Graphic Tools for Tessellation

Dr. Hari Ramakrishna
Professor Department of Computer Science and Engineering

Abstract—an extensive study in the generation of various title and wallpaper patterns is presented in this paper. Monohydral tiling, dihydral tiling, on-periodic tiling and Penrose titling are discussed. Various wallpaper and frieze patterns can be generated Using this software package developed. The tiles can also be warped into new shapes and formed into tessellations. Various types of warping models are suggested to mold the shape of tiles. A mouse driven interface is developed in order to create and interactive environment. The fractal generating methods are also suggested to create various patterns.

***Index Terms*—Tessellation, wallpaper patterns, Polygon operations, Object oriented Graphic tools, Graphic operations, ngon, kaleidoscope image generation**.

I. Introduction

One of the distinctive features of computers is the ability to perform repetitive tasks. Not Only can they reproduce an earlier computation exactly but can also perform the same task over and over with desired accuracy. This ability can be exploited in the generation of various architectural tiling and wallpaper patterns (1-6).

Symmetry in patterns can be utilized in several applications such as floorings, wallpapers, rugs, and jewelry and building facades to create aesthetically pleasing design. It can be noted that the works of art and architecture from all cultures and ages contain symmetrical and repetitive patterns.

A tile can be represented as a polygon. A Polygon with n sides is known as an ***ngon***. These ngons can be arranged to create floor tile and wallpaper patterns. A tile can be visualized initially as an ngon with n number of sides. These tiles can be edited using scaling, rotating and warping to mold them into desired shapes. A ***kaleidoscope*** image generation method is simulated to create various wallpaper patterns. This can be obtained by fractal generation methods.

A graphics system was developed to achieve the above objectives. An ngon can be reflected either about its own or about another ngon's vertices and lines in order to get a pattern. These patterns will form a set of polygons referred as polylist in this article.

This polylist also can be reflected about another ngon's sides or vertices to develop a wallpaper pattern. Moving the selected tile to the vertices of a pattern can create various tessellations. The shape of the tile and the selected pattern determines the type of tessellation such as monohybrid, dihedral and Penrose tiling. Various warping models are also suggested to mold the tile into various required shapes to generate the desired tessellation.

A mouse driven interface for tile editing and tessellation was also developed to make the software user friendly. This package can be applied to generate groups and patterns for friezes and wallpaper. C++ interface can also be adopted to extend the package.

Required data structure, Graphic operations and a simple and light weight graphic framework is discussed and implementation issues are presented with results.

II. DESIGN AND IMPLEMENTATION ISSUES

The implementation of the above system consists of database design, standard analytical geometrical library function, two-dimensional graphics library functions, polygon operations and poly list (pattern) operations and various other user routines for developing application programs. The package can be divided into six layers as indicated below.

[1] The first layer consists of geometrical routines for various operations such as noting the image of a point on any line, reflection of a point about any line, and intersection of a line segment with any line.

[2] The second layer consists of two-dimensional library functions for performing graphical operations.

[3] The third layer consists of routine for the creation of ngons and operations on ngons.

[4] The fourth layer comprises the creation of various patterns and operations on these patterns.

[5] The fifth layer consists of various library functions for the generation of arbitrary elements and patterns.

[6] The sixth layer consists of mouse driven user interface.

II. DATA STRUCTURES FOR TESSILATION REQUIREMENT

For the implementation of this software system, a suitable database for polygons and wallpaper patterns has to be designed. Microsoft Foundation classes and application framework or any similar middleware development frameworks can be utilized for storing graphic data , for graphic primitive operations and for providing effective user interfaces .

For storing polygon frames a structure known as POINTLIST was defined. This is an ordered list of points. A linked list containing two fields, namely POINT structure for storing a point and a link to next record was defined as POINTLIST.

A polygon has various attributes such as color, width, height, reference point and frame. A wallpaper group may be treated as an ordered list of polygons. A POLY (polygon) structure itself can be used in poly list class which implements collection class operations. This poly list can be used to represent a tessellation or a pattern useful for tessellation. The model class structure for the above types is indicated in Table 1.

Table: 1 Basic structure requirements for tessellation

```
Struct POINT
{
     int  x; for storing x-coordinate
     int y; for storing y-coordinate
}

struct POINTLIST
{
   struct POINT p; for storing a vertex of the polygon
   struct POINTLIST*n;  for linking it to the next vertex
}

struct POLY
{
    struct POINTLIST fm; for storing polygon frame
    struct POINT   c;          for storing reference point
    struct POINT  lh;     for width and height of polygon
    int          lc;           for color of polygon
   struch POLY  *n;      for linking it to the next polygon
}
```

The above table presents a basic structure required for polygon methodology handling tessellation operations. The implementation of the same in Microsoft middleware frameworks will be presented in the following Table: 2

The tessellation structure defined can be used in Microsoft visual studio framework of any version. They can be ported for use in any development environment similar to the Microsoft development environment.

