Cellular Automaton Graphics(2)

Morio Kikuchi

Abstract :

Developing a regular polyhedron on a plane, setting discrete coordinates on the development and applying a boundary condition of regular polyhedron to it, we realize a symmetrical graphics.

1. Tetrahedron

Figure 1 is a development of tetrahedron. Each regular triangle is the same as the regular triangle of dihedron in the last and its size of a side is n. The number is the number of the vertex, side and the following are combinations of numbers of vertexes, two sides in correspondence.

1, 5 ; 2, 6 ; 7(1-2), 8(5-6) ; 9(3-1), 10(3-5) ; 11(4-2), 12(4-6)

On these correspondences, coordinates of one pixel are got by inversion of coordinates of the other pixel on the side 3-4. The x coordinate does not change and the y coordinate only changes. Assuming the coordinates of vertex 4 to be (x_t, y_t), $x_t = n - 1$ and $y_t = 2(n - 1)$. Assuming the coordinates before inversion to be (x, y), the coordinates after inversion becomes $(x, y_t - y)$.

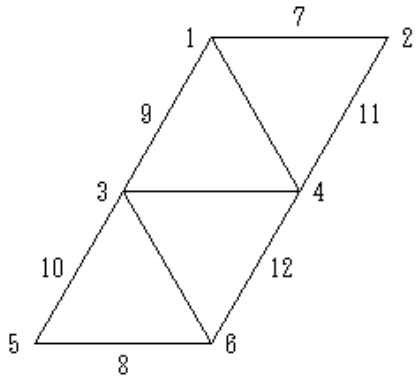

Figure 1

2. Neighborhood view

Figure 2 is neighborhood views on vertex, side. If we try getting a neighborhood view on vertex, connection of regular triangles is needed from tetrahedron. In tetrahedron, there are three regular triangles around a vertex. Therefore, on 2, two regular triangles around 6 are connected and on 5, two regular triangles around 1 are connected and the neighborhood views on them become like vertex 2, vertex 5 in Figure 2. On 1, one regular triangle around 5 is connected and on 6, one regular triangle around 2 is connected, however, even if it is connected, because the target pixel in the regular triangle corresponds with the target pixel in the two regular triangles to which the regular triangle is connected by correspondence of two sides, the connection of regular triangles goes out of use after all. On 3 and 4, because there are three regular triangles around from the beginnning, the connection of regular triangles goes out of use.

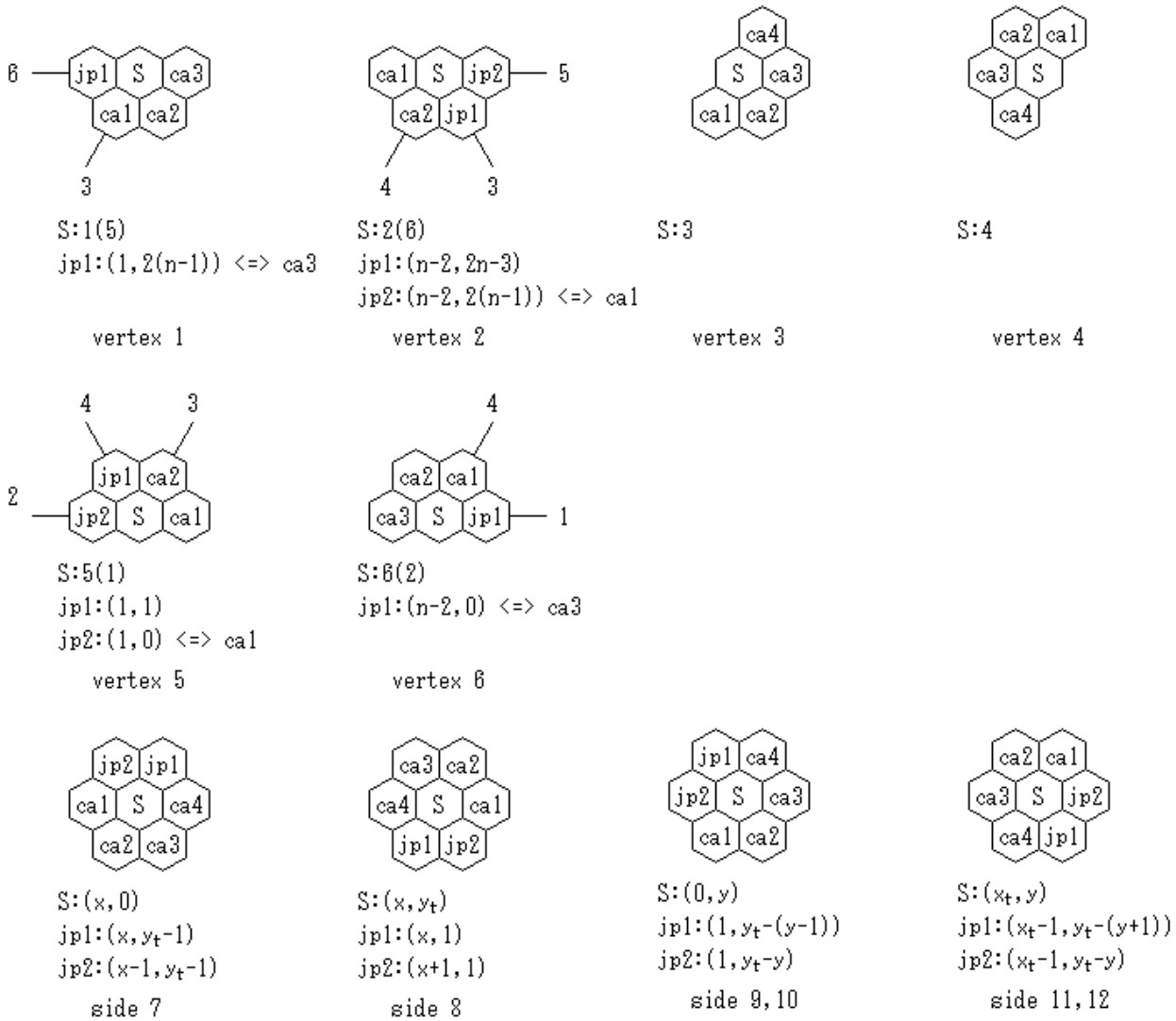

Figure 2

3. Seed point

We consider arrangement of seed points in the case that the number of seed points is three. In Figure 3 left, three seed points are arranged in one regular triangle and in Figure 3 right, one seed point is arranged in one regular triangle. We express them like SP:3, SP:1+1+1 respectively. The black points in each figure are painted beforehand before main painting. The painting of the black point which is the centre of a regular triangle in both the figures is done in the case that $n = 3m + 1$. The arrangement of painting number 2 is done considering the symmetry.

Basically, the number of seed points in one regular triangle is 3 or 1. In dihedron, seed points are arranged like SP:3+3、SP:3 and in tetrahedron, seed points are arranged like SP:3+3+3+3, SP:3+1+1+1, SP:3, SP:1+1+1.

Figure 3

4. Modification of region(the case of dihedron)

We try realizing a symmetrical graphics in a polyhedron which is made from a regular polyhedron replacing a part or all of its figures with other figures. If two regular triangles of dihedron are replaced with the figure(less than tetrahedron) of which development is Figure 4, a polyhedron like Figure 5 is got. We show how to make this polyhedron in Figure 6. First, there are two regular triangles which is marked with R. If the right R is replaced with the replacement figure, the number of regular triangles becomes four. If the left R is replaced with the replacement figure, the number of regular triangles becomes six. We attach numbers to sides in (5) like the following:

・2-1≡9、2-3≡10、5-6≡11、8-7≡12、4-1≡13、4-7≡14、5-8≡15、6-3≡16

The following is the change of correspondence of two sides in Figure 6.

(1)1-3⇔1-4、2-3⇔2-4
(2)1-3′⇔1-4、2-3′⇔2-4、9-3′⇔9-3′
(3)1-3′⇔1-4′、2-3′⇔2-4′、9-3′⇔9-3′、10-4′⇔10-4′
(4)1-3′⇔1-4′、2-3′⇔2-4′、9-1⇔9-1、10-1⇔10-1

In 1-3′⇔1-4′, 2-3′⇔2-4′, assuming coordinates of one pixel to be (x, y), coordinates of the other pixel are (y, x). Assuming that coordinates of the right vertex 1 to be $(x_t, 0)$ and coordinates of the lower vertex 1 to be $(0, y_t)$, $x_t = 2(n-1)$ and $y_t = 2(n-1)$. In 9-1⇔9-1, assuming coordinates of one pixel to be (x, y), coordinates of the other pixel are $(x_t - x, y)$. In 10-1⇔10-1, assuming coordinates of one pixel to be (x, y), coordinates of the other pixel are $(x, y_t - y)$.

