U.S. DEPARTMENT OF COMMERCE, Malcolm Baldrige, *Secretary*
NATIONAL BUREAU OF STANDARDS, Ernest Ambler, *Director*

TileTool: A Graphical Interface for the Exploration of Generalized Penrose Tilings

Sanford Ressler
National Bureau of Standards
March 1986

Introduction

A graphical system to experiment with non-periodic tilings of a plane has been developed as part of research supported by the Bureau of Engraving and Printing (BEP) and the National Bureau of Standards (NBS). Non-periodic tilings have several properties which may be useful in the domain of security printing. Documents such as currency, stocks, bonds, and checks which have values much greater than the material they are composed of, are security documents. The forging of such documents has continued to be a problem as access to sophisticated equipment becomes widespread.

At first glance, non-periodic tilings appear symmetric. Upon further examination, however, one can see that they are non symmetric in their detailed structure (Figures 1 & 2). These properties make duplication more difficult and allow for verification of originals. Verification is possible because the form of the asymmetry can be controlle. The patterns can be made extremely complex in their detail. Easing the manipulation of the parameters used in the tile generation algorithm was one of the goals for the graphics system. A flexible mechanism for changing visual characteristics such as color, texture and positioning, was also desired (Figures 3 & 4).

Non-Periodic Tilings

Penrose tilings [1] and related non-periodic tilings have several interesting properties. First, like all simple tilings they completely fill the plane with copies of a finite number of distinct geometric objects (tiles). In addition, given a tiling subset there is no fixed offset which can be repeatedly applied to locate identical subsets. The subset will not appear in the tiling at fixed (periodic) intervals.

The tilings are effectively projections from N dimensional space onto a 2D plane [1]. The particular algorithm used in TileTool is a generalization of the method of N.G. deBruijn [2], equivalent, in one case, to those produced by the geometric approach of R. Penrose [3]. It was first implemented at NBS by Seymour Haber and forms the heart of the tile generation portions of the tool. The tiles are generated by this algorithm in the following manner:

- Take NG grids, where a grid is a set of parallel lines (Figure 5a)
- The number of lines, line range, is specified as a non-negative integer, where a line range of 0 gives one line, a line range of 1 gives 3 lines numbered -1 0 1, etc. (Figure 5a)
- Copies of the grid are rotated. If we have NG grids, each grid is rotated (OV * π) / NG radians relative to the previous grid rotation. (Figure 5b) Variations of OV (Ovalness) causes the tile to warp into ovals.

- Each grid is then translated a slight, usually random, amount to form an irregular meshwork where no more than two lines pass through any point (Figures 5c & 5d).
- Each vertex in the mesh has exactly four surrounding areas. Associated with each area is a unique point on the plane; the four points surrounding each vertex can be connected to form a rhombus. A single point is derived from each of these four mesh areas via complex vector manipulations. Each of these points is connected to form one rhombus. This process is continued to complete the tiling. See appendix for details of the algorithm.

Goals

Our goal was to produce an interactive system which would allow the user to generate tilings quickly. Control of the visual elements involved is quite important. Flexibility in the manipulation of the visual elements was necessary to allow the system to be easily integrated into a number of applications. By allowing the user to experiment with the tilings in a non-obtrusive way, many possibilities for the use of the tiles may be generated. Most of the available controls in the system deal with the visual presentation of the tiling. Visual cues such as color, polygon outlining, patterning, scaling and positioning are all manipulatable. Much finer control over these parameters will surely be necessary to bring the system out of a research environment and into a production setting.

Implementation Environment

TileTool was implemented on a Sun Microsystems workstation with a three-button mouse. A high level tool provided by Sun, panel subwindows, was used in the implementation. These panels are a mechanism for providing graphic input in the form of buttons, sliders, checklists, and menus. The use of the panel subwindows aided greatly in the implementation of the interface. The panels also proved limiting by not allowing the use of many typical graphics techniques, such as dragging and explicit button up/down detection while in a panel. Thus, the interface is a compromise between generality and ease of implementation.

User Interface

The challenge for creating a user interface is to present the user with an environment which is flexible yet not overwhelming to use. The user must have the ability to manipulate parameters used by the tile generating algorithm as well as a large selection of visual parameters.

One particularly interesting class of images is that of circular shapes with the interior tiled. These are in essence a single "cell" of a tiling, produced when the line range is zero (Figures 3 & 6).

The user must manipulate the following variables in order to affect the patterns:

- NG: Number of Grids; This is effectively the number of dimensions. NG affects the overall complexity of the tiling. Figure 3 illustrates tilings with increasing NG.
- LR: Line Range; The extent of each grid. LR affects the extent of the tiling, rapidly increasing the size of the tiling as it is increased. Figure 4 illustrates three tilings, with LR of 0 in red, 1 in green and 2 in blue (The gray background is for effect only).
- OV: Ovalness; The total amount of rotation for all the grids. OV affects the roundness of the tiling, and can warp it into an oval or a pure grid (Figure 7).
- Central/Non-Central: When central, the translations applied to each grid are reduced by several orders of magnitude. A central tiling is much more symmetrical in appearance than is a non-central tiling (Figure 8).
- There is no direct control over the translations aside from the Central/Non-Central flag. At the present time the values for the translations are generated randomly.

The rhombuses which are not connected with the central fully tiled (Figure 8). are a side effect of the way the present algorithm works. If for example we are using five grids, meshes surrounded by lines from all five grids will be part of the connected fully tiled area. As we travel farther away from the center, meshes are formed by four, and three grids. These meshes evolve into rhombuses which are not connected to the main area. Simply increasing the line range will effectively increase the extent of the fully connected area. If it were possible to have an infinite number of lines then the infinite plane would be completely tiled.