Table: 2 Tessellation Structure using MFC

```
class CPointList
{
private:
             CPoint   point;
             CPointList          *p;
             CPointList          *n;
public:
// Basic class initialization operations
             CPointList();
             CPointList(int i,int j);
             CPointList(CPoint cp);
             ~CPointList();
//Data Management Operations  used to Edit the Data
             void       ClearPointList();
             CPointList*        Copy();
             int          NoOfPoints();
             CPoint                GetPoint();
              void       SetPoint(CPoint cp);
             void       AddPoint(CPoint);

// Serialization Persistence operations
             void       Serialize(CArchive& ar);

// Drawing Display Operations   used in CView
             void       Draw(CDC *dc);

// Graphic Transformations Operations
             void       Move(CPoint p);
             void       Rotate(CPoint p,int an);
             void       Reflect(CPoint);
             void   Scale(CPoint p,float sx=1.0,float sy=1.0);

// Linked list PointList Operations
             BOOLEAN                    IsEmpty();
             BOOLEAN                    IsOne();
             BOOLEAN                    IsLast();
             BOOLEAN                    IsFirst();
             CPointList*                Head();
             CPointList*                Tail();
             CPointList*                Next();
             CPointList*                Previous();
};

class CPoly
{
public:
             CPointList          m_pList;
             int m_Sides;
             int m_Length;
             CPoly(CPoint point,int s,int l);
             void CreatePoly(CPoint point,int s,int l);
             CPoly();
             ~CPoly();
             void       Show(CDC*) ;
};
```

III. BASIC FUNCTION LIBRARIES FOR TESSILATION

Two- dimensional geometrical and graphics functions are required for performing operations on points and line segments. Two-dimensional transformations such as rotation and scaling are performed with reference to a specified point. The reference point may be altered from the global origin to an arbitrary point during processing. Some of functions and their operations are listed in Table 3.

Table: 3 Tessellation Libraries

1 **Rotate**(POINT *p,intan)
Rotation about the origin
2 **RotatePt Rel(**POINTc,POINT *p,int an)
Rotation about point c
3 **ScalePt** (POINT *p,float sx,float sy)
Scaling about the origin
4 **Scale PtRel**(POINT c,POINT *p,float sx,float sy)
Scaling about point c
5 **RefPtHax**(POINT *p,int ty)
Reflection about line y = ty
6 **RefPtVax**(POINT *p,int tx)
Reflection about line x =tx
7 **RefPtCen**(POINT *p,POINT p1)
Reflection about point p1
8 **RefPtLn**(POINT *p,POINT P1,P2)
Reflection about line p1p2
9 **FootPtLn**(POINT *,POINT p1,p2)
Locating the foot of point p on Line segment p1p2
10 **intXPtLnLn**(POINT *p,POINT p1,p2,p3,p4)
Locating the point of intersection of line segment p1p2 with linep3p4

Function 10 of Table 1 used for polygon warping; the routine also returns the nature of point of intersection. A returned value of –1 indicates that the line segments are parallel, +1 indicates that p1 is the point of intersection, 2 indicates that p2 is the point of intersection, 0 indicates that the point of intersection p is on the segment, and -2 indicates that the point of intersection is out side line segment p1p2 but on line p1p2.

A set of library functions are developed for the generation of polygons in order to store the tile models. Performing various classified operations on polygon such as translation, scaling, rotation and reflection can generate tiles of desired shapes.

The most interesting operation on polygon in this particular context is warping. Warping has both magnitude and direction; it is classified into isometric and Penrose warping. Isometric warping is used for generation of dihedral tiling, and Penrose tiling [Fig: 3, 4].

IV. LIBRARY FOR WALLPAPER PATTERN

A Polygon can be reflected about its own sides and vertices for the generation of a wallpaper group. This is stored as a poly list; a poly list can be reflected about the sides and vertices of a polygon, which is obtained by scaling the first polygon. This will form another poly list. By repeating this process, a wallpaper pattern can be obtained. The nature of the pattern changes with the shape of the polygon selected [Fig: 2]. the library function developed and their operations are listed in Table 2.