In Figure 5, assuming seed points to be SP:1+1+1 in the lower replacement figure, the pixels to be painted beforehand are the lowest point and highest point in Figure 5 and these are 4, 2 in Figure 6-(5) respectively. At 4, 2, the number of regular triangles around a vertex is three and at the other three vertexes(1;3;7, 6;8, 5), it is four.

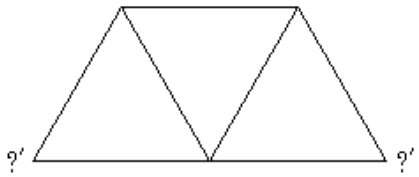

Figure 4

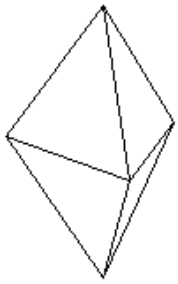

Figure 5

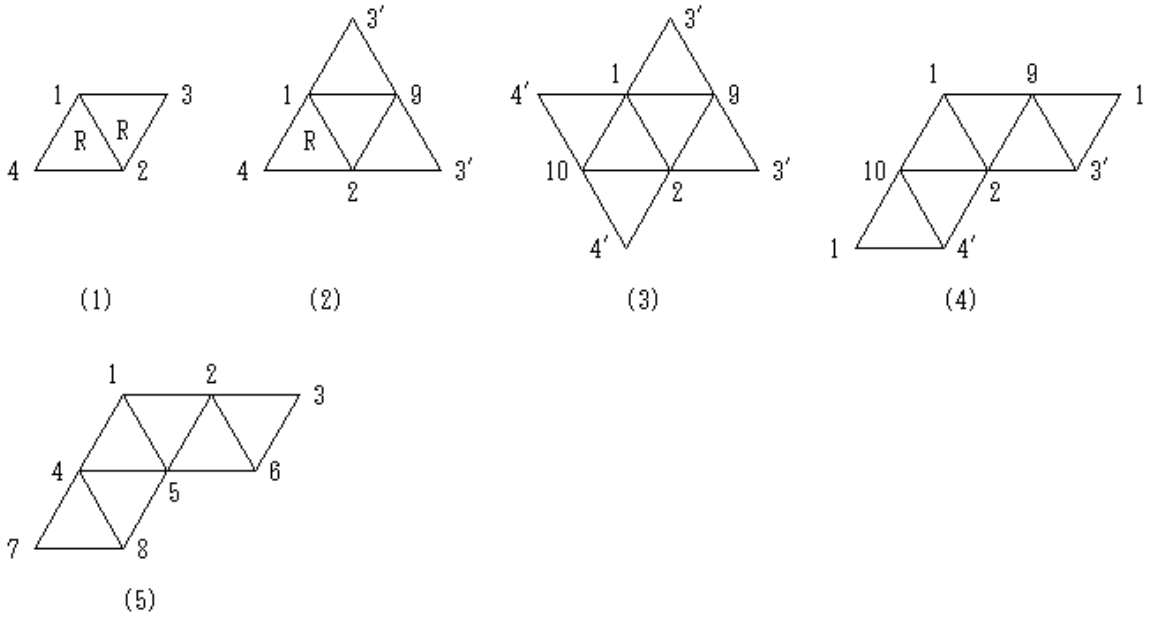

Figure 6

5. Modification of region(the case of tetrahedron)

If four regular triangles of tetrahedron are replaced with the replacement figure, a polyhedron like Figure 7 is got. We show how to make this polyhedron in Figure 8. First, there are four regular triangles which is marked with R. If the Rs are replaced with the replacement figure one by one, the number of regular triangles becomes twelve finally. We attach numbers to sides in (6) like the following:

・2-1≡13、2-3≡14、3-4≡15、9-10≡16、11-10≡17、11-12≡18、5-1≡19、5-9≡20、8-4≡ 21、8-12≡22

The following is the change of correspondence of two sides in Figure 8.

(1)1-2⇔5-6、3-1⇔3-5、4-2⇔4-6
(2)1-2′⇔5-6、3-1⇔3-5、4-2′⇔4-6、13-2′⇔13-2′
(3)1-2′⇔5′-6、3-1⇔3-5′、4-2′⇔4-6、13-2′⇔13-2′、14-5′⇔14-5′
(4)1-2′⇔5′-6′、3-1⇔3-5′、4-2′⇔4-6′、13-2′⇔13-2′、14-5′⇔14-5′、15-6′⇔15-6′
(5)1′-2′⇔5′-6′、13-2′⇔13-2′、14-5′⇔14-5′、15-6′⇔15-6′、16-1′⇔16-1′

Assuming that coordinates of the lower vertex $5'$ to be $(0,\ y_t)$ and coordinates of the right vertex $1'$ to be $(x_{tm},\ 0)$, $y_t = 2(n-1)$ and $x_{tm} = 2(n-1)$. In 13-2′⇔13-2′, 14-5′⇔14-5′, assuming coordinates of one pixel to be $(x,\ y)$, coordinates of the other pixel are $(x,\ y_t - y)$. In 15-6′⇔15-6′, assuming coordinates of one pixel to be $(x,\ y)$, coordinates of the other pixel are $(2x_{tm} - x,\ y)$. In 16-5′⇔16-5′, assuming coordinates of one pixel to be $(x,\ y)$, coordinates of the other pixel are $(x_{tm} - x,\ y)$. In 1′-2′⇔5′-6′, coordinate transformation becomes parallel movement. In 1′-2′⇒5′-6′, $\Delta x = -x_{tm}$ and $\Delta y = y_t$ and in 1′-2′⇐5′-6′, $\Delta x = x_{tm}$ and $\Delta y = -y_t$.

In Figure 7, assuming seed points to be SP:1+1+1 in the lower replacement figure, the pixels to be painted beforehand are the lowest point and highest point in Figure 7 and these are 11, 1;3;9 in Figure 8-(6) respectively. At 2, 5, 8, 11, the number of regular triangles around a vertex is three and at the other four vertexes(1;3;9, 4;10;12, two vertexes on the inside), it is six.

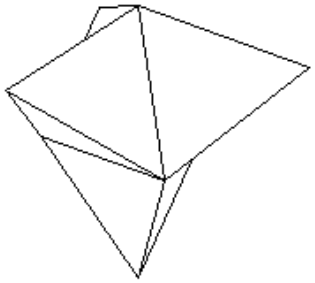

Figure 7

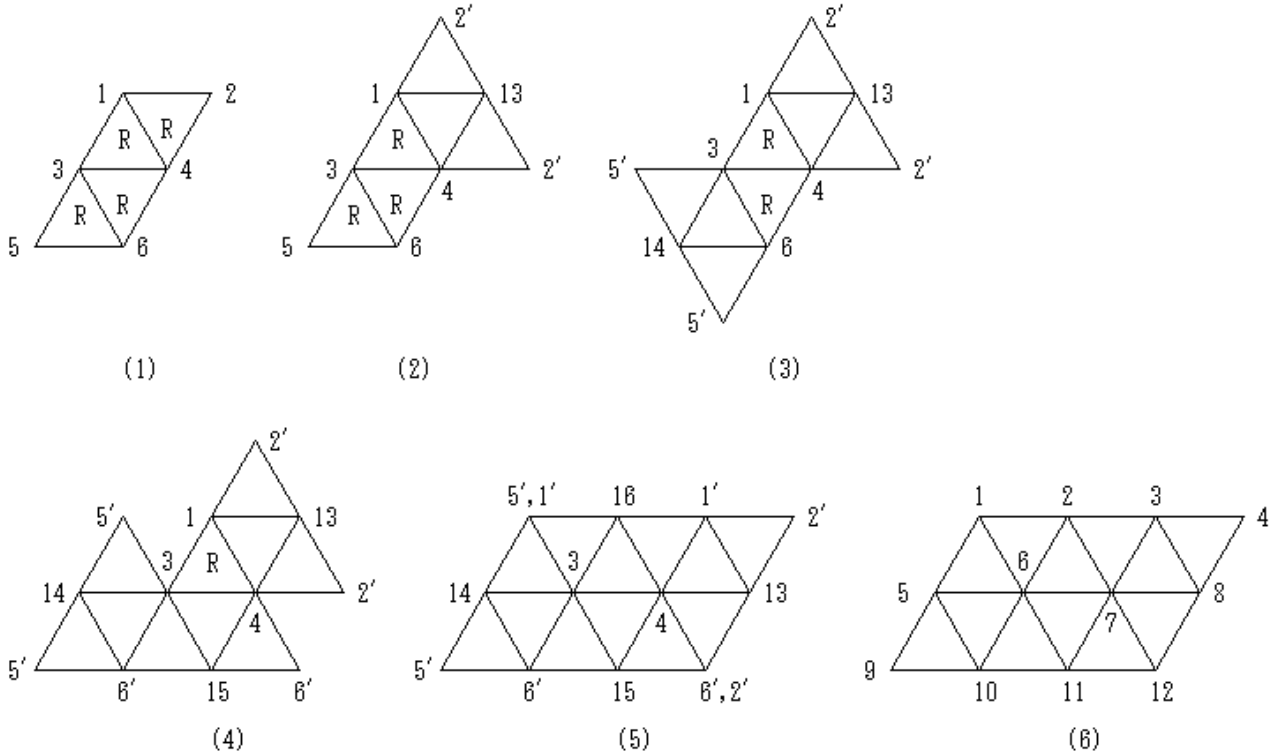

Figure 8

6. Assignment

Try making neighborhood views on vertexes and sides in the case of Figure 6, Figure 8.