Some terminology used below in describing the interface is: Selection or clicking, which refers to pressing the left mouse button down and then releasing to pick something on the screen. To click on some item means that the left mouse button is pressed anad released while the cursor is on top of that item. A pop-up menu is a menu which appears at the cursor position when the right mouse button is pressed. While still pressing the right mouse button, the user may move the cursor up and down within the menu to pick one entry. The menu disappears when the button is released. A pop-up box is a box which appears on the screen that contains information and requires the user to press any button to continue, or a specific button to confirm something. A panel is a rectangular region within which a number of selectable related items are located.

A series of control panels was created because the user can interact with a large number of parameters. These panels may be opened or closed by selecting them from the main control panel. The presentation of the various control panels is iconic in form, for fast recognition by the experienced user and for a compact visual layout (Figure 9). When first confronted with the array of icons the images are highly confusing. A consistent mechanism for help was established to remedy the situation and to provide for explanations of the various parameters and icons. The right mouse button pops up menus, which always provide help, either in the form of text, images or both (Figure 10).

Each panel has a title bar (Figure 11) which contains the icon for the panel and the textual name of the panel. The title bar is the main area for more detailed help and information. Clicking on the title bar itself will cause a box of text to appear describing in general terms the purpose of that particular panel (Figure 12). Popping up a menu on (Figure 13) the title bar will present the user with a short description of each item in that panel. Selecting an item from the menu will pop up a box containing a detailed description of that item (Figure 14). In this way the user has three levels of help available at any time. If the user pops up a menu for any individual item a small description of that item appears (Figure 15). The menus provide descriptions for iconic images.

It was also felt desirable to allow the user to make all of the control panels invisible so as to view a "clean canvas". The icon on the main control panel's title bar implements this feature and then changes the cursor to an image of the mouse with the middle button highlighted to indicate that the middle mouse button will cause the control panels to appear again.

There are no message areas; the values of the various variables are explicitly represented. This "what you see is what you get" design eliminates a potential source of confusion to the user. Any supplemental information, either help or error messages, is presented in pop-up boxes which require the user to press a button to continue.

Another principle used in designing the interface is redundancy. Usually there is more than one way to accomplish something. Selecting a panel can occur by selecting the panel icon or using the menu selection. Many numeric values may be set by a technique, called the incrementor, which provides three redundant ways to change a parameter.

Incrementor

The incrementor is a technique for visually organizing a set of variables and for changing the values of those variables. This technique is a visually compact, flexible and consistent mechanism for modifying the variables presented to the user. The names of the variables appear in the center of one of two pairs of arrows (Figure 16). The current value of the variable is visible next to the arrows. Selecting points on the arrows cause the current value to be incremented, or decremented, by the increment amount. The arrows consist of 4 parts each. Each part will multiply the increment by a factor of 1, 10, 100, or 1000 as the distance from the center is increased. A pop-up menu visually and textually describes the actions (Figure 17).

Selecting the variable name, or popping up a menu while over the name, lets the user pick a variable to manipulate. If desired, the user may also simply select the value, by clicking on the left mouse button, and use the keyboard to type on top of the current value to change it. The increment value itself may be changed by selecting it and then typing on top of the value. The current value of all variables may be seen by popping up a menu when on top of the variable name in the center of the arrows. (Figure 18). The user has three effective and redundant mechnisms for changing the value: using the arrows; using the pop-up menus; and direct overtyping.

Shortcuts

There are several mechanisms to shortcut some operations of the system. To close a panel, the user may click on the icon in the title bar. This is usually more convenient than going back up to the main control panel to close the panel. Another shortcut is to start a tiling by selecting any point on the canvas to begin a tiling. This allows the user to start a tiling while all the panels are invisible.

The primary shortcut available is the "All Variables" panel which uses the incrementor technique to give the user access to all the numeric variables in the system (Figure 18). An experienced user can quickly open up just the one panel and set the variable desired, without opening and closing several panels.

Summary

A system has been built which enables researchers to generate images based on generalized Penrose tilings. A large variety of visually interesting patterns which conform to the constraint of non-periodicity may be rapidly produced. The interaction has been made usable by organizing a set of flexible, consistent, and redundant mechanisms for the selectio and modification of the various parameters. This graphical tool will aid in the design and examination of these patterns.

Acknowledgments

The author would like to thank Ted Hopp of NBS and Edward Graminsky of BEP for their support, Seymour Haber of NBS who implemented the original version of the de Bruijn algorithm and also explained many of the aspects of non-periodic tilings, and Jonathan Kay who coded the first version of the incrementor. Last, but never least, Russ Kirsch for making this project possible.

References

[1] Kramer, P. & Neri R. *"On Periodic and Non-periodic Space Fillings of Em Obtained by Projection"*, Acta Cryst., pp 580-587, 1984.

[2] de Bruijn, N.G. *"Algebraic Theory of Penrose's Non-Periodic Tiling of the Plane I, II"*, Kon. Nederl. Akad. Wetensch. Proc Ser A 84 (1) pp 39-66, 1981.

[3] Penrose, R. *"Pentaplexity"*, Math Intelligencer V2 #1 pp 32-37, 1979.

[4] Gardner, M. *"Mathematical Games"* , Scientific American, Jan. 1977 pp 110-121.

[5] Shneiderman, Ben *"Creating Direct Manipulation User Interfaces"*, notes from ACM Development Seminar Dec 5, 1985.

[6] Sun Microsystems, *"Programmers Reference Manual for Sunwindows"*, May 1985.

Appendix

Following are the formulas used in the tiling algorithm:

θ_j is the angle through which grid number j has been rotated:

$$\theta_j = j \ (\text{OV } \pi)$$

$$\overline{\text{NG}}$$

∂_j is the amount by which grid number j has been translated.