Table: 4 Wallpaper Pattern Libraries

1 RefPolyPolyVx(POLY *p,p1,p2)
Generation of polylist pbyreflecting polygonp1 on the vertices ofpolygon p2

2 RefPolyPolyLn(POLY *p,p1,p2)
Generation ofpolylist pbyreflecting polygonp1on the sides of polygon p2

3 MovePolyPolyVx(POLY *p,p1,p2)
Generation ofpolylist pbymoving polygonp1 to the vertices ofpolygon p2

4 RefPlistPolyVx(POLY *p,p1,p2)
Generationof polylist pbyreflecting polylistp1on the vertices ofpolygon p2

5 RefPlistPolyLn(POLY *p,p1,p2)
Generationof polylist pbyreflecting polylist p1about the sides ofpolygon p2

6 MovePlistPolyLn(POLY *p,p1,p2)
Generation of polylist pbymoving polylistp1 to the vertices of polygon p2

7 MovePolyPlist(POLY *p,p1,p2)
Generation of polylist p by moving polygon p1 onto points of polylist p2

The patterns generated can be modified into the required shapes and sizes through translation, rotation and scaling. Reflecting can generate various wallpaper groups. A polygon on its sides and vertices; the group will change with the nature of the polygon. Although an infinite number of wallpaper patterns are possible, their symmetries fall into only 17 different groups. A few of them are used extensively for tessellations.

V. APPLICATIONS AND OBSERVATIONs

The package has several applications especially in the generation of various flooring and wallpaper patterns. A mouse driven tile editor was developed using these functions. The tile editor provides a prototype mouse driven interface for designing various tiles and generation of tessellations.

Any arbitrary shape of tile can be generated using these library functions. It also provides various possible wallpaper groups needed for tessellation. A tile shape and pattern can also be selected from the library for generating desired tiling patterns. The tile editor also provides various routines for scaling, routines for scaling, rotating and warping the tile in order to modify the tile shape and patterns.

Various routines in C++ were developed for the creation of wallpaper groups, frieze Groups and patterns. The pattern obtained by reflection, rotation, glide reflection along a line on a motif is known as frieze groups. They are classified into seven groups, namely N group (by translating the motif by ‘d” distance repeatedly along x-axis), G group (glide Reflection of pattern), H group (containing half turns), P group (reflection in a perpendicular line), C group (reflection in centerline), HC group (half turn and reflection in centerline) and HG group (half turn and glide reflection).

Some of the tile shapes, tessellations and wallpaper patterns generated to illustrate the capabilities of the model.

A Miscrosoft Visual studio based application and some sample figures are presented in the following figures

CONCLUSIONS

Computers can reproduce a shape exactly again and again; this ability can be exploited to develop various shapes and patterns. Various shapes of flooring tiles can be obtained by performing various operations such as rotation, translation, reflection and scaling on a polygonal shape. A Polygon with n sides is known, as an ngon is this system.

A software package has been developed to generate general tile shapes and patterns. The library functions developed for the package are described in this brief article. The package incorporates a mouse driven interface that allows the selection of tiles and patterns from the library group. The tiles and patterns can also be developed into new shapes and patterns with the help of the tile editor. The capabilities of the package are illustrated through a few typical examples.

Figure 2,3,4,5 presents outputs generated by these graphic libraries to demonstrate the power of computer graphics in simulation through warping and tessellations.

ACKNOWLEDGMENT

The authors wish to thank faculty department of computer science and engineering for their motivation for the work.

REFERENCES

[1] Hari Ramakrishna (1995). Three – dimensional interactive computer graphics package for civil engineering applications, Proceedings of the National Conference on Civil Engineering Materials and Structures, Hyderabad, India.

[2] Hari Ramakrishna (1996). "**Applications of Computer Graphics in flooring and wall paper Patterns**” First National Conference on Computer Aided Structural Analysis and Design, Hyderabad, Jan’ 1996.

[3] Hari Ramakrishna (1996). “**Applications of Computer Graphics in Interior Design**”, Annual proceedings of Institutes of Engineers at Hyderabad, Nov’ 1996.

[4] K.W. CHUNGA,H.S.Y. CHANA,B.N. WANGB “TESSELLATIONS WITH SYMMETRIES OF THE WALLPAPER GROUPS AND THE MODULAR GROUP IN THE HYPERBOLIC 3-SPACE FROM DYNAMICS”, Computers & Graphics,Volume 25, Issue 2, April 2001, Pages 333–341

[5] Hearn, d. and Baker, M.P (1992). Computer graphics, Prentice Hall, second edition.

[6] Newman,W.S and Sproul, R.S (1981),Principles of interactive computer graphics McGraw-Hill International, second edition.

[7] Foley, J.D and VanDam, A (1984), Fundamentals of interactive computer graphics. The systems programming series, Addison-Wesley Publishing Company.