7. Concrete example

Figure 9 is a symmetrical graphics by Figure 2 and the following are its data.

・$n = 22$

・coordinates of painting number 1：painting point 'a':(2, $n-1-2$), painting point b:(4, $n-1+2$), painting point c:(2, $n-1+4$)

・coordinates of painting number 2：painting point 'a':$\Delta x = 1$, painting point b:$\Delta x = 1; \Delta y = 1$, painting point c:$\Delta y = 1$

・choice of CW, CCW：the same as the last

・painting algorithm：angle method

・painting timing：the same as the last

・push to stack：the same as the last

The array which is used for painting is initialized like the following:

- target pixel : 15
- vertex$(0,\ n-1)$: -1
- centre$(n-(n/3+1),\ n/3)$ of regular triangle : -1
- wall pixel : 0

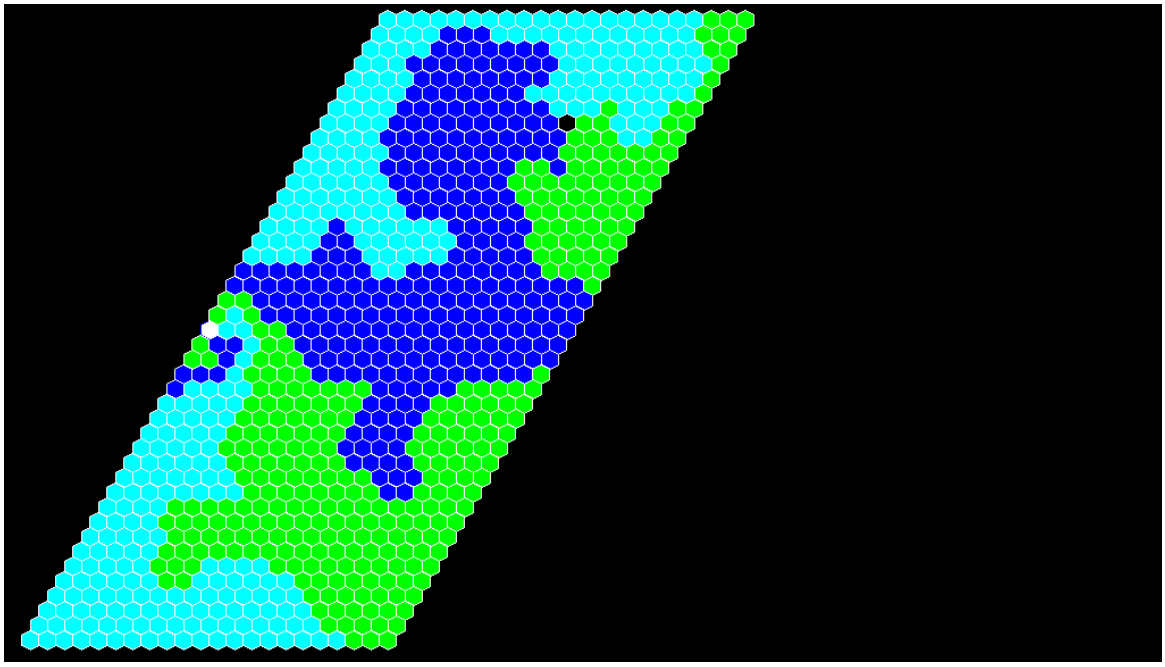

Figure 9

Figure 10 is a symmetrical graphics by Figure 6 and the following are its data.

- vertex $(0,\ n-1)$, $(n-1,\ 0)$: -1

The others are the same as Figure 9.

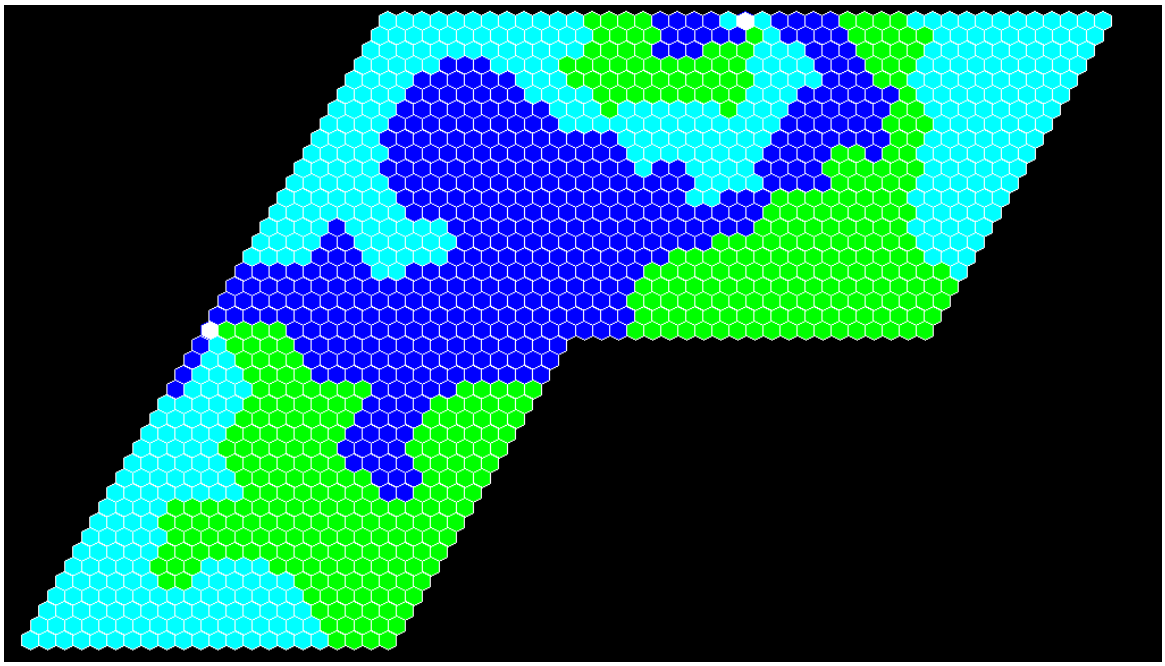

Figure 10

Figure 11 is a symmetrical graphics by Figure 8 and the following are its data.

- vertex $(0,\ n-1)$, $(x_{tm},\ n-1)$: -1

The others are the same as Figure 9.

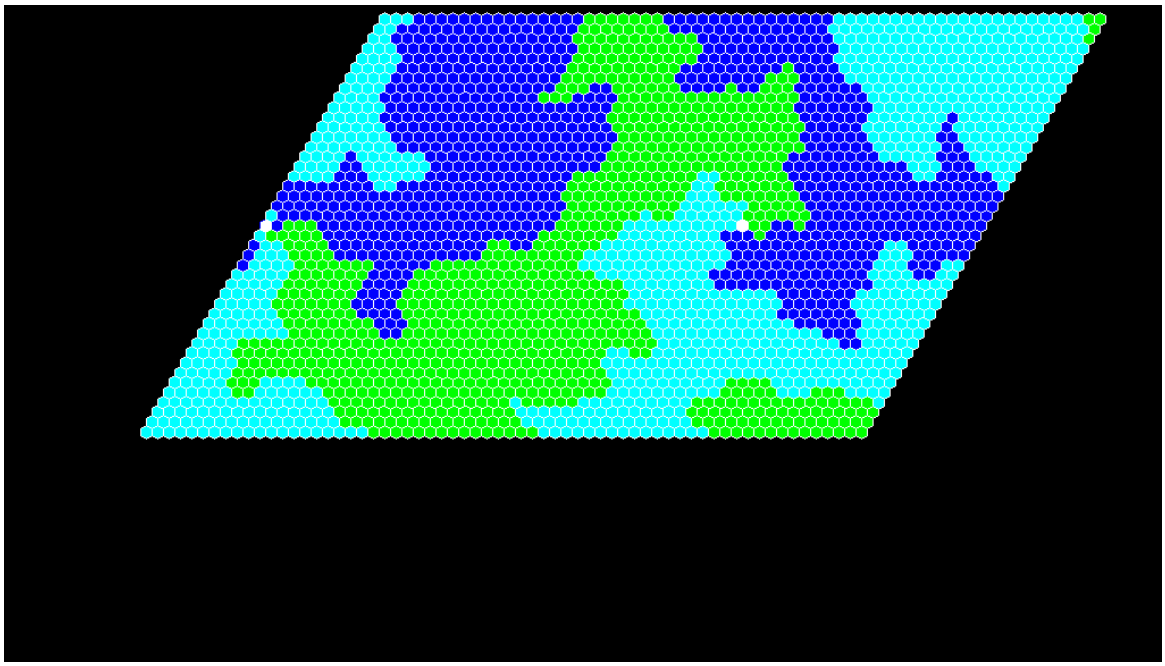

Figure 11

****

セルラーオートマトングラフィクス (2)