The intersection point (x,y) of line number m in grid number r, with line number n in grid number s, is:

$$x = \frac{(m - \partial_r)\sin\theta_s - (n-\partial_s)\sin\theta_r}{\sin(\theta_s-\theta_r)}$$

$$y = \frac{-(m-\partial_r)\cos\theta_s + (n-\partial_s)\cos\theta_s}{\sin(\theta_s-\theta_r)}$$

The four vertices of the rhombus derive from this intersection point, in order are:

(u,v), $(u+\cos\theta_r, v+\sin\theta_r)$, $(u+\cos\theta_r+\cos\theta_s, v+\sin\theta_r+\sin\theta_s)$ and $(u+\cos\theta_s, v+\sin\theta_s)$

where u and v are determined as follows:

The integers $k_1,k_2,\ldots,k_N$ (N = NG), are first defined by:

$k_1 = m$

$k_2 = n$

for $j \neq r,s$, $k_j = 1 + \text{INT}(\partial_j + x\cos\theta_j + y\sin\theta_j)$

where "INT(j)" denotes the integer part of "j" in the usual mathematical sense. Finally:

$u = k_1\cos\theta_1 + k_2\cos\theta_2 + \ldots + k_N\cos\theta_N$

$v = k_1\sin\theta_1 + k_2\sin\theta_2 + \ldots + k_N\sin\theta_N$

The following C code fragments, derived from Seymor Haber's program, illustrate the tiling portion of TileTool:

```
double Theta[MAXNG], Gamma[MAXNG], Cs[MAXNG], Sn[MAXNG], K[MAXNG];
int NG; /* Number of Grids */
int LR; /* Line Range */

for (j = 1; j <= NG; j++)                /* setup the theta rotations */
     Thta[j] = Ov * j * Pi/NG;

for (j = 2; j <= NG; j++) {   /*  create  the  gamma's,  the  random
translations */
     temp = random();
     Gamma[j] = temp;
     runsum += temp;
}
Gamma[1] = -runsum;             /* so sum of all gammas is 0 */
for (j = 1; j <= NG; j++) {   /* create sin and cos tables from thetas */
     Cs[j] = cos(Thta[j]);
     Sn[j] = sin(Thta[j]);
}
for (r = 1; r < NG; r++) {          /* create the rhombuses */
     for (s = r+1; s <= NG; s++) {
          delta = (Sn[s] * Cs[r]) - (Cs[s] * Sn[r]);
          for (m = -LR; m <= LR; m++) {
               for (n = -LR; n <= LR; n++) {
                    mt = m - Gamma[r];
                    nt = n - Gamma[s];
                    x = ((mt * Sn[s]) - (nt * Sn[r]))/dlta;
                    y = -((mt * Cs[s] - (nt * Cs[r]))/dlta;
                    for (j = 1; j <= NG; j++) {
                         if (j == r)
                             K[j] = m;
                         else {
                              if  (j == s)
                                   K[j] = n;
                              else {
                                   tk = Gamma[j] + x * Cs[j] + y * Sn[j];
                                    K[j] = INT(tk);
                              }
                         }
                    }
                    /* x1..x4, y1..y4 are the 4 rhombus verticies */
                    x1 = y1 = 0.0;       /* The starting rhombus vertex */
                    for (j = 1; j <= NG; j++) {
                         x1 += (K[j] * Cs[j]);
                         y1 += (K[j] * Sn[j]);
                    }
                    x2 = x1 + Cs[r];     y2 = y1 + Sn[r];
                    x3 = x2 + Cs[s];     y3 = y2 + Sn[s];
                    x4 = x3 - Cs[r];     y4 = y3 - Sn[r];
               }
          }
     }
}
```