Figure 1: Sample MFC Application

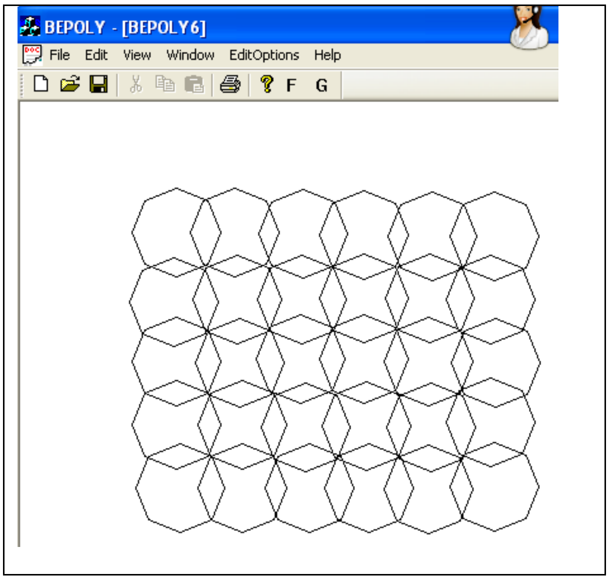

Figure 2: Typical groupings of polygons generated through graphic operations Ref Table 3,4

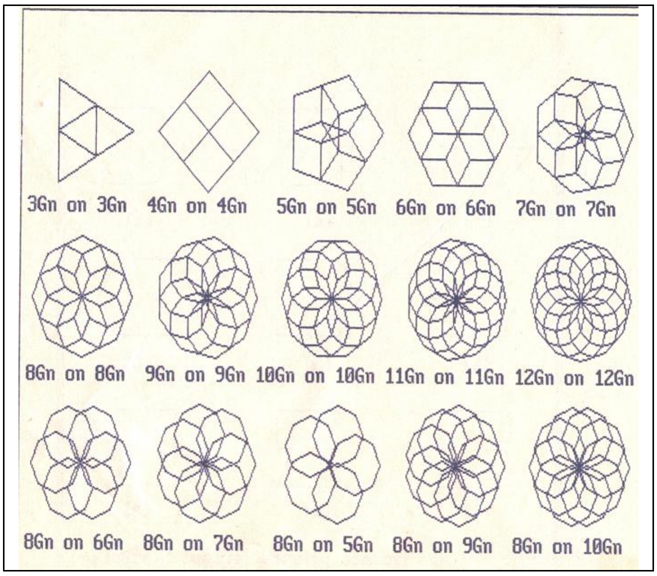

Figure 3: Typical warping shapes (Different warping groups generated Ref 2, 3)

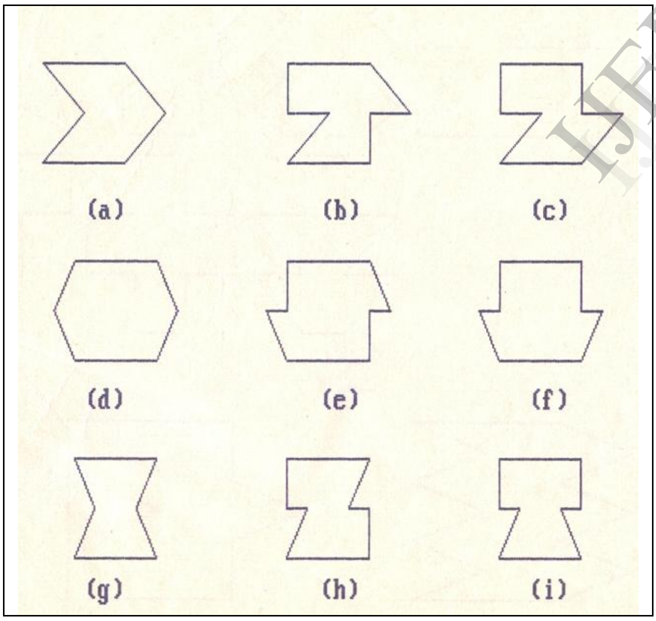

Figure 4: Typical Tile Shapes generated using warping operations shown in figure 3

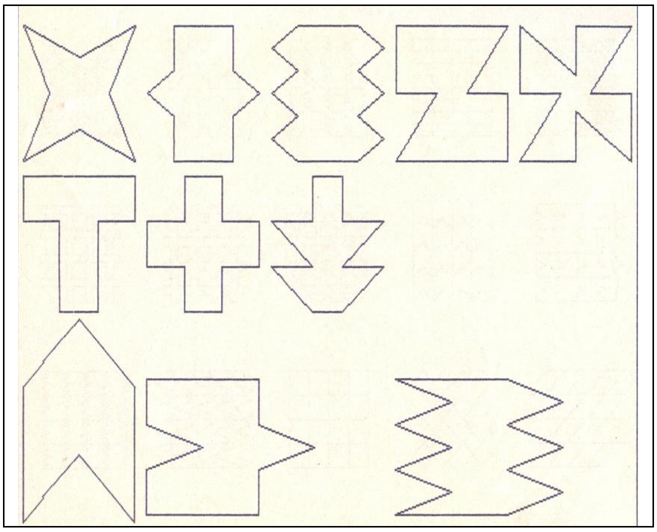

Figure 5: Tessellation Patterns generated using tile shapes of figure 4