菊池盛雄

アブストラクト：
正多面体を平面上に展開し、この展開図形に離散座標を設定し、正多面体の境界条件を適用して対称なグラフィクスを実現します。

1. 正 4 面体

図 1 は正四面体の展開図です。各正三角形は前回の正二面体の正三角形と同じものであり、辺の大きさは n です。数字は頂点、辺の番号であり、以下は対応する頂点、辺の番号の組合せです。

1, 5；2, 6；7(1-2), 8(5-6)；9(3-1), 10(3-5)；11(4-2), 12(4-6)

この対応に関して、一方のピクセルの座標は他方のピクセルの座標の辺 3-4 に関する反転によって得られます。x 座標は変わらず y 座標だけが変わります。頂点 4 の座標を $(x_t,\, y_t)$ とすると $x_t = n - 1$、$y_t = 2(n-1)$ です。反転前の座標を $(x,\, y)$ とすると、反転後の座標は $(x,\, y_t - y)$ となります。

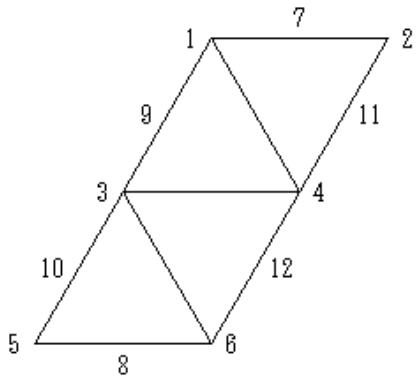

図 1

2. 近傍図

図 2 は頂点、辺に関する近傍図です。頂点に関する近傍図を求める場合、正四面体からは正三角形の連結が必要となります。正四面体では頂点の周りには三つの正三角形があります。したがって、2 では 6 の周りの二つの正三角形を連結し、5 では 1 の周りの二つの正三角形を連結し、図 2 の vertex 2、vertex 5 のようになります。1 では 5 の周りの一つの正三角形を、6 では 2 の周りの一つの正三角形を連結しますが、連結してもこの正三角形のターゲットピクセルは辺の対応により連結される側の二つの正三角形のターゲットピクセルと対応するので、結局正三角形の連結は不要となります。3、4 では最初から周りに三つの正三角形があるので正三角形の連結は不要となります。

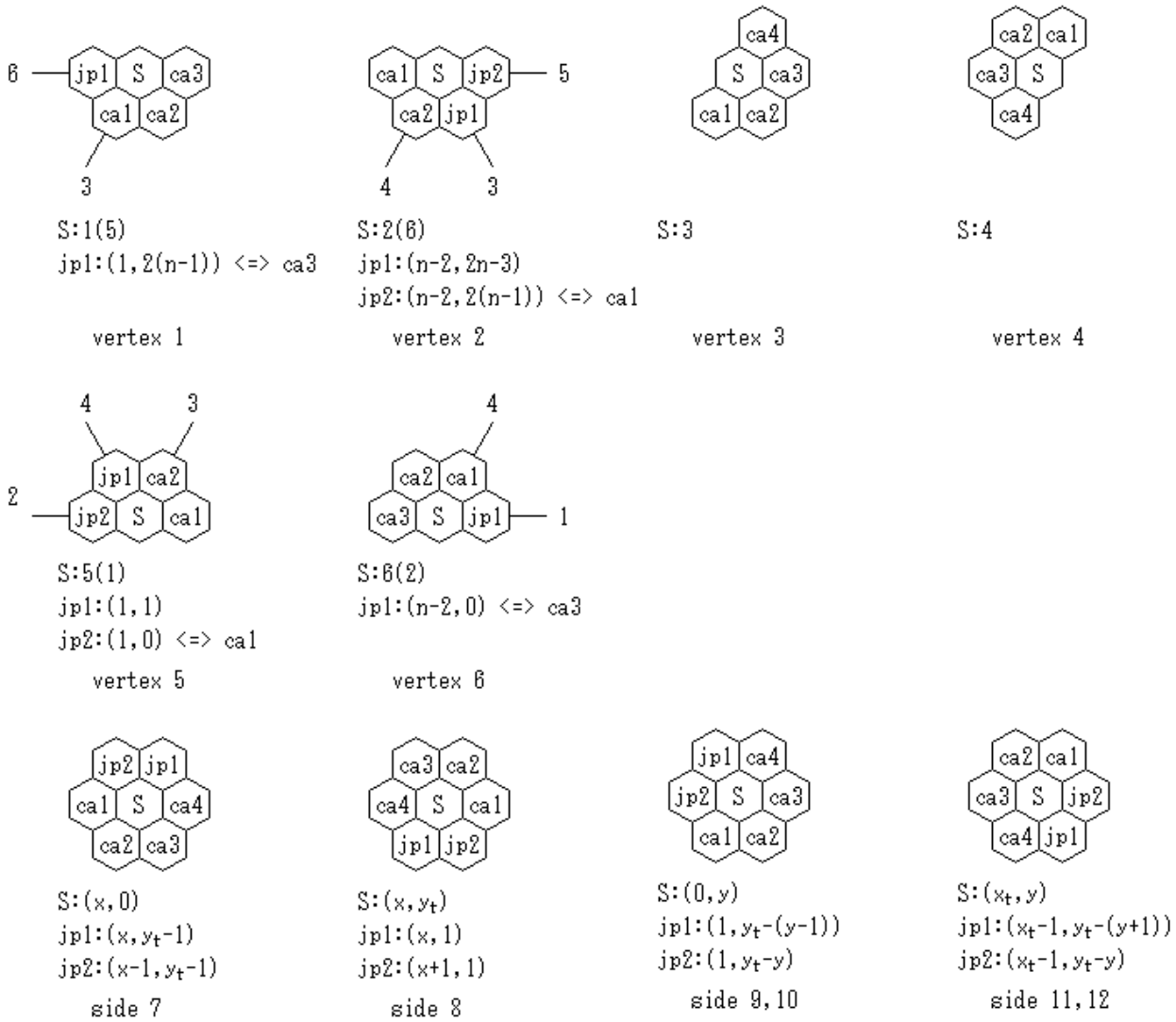

図 2

3. シードポイント

シードポイントの数が 3 の場合のシードポイントの配置を考えます。図 3 左では一つの正三角形の中に三つのシードポイントが、図 3 右では一つの正三角形の中に一つのシードポイントが配置されています。これらの配置を各々、SP:3、SP:1+1+1 と表記します。本塗りつぶしの前に各図で示される黒点をあらかじめ塗りつぶしておきます。両図における正三角形の中心である黒点の塗りつぶしは $n = 3m + 1$ の場合に行います。塗番号 2 の配置は対称性を考慮して行います。

基本的には一つの正三角形におけるシードポイントの数は 3 または 1 です。正二面体では SP:3+3、SP:3、正四面体では SP:3+3+3+3、SP:3+1+1+1、SP:3、SP:1+1+1 のようにシードポイントを配置します。

図 3

4. 領域の加工 (正二面体の場合)

正多面体の図形の一部または全部を別の図形で置き換えた多面体において対称なグラフィクスを実現することを試みます。正二面体の二つの正三角形を展開図が図 4 である図形 (正四面体未満) で置き換えると図 5 のような多面体が得られます。この多面体の作り方を図 6 に示します。最初は正三角形 R が二つあります。右側の R を置換図形で置き換えると正三角形は四つになります。左側の R を置換図形で置き換えると正三角形は六つになります。(5) における辺には以下のように番号を付します。

・2-1≡9、2-3≡10、5-6≡11、8-7≡12、4-1≡13、4-7≡14、5-8≡15、6-3≡16

図 6 における辺の対応の推移を示します。

(1)1-3⇔1-4、2-3⇔2-4
(2)1-3′⇔1-4、2-3′⇔2-4、9-3′⇔9-3′
(3)1-3′⇔1-4′、2-3′⇔2-4′、9-3′⇔9-3′、10-4′⇔10-4′
(4)1-3′⇔1-4′、2-3′⇔2-4′、9-1⇔9-1、10-1⇔10-1