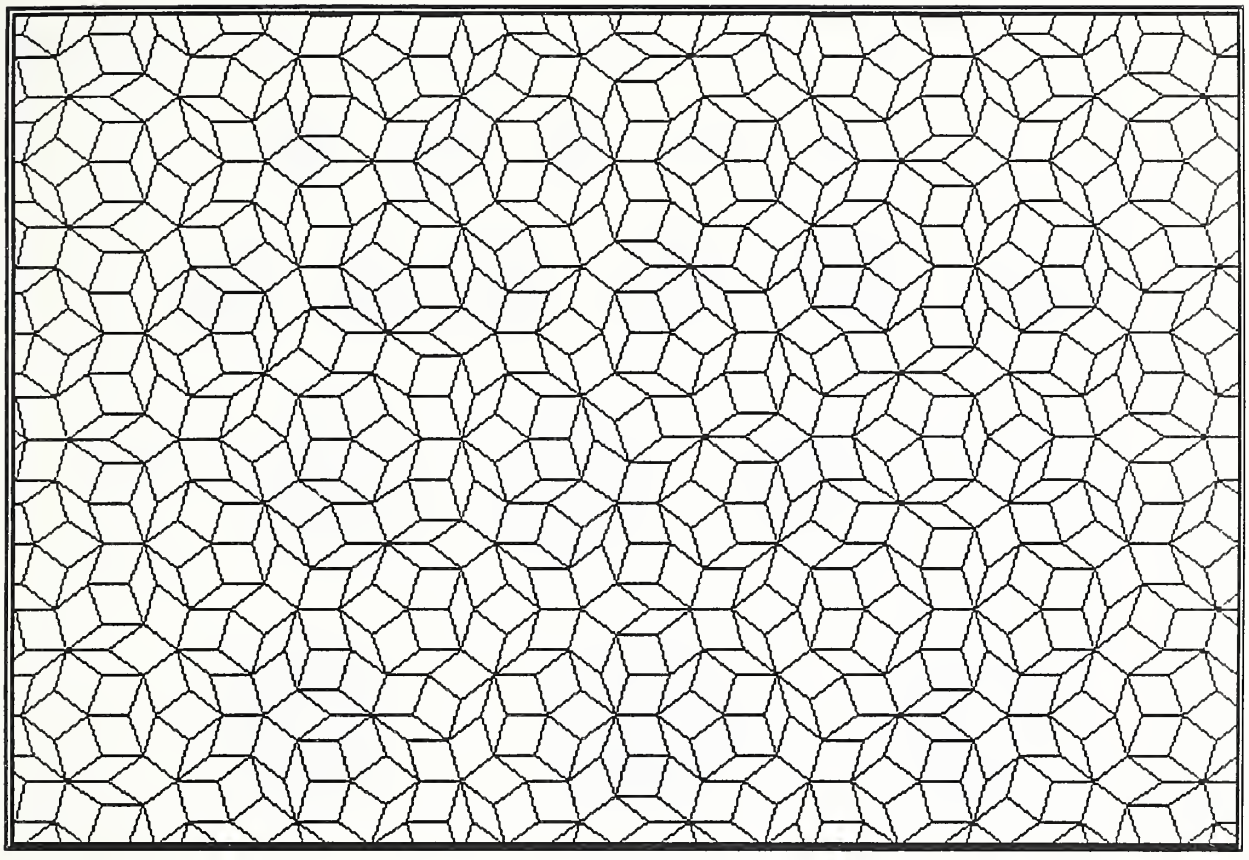


Figure 1

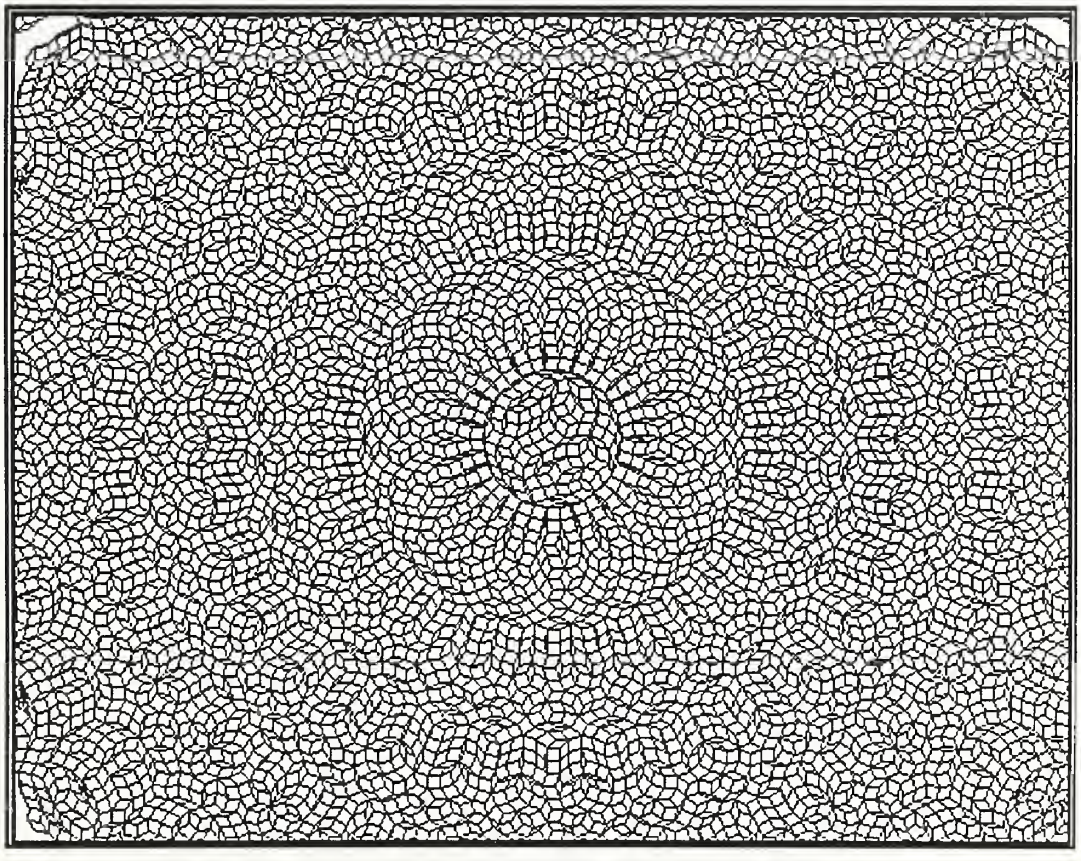


Figure 2

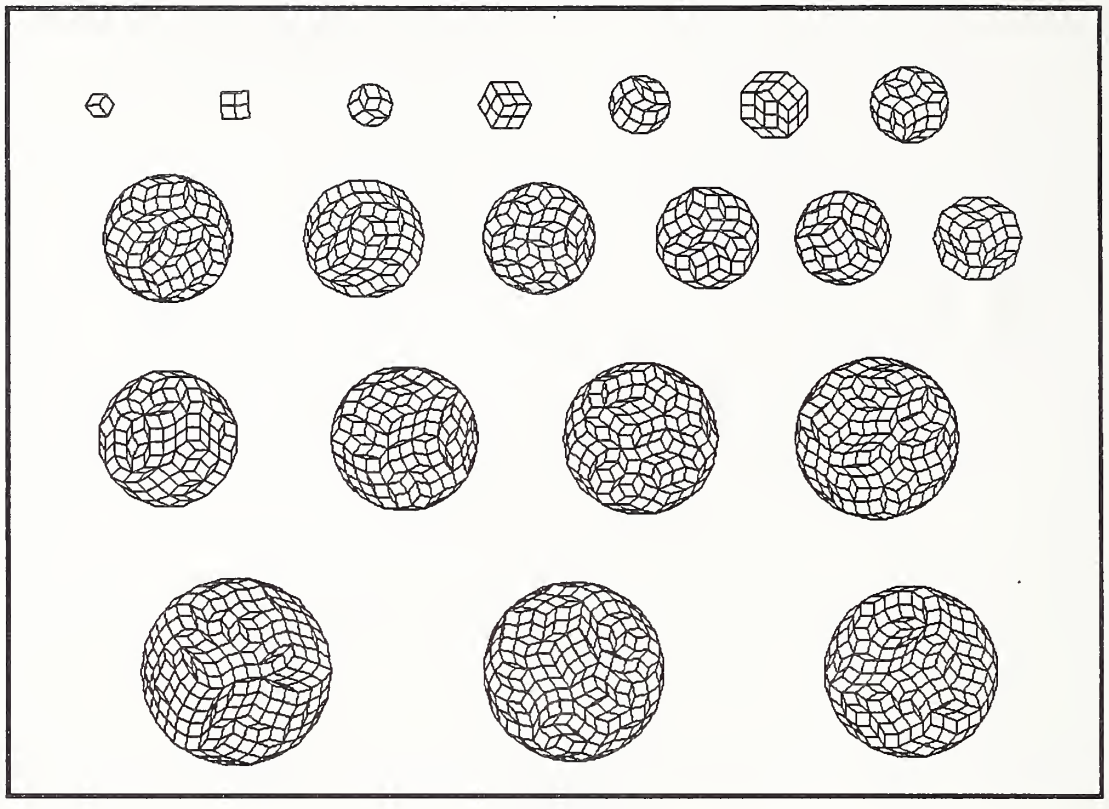


Figure 3

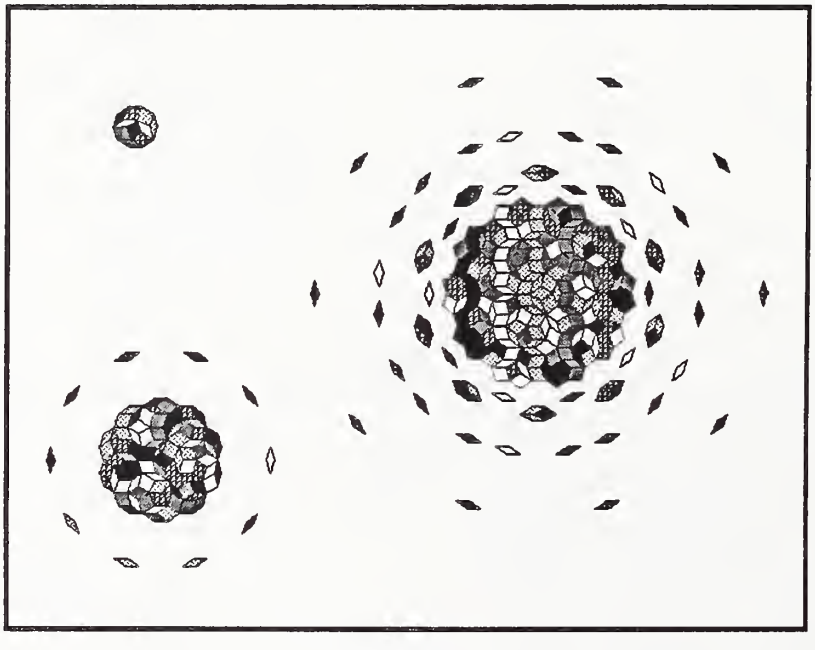
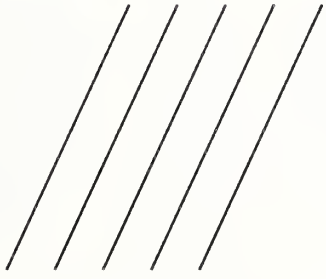
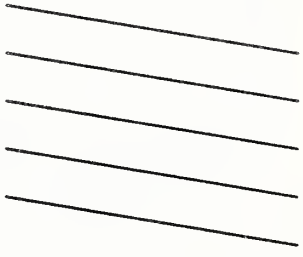


Figure 4

A grid with
line range = 2

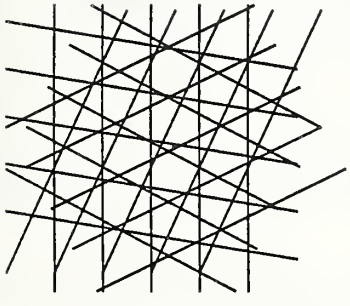
Figure 5a





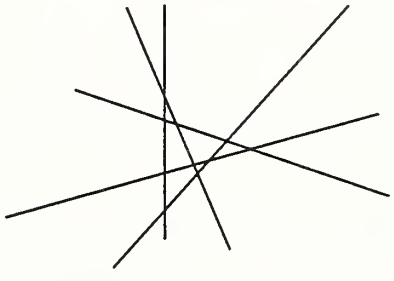
Five grids after rotations
Each grid rotated by (Ovalness *PI) / Number of Grids

Figure 5b



Meshwork formed
by overlaying 5 grids.

Figure 5c



Detail of meshwork.
Note that there are never more
than two lines intersecting any
individual point.