1-3′⇔1-4′、2-3′⇔2-4′ においては、一方のピクセルの座標を $(x,\, y)$ とすると他方のピクセルの座標は $(y,\, x)$ です。右側の頂点 1 の座標を $(x_t,\, 0)$ とし、下側の頂点 1 の座標を $(0,\, y_t)$ とすると $x_t = 2(n-1)$、$y_t = 2(n-1)$ です。9-1⇔9-1 においては、一方のピクセルの座標を $(x,\, y)$ とすると他方のピクセルの座標は $(x_t - x,\, y)$ です。10-1⇔10-1 においては、一方のピクセルの座標を $(x,\, y)$ とすると他方のピクセルの座標は $(x,\, y_t - y)$ です。

図 5 においてシードポイントを下側の置換図形において SP:1+1+1 とすれば、あらかじめ塗りつぶしておくピクセルは図 5 の最下点と最上点であり、これらは各々図 6-(5) の 4、2 です。4、2 では頂点の周りにある正三角形の数は 3 であり、他の 3 頂点 (1;3;7、6;8、5) においては 4 です。

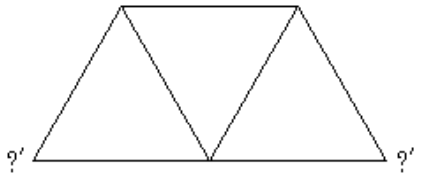

図 4

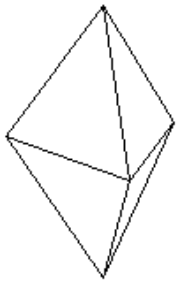

図 5

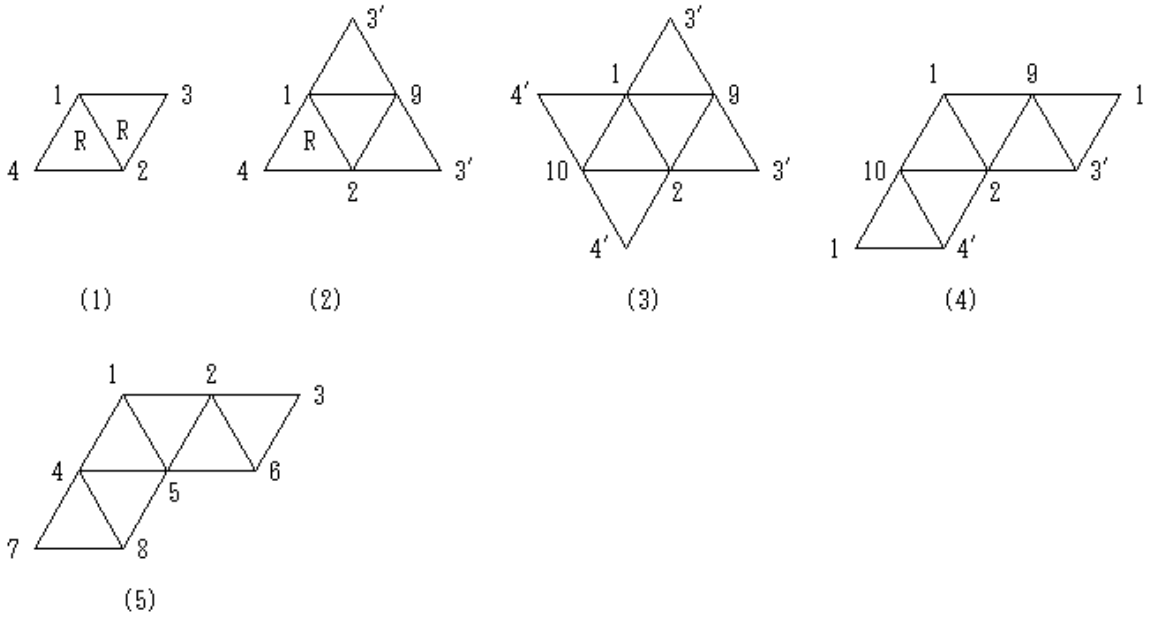

図 6

5. 領域の加工 (正四面体の場合)

正四面体の四つの正三角形を置換図形で置き換えると図 7 のような多面体が得られます。この多面体の作り方を図 8 に示します。最初は正三角形 R が四つあります。R を一つずつ置換図形で置き換えると最終的に正三角形は十二になります。(6) における辺には以下のように番号を付します。

・2-1≡13、2-3≡14、3-4≡15、9-10≡16、11-10≡17、11-12≡18、5-1≡19、5-9≡20、8-4≡ 21、8-12≡22

図 8 における辺の対応の推移を示します。

(1)1-2⇔5-6、3-1⇔3-5、4-2⇔4-6
(2)1-2′⇔5-6、3-1⇔3-5、4-2′⇔4-6、13-2′⇔13-2′
(3)1-2′⇔5′-6、3-1⇔3-5′、4-2′⇔4-6、13-2′⇔13-2′、14-5′⇔14-5′
(4)1-2′⇔5′-6′、3-1⇔3-5′、4-2′⇔4-6′、13-2′⇔13-2′、14-5′⇔14-5′、15-6′⇔15-6′
(5)1′-2′⇔5′-6′、13-2′⇔13-2′、14-5′⇔14-5′、15-6′⇔15-6′、16-1′⇔16-1′

下側の頂点 5′ の座標を $(0,\, y_t)$ とし、右側の頂点 1′ の座標を $(x_{tm},\, 0)$ とすると $y_t = 2(n-1)$、$x_{tm} = 2(n-1)$ です。13-2′⇔13-2′ と 14-5′⇔14-5′ においては、一方のピクセルの座標を $(x,\, y)$ とすると他方のピクセルの座標は $(x,\, y_t - y)$ です。15-6′⇔15-6′ においては、一方のピクセルの座標を $(x,\, y)$ とすると他方のピクセルの座標は $(2x_{tm} - x,\, y)$ です。16-5′⇔16-5′ においては、一方のピクセルの座標を $(x,\, y)$ とすると他方のピクセルの座標は $(x_{tm} - x,\, y)$ です。1′-2′⇔5′-6′ においては座標変換は平行移動になります。1′-2′⇒5′-6′ では $\Delta x = -x_{tm}$、$\Delta y = y_t$ であり、1′-2′⇐5′-6′ では $\Delta x = x_{tm}$、$\Delta y = -y_t$ です。

図 7 においてシードポイントを下側の置換図形において SP:1+1+1 とすれば、あらかじめ塗りつぶしておくピクセルは図 7 の最下点と最上点であり、これらは各々図 8-(6) の 11、1;3;9 です。2、5、8、11 では頂点の周りにある正三角形の数は 3 であり、他の 4 頂点 (1;3;9、4;10;12、内部の 2 頂点) においては 6 です。

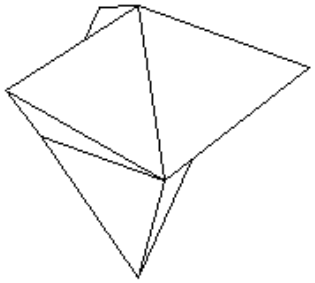

図 7

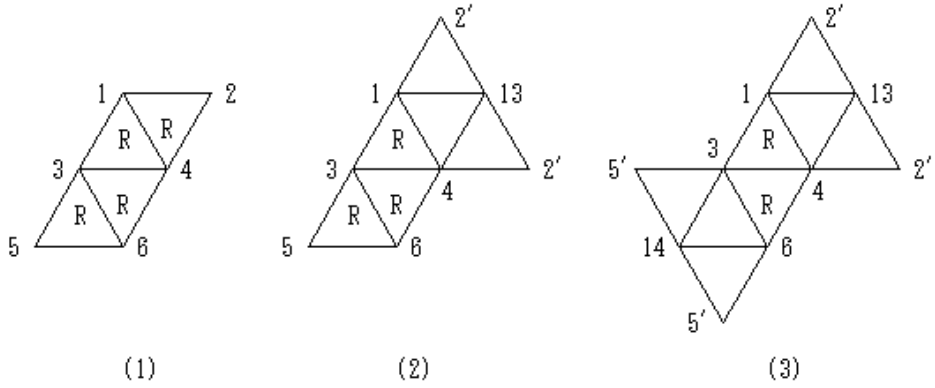

図 8

6. 課題

図 6、図 8 の場合の頂点、辺に関する近傍図を作成してみてください。

7. 具体例

図 9 は図 2 による対称グラフィクスであり、以下はそのデータです。

・$n = 22$
・塗番号 1 の座標：塗点 a:$(2,\ n-1-2)$、塗点 b:$(4,\ n-1+2)$、塗点 c:$(2,\ n-1+4)$
・塗番号 2 の座標：塗点 a:$\Delta x = 1$、塗点 b:$\Delta x = 1; \Delta y = 1$、塗点 c:$\Delta y = 1$
・CW、CCW の選択：前回と同じ
・塗りつぶしアルゴリズム：角度法
・塗り方：前回と同じ
・スタックへの座標のプッシュ：前回と同じ

塗りつぶしに用いられる配列は以下のように初期化します。

・ターゲットピクセル：15
・正三角形の中心 $(n-(n/3+1),\, n/3)$：-1
・頂点 $(0,\, n-1)$：-1
・壁ピクセル：0

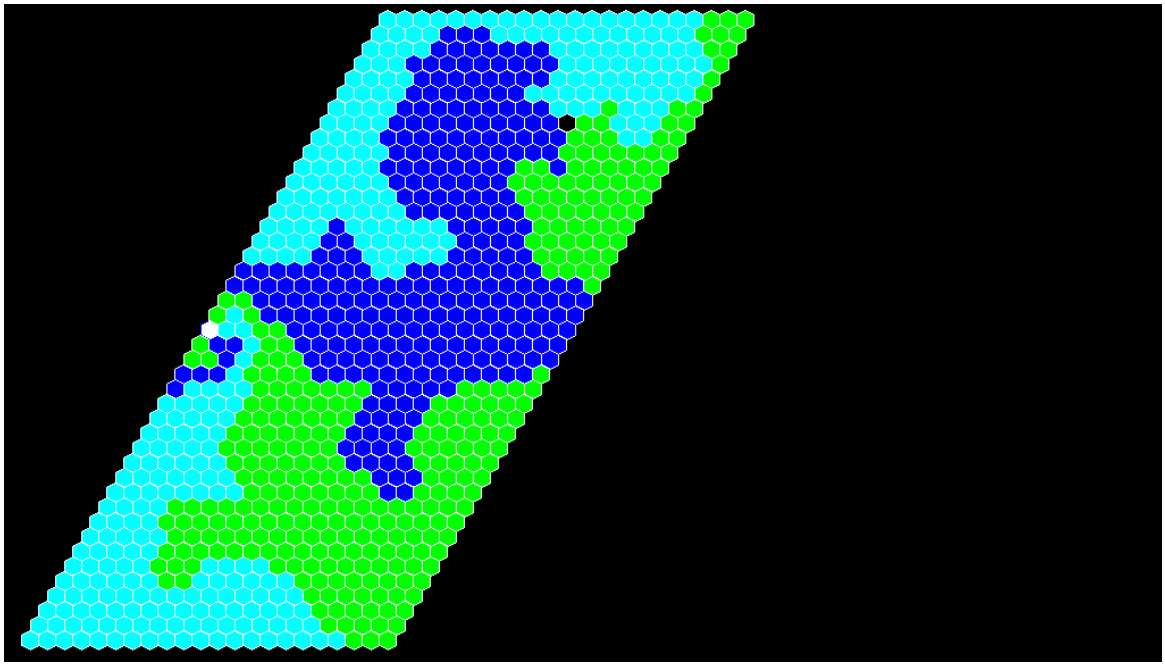

図 9

図 10 は図 6 による対称グラフィクスであり、以下はそのデータです。

・ターゲットピクセル：15
・頂点 $(0,\, n-1)$、$(n-1,\, 0)$：-1
・壁ピクセル：0

他は図 9 と同じです

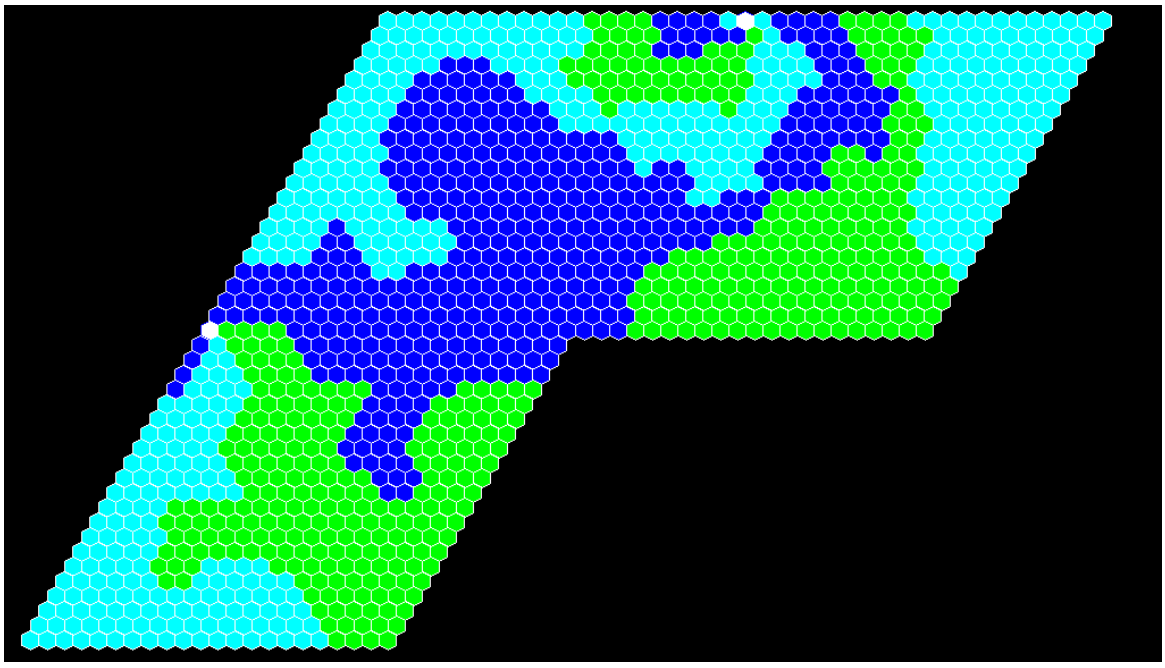

図 10

図 11 は図 8 による対称グラフィクスであり、以下はそのデータです。

・ターゲットピクセル：15
・頂点 $(0,\, n-1)$、$(x_{tm},\, n-1)$：-1
・壁ピクセル：0

他は図 9 と同じです

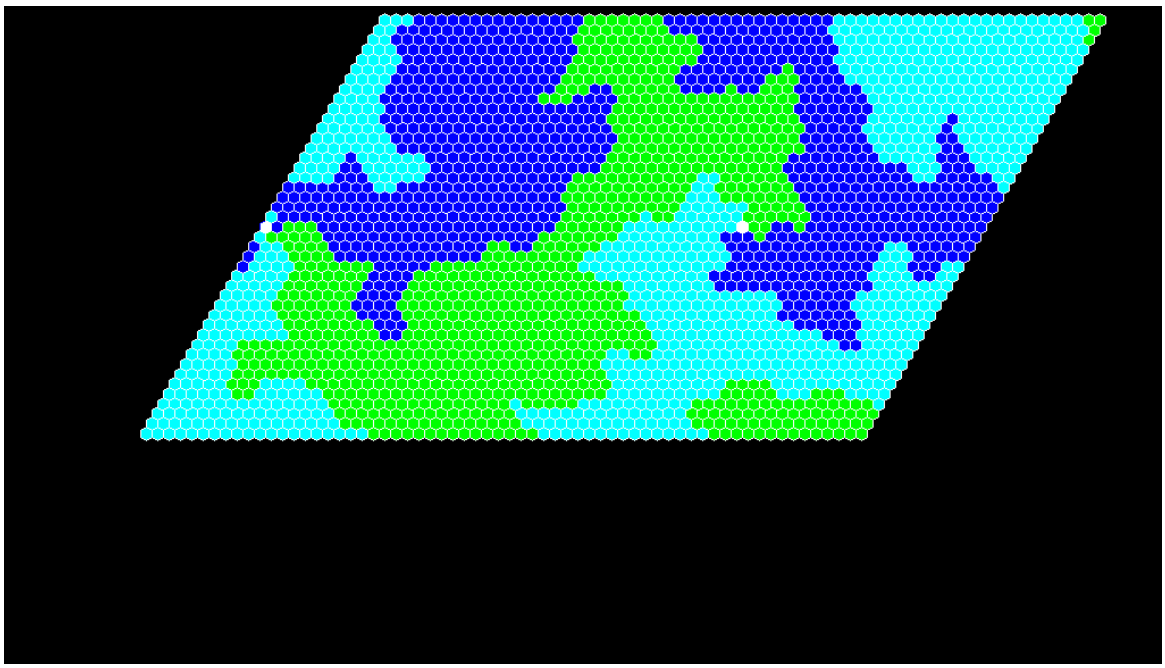

図 11

****

List 1:cag_2.c

```
/* t2.12                                                              */
/* 2018 Morio Kikuchi                                                 */

#include <windows.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>

#define VGACOLORS /*16*/50
#define GKS GetKeyState
#define ASIZE_MS ((1024*768)/CPMAX)
#define CPMAX 3/*12*/
#define X0 (330)
#define Y0 (10)
#define PIXSIZE 15
#define RESO 22

#define ICEIL(a,b) (((a)+((b)-1))/(b))

char refill,pauseflag,fieldflag,jmpflag;
char charcode,charflag;
int Ca,C1,C2,C3,C4,C5,C7,x[8],y[8],x_[8],y_[8],cc_sum[VGACOLORS];
int X,Y,X_,Y_,Nx,Ny,Nxp,Nxm,Nyp,Nym;
int algo,combination,drn,ig;
int yt,ssize,std_x,std_y,last_x,last_y;
long asize=ASIZE_MS;

/*unsigned */char **pixel;
int DS[][2]={{640,480},{640,480},{800,600},{1024,768},{1280,1024},{1600,1200}};
long fp_mem[CPMAX];
double hexagon_x[6],hexagon_y[6];

char function,usflag;
unsigned char yorn;
int XRESO,YRESO,WB,DX_FRAME,DY_FRAME,DY_CAPTION;
FILE *fp;

typedef struct {int xx,yy,xx_,yy_;} ss;
ss s;ss rtn[CPMAX][ASIZE_MS];
typedef struct {
unsigned char red,green,blue;} srgb;
```

```
srgb irgb[VGACOLORS];
typedef struct {
unsigned long back_;int back,fore;} bf;
bf bfset[]={{WHITENESS,15,0},{BLACKNESS,0,15}};

HINSTANCE hinstance;
HWND hwnd;
HDC hdcdisplay,hdctmp1;
HBITMAP hbitmap1;
HPEN hpen;
HBRUSH hbrush;

void closegraph_(void),initpalette(void),BitBlt_full(void),setup(void),
     cleardevice_(char,int,int,int,int),field(void),rectangle_(int,int,int,int),
     delay_(long),beep(long),kbhit_(void),restore_3(void),initgraph_return(void),
     use_subroop(void),keydowns_f2(void),bitblt(char,int,int,int,int,int,int),
     arrayreset(void),fwrite_mem(int),fread_mem(int),putpixel_(int,int,int),
     check_rcount(void);
unsigned char subroop(void);
int initgraph_(void),setup_(void),fourfloor_fiveceil(double),random_(int),
    getpixel_(int,int,int,int),cag_r(void);
long ftell_mem(int);
double getangle(int,int);

COLORREF PALETTE(int color);
LRESULT CALLBACK wndproc_by_kbhit_(HWND,UINT,WPARAM,LPARAM);
int wndproc_filer(HWND,UINT,WPARAM,LPARAM);


int main(int argc,unsigned char **argv)
{
long mytime;

WB=1;
refill=1;

if(initgraph_()==1) return 1;

cleardevice_(1,0,0,XRESO,YRESO);
BitBlt_full();

if(setup_()==1) return 1;

if(argc>1) {time(&mytime);srand((unsigned int)mytime);}
else
srand(1);
```

```
yt=2*(RESO-1);

combination=1;
drn=4;

field();
cag_r();

while(1){
check_rcount();
printf(" \n");

if(refill==0) break;
beep(50);

delay_(6000);
if(pauseflag==1) {pauseflag=0;use_subroop();}
if(refill==0) break;

/*drn=random_(6);*/

field();
cag_r();
}/**while(1)**/

closegraph_();

return 0;
}/** main **/


void ls_image(char flag,char *file,int x,int y,int dx,int dy)
{
unsigned long xsize,ysize,size;
unsigned long width,height,imagesize;
unsigned long bits,bytesPerPixel,lineSizeDW,lineSize;
HDC hdce,hdc;
HBITMAP hbitmape;
BITMAPFILEHEADER bfh;
BITMAPINFOHEADER bih;
BYTE *gdata;
FILE *fpo,*fpi;

if(flag<=3){                        /* save */
if(flag==0){
```

```
}
else if(flag==1){
}
else if(flag==2){
}
else if(flag==3){
}
else return;

if((fpo=fopen(file,"wb"))==NULL) {printf("Can't open a file.\n");return;}

width=dx;
height=dy;

bits=/*16*/24/*32*/;
bytesPerPixel=bits/8;
lineSizeDW=bytesPerPixel*width;
lineSizeDW=ICEIL(lineSizeDW,sizeof(long));
lineSize=lineSizeDW*sizeof(long);
imagesize=lineSize*height;

bfh.bfType=0x4d42;                    /* "BM" */
bfh.bfSize=54+imagesize;
bfh.bfReserved1=0;
bfh.bfOffBits=54;
bfh.bfReserved2=0;

bih.biSize=40;