Figure 5d

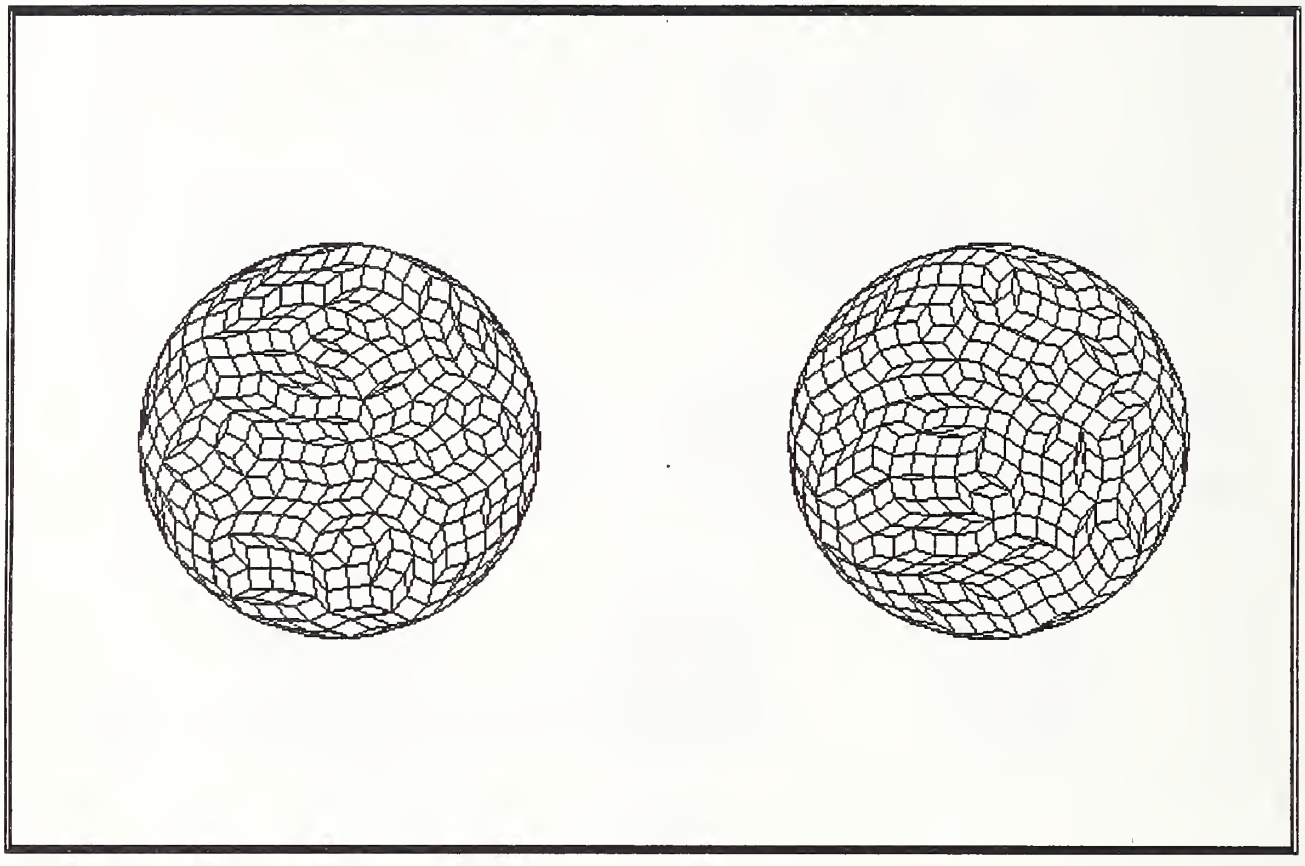


Figure 6

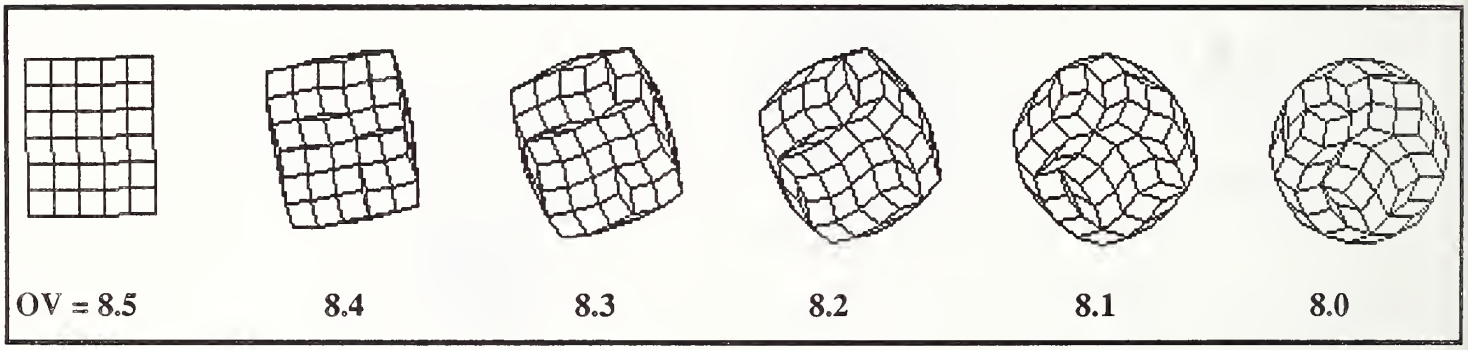


Figure 7

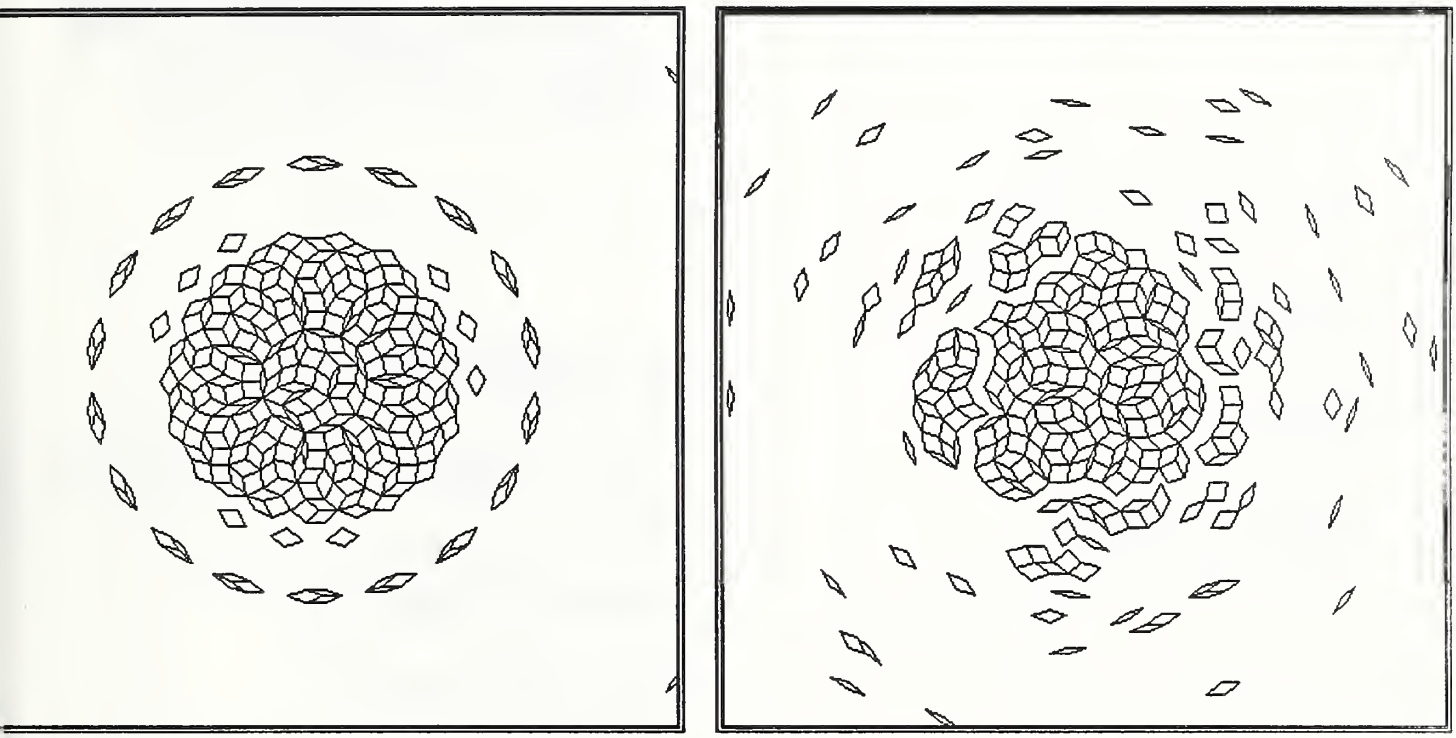


Figure 8

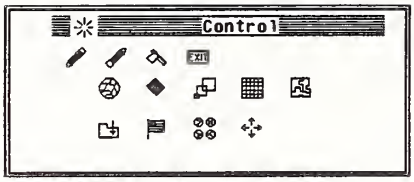


Figure 9

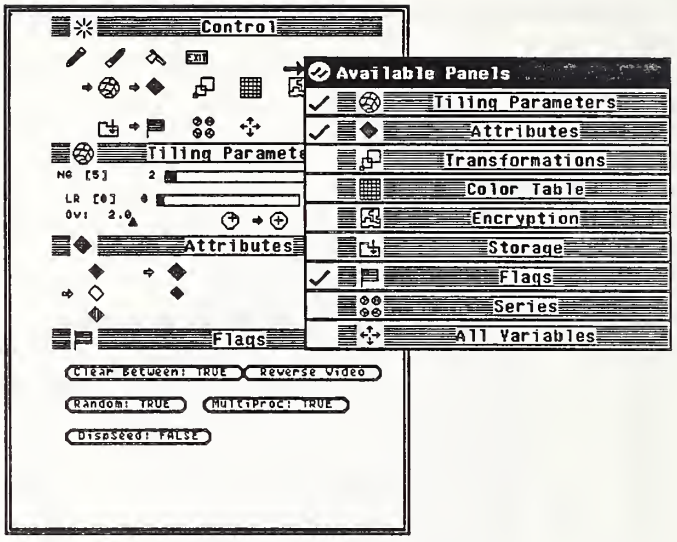


Figure 10

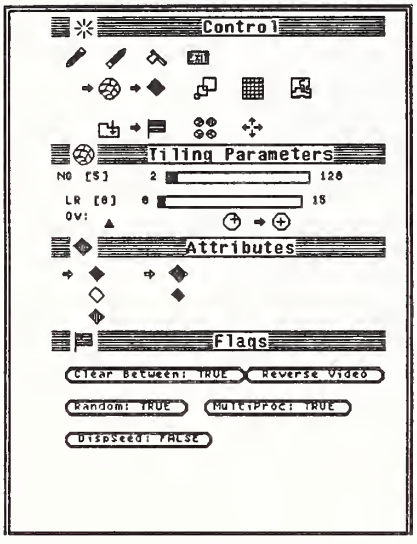


Figure 11

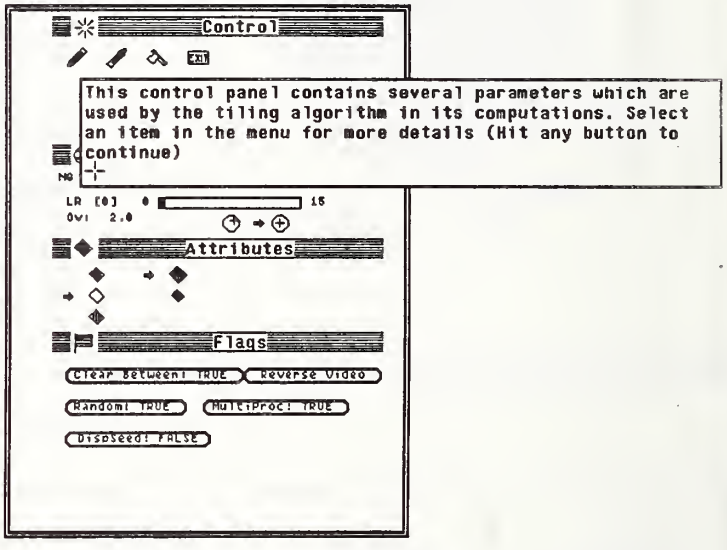


Figure 12

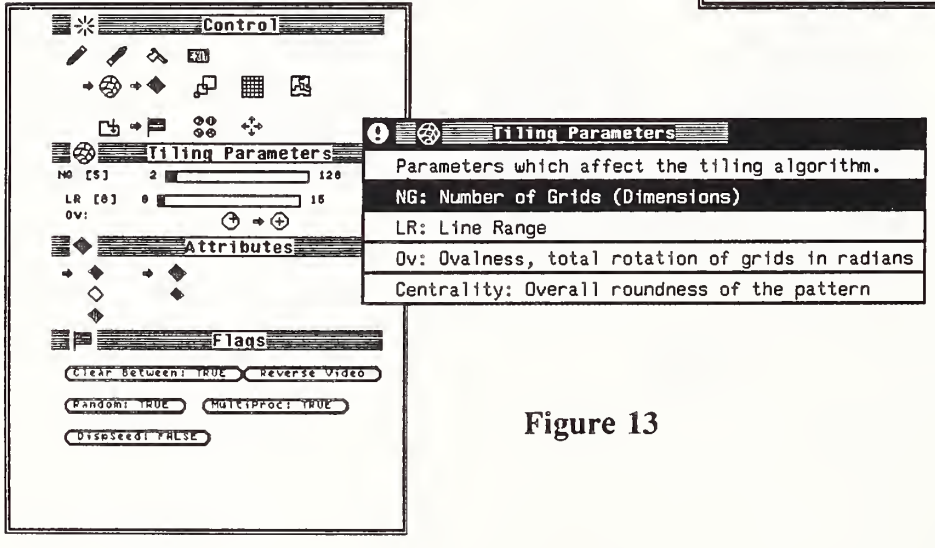


Figure 13

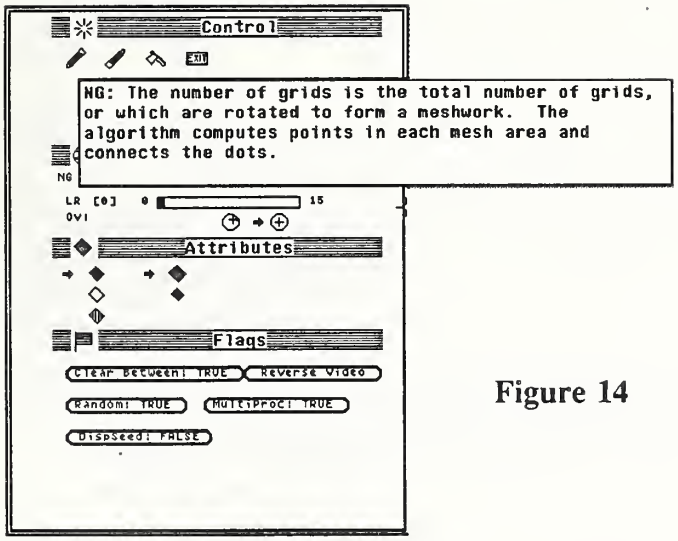


Figure 14

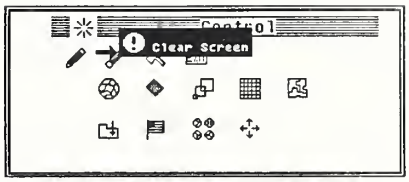


Figure 15

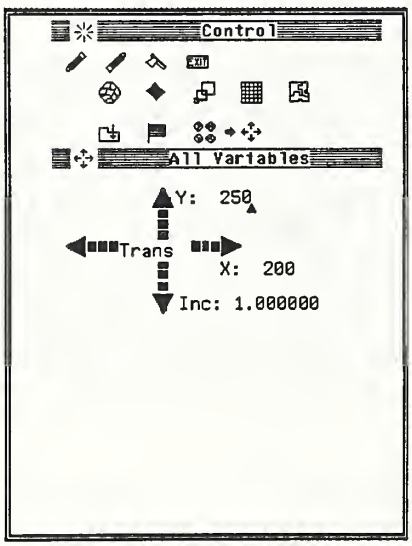


Figure 16

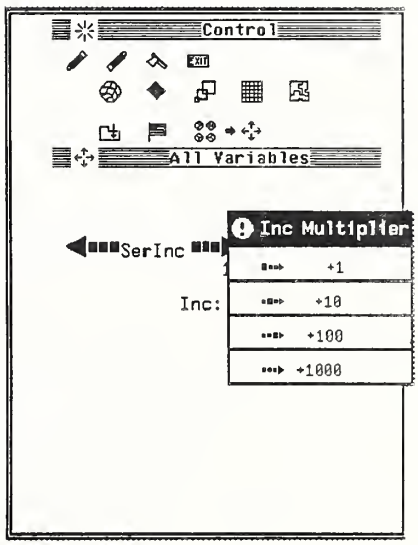


Figure 17

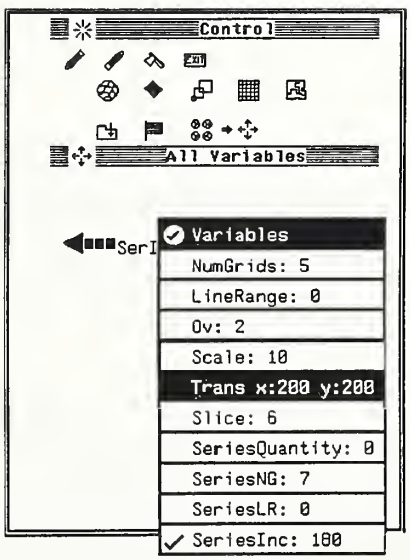


Figure 18