bih.biWidth=width;
bih.biHeight=height;
bih.biPlanes=1;
bih.biBitCount=bits;
bih.biCompression=0;
bih.biSizeImage=imagesize;
bih.biXPelsPerMeter=0;
bih.biYPelsPerMeter=0;
bih.biClrUsed=0;
bih.biClrImportant=0;

if(flag<=1)
/*hdce=CreateCompatibleDC(hdctmp2)*/;
else if(flag==2)
hdce=CreateCompatibleDC(hdctmp1);
else{
hdc=CreateDC("DISPLAY",NULL,NULL,NULL);
hdce=CreateCompatibleDC(hdc);
```

```
}

hbitmape=CreateDIBSection(hdce,(LPBITMAPINFO)&bih,DIB_RGB_COLORS,&gdata,NULL,0);
SelectObject(hdce,hbitmape);

if(flag<=1)
/*BitBlt(hdce,0,0,dx,dy,hdctmp2,x,y,SRCCOPY)*/;
else if(flag==2)
BitBlt(hdce,0,0,dx,dy,hdctmp1,x,y,SRCCOPY);
else
BitBlt(hdce,0,0,dx,dy,hdc,x,y,SRCCOPY);

size=bih.biSizeImage;

fwrite(&bfh,14,1,fpo);
fwrite(&bih,40,1,fpo);
fwrite(gdata,size,1,fpo);

fclose(fpo);

if(flag==3) DeleteDC(hdc);
DeleteDC(hdce);
DeleteObject(hbitmape);
}
else{                                   /* load */
if((fpi=fopen(file,"rb"))==NULL) {printf("Can't open the file.\n");return;}

fread(&bfh,14,1,fpi);
if(bfh.bfType!=0x4d42) {fclose(fpi);printf("Not BM.\n");return;}
fread(&bih,40,1,fpi);

fseek(fpi,bfh.bfOffBits,0);
size=bih.biSizeImage;
gdata=(BYTE *)malloc(size);
fread(gdata,size,1,fpi);

/*StretchDIBits(hdctmp2,x,y,bih.biWidth,bih.biHeight,0,0,bih.biWidth,bih.biHeight,
              gdata,(LPBITMAPINFO)&bih,DIB_RGB_COLORS,SRCCOPY);*/

fclose(fpi);
free(gdata);
}
}/** ls_image **/


void fprintf_(char *str,int v2,int v3,int v4,int v5,int v6)
```

```
{
FILE *fp;

fp=fopen("cpage.bin","ab");

fprintf(fp," %s %d %d %d %d %d\n",str,v2,v3,v4,v5,v6);

fclose(fp);
}/** fprintf_ **/


void use_subroop(void)
{
char function_old,charflag_old;

usflag=1;

function_old=function;function=2;
charflag_old=charflag;

yorn=subroop();

function=function_old;
charflag=charflag_old;
}/** use_subroop **/


unsigned char subroop(void)
{
charflag=1;

while(1){
kbhit_();
if(charflag==0) return charcode;
}
}/** subroop **/


void keydowns_f2(void)
{
if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0) charflag=0;
else if(GKS('S')<0) {ls_image(2,"ss.bmp",0,0,XRESO,/*YRESO*/580);beep(300);}
}/** keydowns_f2 **/


void restore_in_PAINT(void)
```

```
{
ValidateRect(hwnd,NULL);

bitblt(1,0,0,XRESO,YRESO,0,0);
}/** restore_in_PAINT **/


void setup(void)
{
XRESO=1024-4;YRESO=768-24*2;

/*UNITDX=12;UNITDY=24;
COLUMN=XRESO/UNITDX;
ROW=YRESO/UNITDY;
CSRDY=UNITDY;
CSRCOLOR=15;*/
}/** setup **/


int setup_(void)
{
int i;

/*if(XRESO<=DS[0][0]-4){}
else {XRESO=DS[0][0]-4;}
if(YRESO<=DS[0][1]-24*4){}
else {YRESO=DS[0][1]-24*4;}*/

/*YRESO=480;*/

/*delta_x=(XRESO-2)/4;
XRESO=delta_x*4+2;*/
/*delta_x=(YRESO-4)/8;
XRESO=delta_x*8+4;*/
/*delta_x=(YRESO-8)/16;
XRESO=delta_x*16+8;
YRESO=delta_x*16+8;*/

/*COLUMN=XRESO/UNITDX;
ROW=YRESO/UNITDY;*/

pixel=(/*unsigned */char **)malloc(sizeof(/*unsigned */char *)*XRESO);
if(pixel==NULL){
DeleteDC(hdctmp1);
DeleteObject(hbitmap1);
initgraph_return();return 1;}
```

```
i=0;
while(1){
pixel[i]=(/*unsigned */char *)malloc(sizeof(/*unsigned */char)*YRESO);

if(pixel[i]==NULL){
while(1){
i--;
if(i<0) break;
free(pixel[i]);
}
free(pixel);
DeleteDC(hdctmp1);
DeleteObject(hbitmap1);
initgraph_return();return 1;}

i++;
if(i==XRESO) break;
}

return 0;
}/** setup_ **/


int initgraph_(void)
{
int i,width,height;
WNDCLASS wndclass;

setup();

wndclass.hInstance    =hinstance;
wndclass.lpszClassName="CAGCLASS";
wndclass.lpszMenuName =NULL;
wndclass.lpfnWndProc  =wndproc_by_kbhit_;
wndclass.style        =0;
wndclass.hIcon        =LoadIcon(hinstance,"MYICON");
wndclass.hCursor      =LoadCursor(NULL,IDC_ARROW);
wndclass.cbClsExtra   =0;
wndclass.cbWndExtra   =0;
if(WB==0)
wndclass.hbrBackground=GetStockObject(WHITE_BRUSH);
else
wndclass.hbrBackground=GetStockObject(BLACK_BRUSH);

if(RegisterClass(&wndclass)==0) return 1;
```

```
hwnd=CreateWindow("CAGCLASS"," CAG",
                  /*WS_POPUP,*/
                  WS_OVERLAPPED | WS_CAPTION | WS_SYSMENU | WS_MINIMIZEBOX,
                  0,0,XRESO+DX_FRAME,YRESO+DY_CAPTION+DY_FRAME,