NBS-114A (REV. 2-8C)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET (See instructions)	1. PUBLICATION OR REPORT NO. NBSIR 86-3488	2. Performing Organ. Report No.	3. Publication Date NOVEMBER 1986

4. TITLE AND SUBTITLE

TileTool: A Graphical Interface for the Exploration of Generalized Penrose Tilings

5. AUTHOR(S)

Sanford Ressler 734 x2461

6. PERFORMING ORGANIZATION (If joint or other than NBS, see instructions)

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20234

7. Contract/Grant No.

8. Type of Report & Period Covered

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (Street, City, State, ZIP)

Bureau of Engraving and Printing
Office of Research and Technical Services
Washington, D.C.

10. SUPPLEMENTARY NOTES

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)

A graphical system to experiment with non-periodic tilings of a plane has been developed. Non-periodic tilings have several properties which may be useful in the domain of security printing. A large variety of visually interesting patterns which conform to the constraint of non-periodicity may be rapidly produced with this system. The interaction has been made usable by organizing a set of flexible, consistent, and redundant mechanisms for the selection and modification of the various parameters.

12. KEY WORDS (Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)

computer graphics; user interface; penrose tilings; non-periodic tilings; graphic input

13. AVAILABILITY

☒ Unlimited

☐ For Official Distribution. Do Not Release to NTIS

☐ Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.

☒ Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14. NO. OF PRINTED PAGES

17

15. Price

$9.95