                  NULL,NULL,hinstance,NULL);
if(hwnd==NULL) {MessageBox(NULL,"Memory space is not left.","CAG",MB_OK);return 1;}

SetWindowPos(hwnd,HWND_TOP,0,0,0,0,SWP_NOMOVE | SWP_NOSIZE);
ShowWindow(hwnd,SW_SHOWDEFAULT);

/*hdcdisplay=BeginPaint(hwnd,&paintstruct);*/
hdcdisplay=GetDC(hwnd);

hbitmap1=CreateCompatibleBitmap(hdcdisplay,XRESO,YRESO);
hdctmp1=CreateCompatibleDC(hdcdisplay);  /* text, dialog, menu */
SelectObject(hdctmp1,hbitmap1);
SetBkMode(hdcdisplay,TRANSPARENT);
SetBkMode(hdctmp1,TRANSPARENT);

initpalette();

SetBkColor(hdcdisplay,PALETTE(bfset[WB].back));
SetBkColor(hdctmp1,PALETTE(bfset[WB].back));

hexagon_x[0]=0.;
hexagon_x[1]=ff_fc(0.5*PIXSIZE);
hexagon_x[2]=ff_fc(1.*PIXSIZE);
hexagon_x[3]=ff_fc(1.*PIXSIZE);
hexagon_x[4]=ff_fc(0.5*PIXSIZE);
hexagon_x[5]=ff_fc(0.*PIXSIZE);

hexagon_y[0]=0.;
hexagon_y[1]=ff_fc((-sqrt(3)/6)*PIXSIZE);
hexagon_y[2]=0.;
hexagon_y[3]=ff_fc((sqrt(3)/3)*PIXSIZE);
hexagon_y[4]=ff_fc((sqrt(3)/2)*PIXSIZE);
hexagon_y[5]=ff_fc((sqrt(3)/3)*PIXSIZE);

return 0;
}/** initgraph_ **/


void initgraph_return(void)
{
/*EndPaint(hwnd,&paintstruct);*/
```

```
ReleaseDC(hwnd,hdcdisplay);
DestroyWindow(hwnd);
/*UnregisterClass("CAGCLASS",hinstance);*/

MessageBox(NULL,"Memory space is not left.","CAG",MB_OK);
}/** initgraph_return **/


void closegraph_(void)
{
int i;

i=0;
while(1){
free(pixel[i]);
i++;
if(i==XRESO) break;
}
free(pixel);

DeleteDC(hdctmp1);
DeleteObject(hbitmap1);

/*EndPaint(hwnd,&paintstruct);*/
ReleaseDC(hwnd,hdcdisplay);
DestroyWindow(hwnd);
/*UnregisterClass("CAGCLASS",hinstance);*/
}/** closegraph_ **/


void initpalette(void)
{
int i;

irgb[0].red=0;irgb[0].green=0;irgb[0].blue=0;

irgb[9].red=0;irgb[9].green=0;irgb[9].blue=255;  /* blue */
irgb[10].red=0;irgb[10].green=255;irgb[10].blue=0;  /* green */
irgb[11].red=0;irgb[11].green=255;irgb[11].blue=255;  /* cyan */
irgb[12].red=255;irgb[12].green=0;irgb[12].blue=0;  /* red */
irgb[13].red=255;irgb[13].green=0;irgb[13].blue=255;  /* magenta */
irgb[14].red=255;irgb[14].green=255;irgb[14].blue=0;  /* yellow */

irgb[15].red=255;irgb[15].green=255;irgb[15].blue=255;

for(i=7;i<9;i++){                   /* 7, 8 */
```

```
irgb[i].red=128+32*(9-i);
irgb[i].green=irgb[i].red;
irgb[i].blue=irgb[i].red;
}

for(i=16;i<20;i++){                 /* 16 -> 19 */
irgb[i].red=255-24*(20-i);
irgb[i].green=irgb[i].red;
irgb[i].blue=irgb[i].red;
}

for(i=1;i<7;i++){                   /* 1 -> 6 */
if(irgb[9+(i-1)].red==255)
irgb[i].red=irgb[9+(i-1)].red-24*1;
if(irgb[9+(i-1)].green==255)
irgb[i].green=irgb[9+(i-1)].green-24*1;
if(irgb[9+(i-1)].blue==255)
irgb[i].blue=irgb[9+(i-1)].blue-24*1;
}

for(i=20;i<26;i++){                 /* 20 -> 25 */
if(irgb[9+(i-20)].red==255)
irgb[i].red=irgb[9+(i-20)].red-24*2;
if(irgb[9+(i-20)].green==255)
irgb[i].green=irgb[9+(i-20)].green-24*2;
if(irgb[9+(i-20)].blue==255)
irgb[i].blue=irgb[9+(i-20)].blue-24*2;
}

for(i=26;i<32;i++){                 /* 26 -> 31 */
if(irgb[9+(i-26)].red==255)
irgb[i].red=irgb[9+(i-26)].red-24*3;
if(irgb[9+(i-26)].green==255)
irgb[i].green=irgb[9+(i-26)].green-24*3;
if(irgb[9+(i-26)].blue==255)
irgb[i].blue=irgb[9+(i-26)].blue-24*3;
}

for(i=32;i<38;i++){                 /* 32 -> 37 */
if(irgb[9+(i-32)].red==255)
irgb[i].red=irgb[9+(i-32)].red-24*4;
if(irgb[9+(i-32)].green==255)
irgb[i].green=irgb[9+(i-32)].green-24*4;
if(irgb[9+(i-32)].blue==255)
irgb[i].blue=irgb[9+(i-32)].blue-24*4;
}
```

```
for(i=38;i<44;i++){                 /* 38 -> 43 */
if(irgb[9+(i-38)].red==255)
irgb[i].red=irgb[9+(i-38)].red-24*5;
if(irgb[9+(i-38)].green==255)
irgb[i].green=irgb[9+(i-38)].green-24*5;
if(irgb[9+(i-38)].blue==255)
irgb[i].blue=irgb[9+(i-38)].blue-24*5;
}

for(i=44;i<50;i++){                 /* 44 -> 49 */
if(irgb[9+(i-44)].red==255)
irgb[i].red=irgb[9+(i-44)].red-24*6;
if(irgb[9+(i-44)].green==255)
irgb[i].green=irgb[9+(i-44)].green-24*6;
if(irgb[9+(i-44)].blue==255)
irgb[i].blue=irgb[9+(i-44)].blue-24*6;
}
}/** initpalette **/


void BitBlt_full(void)
{
bitblt(1,0,0,XRESO,YRESO,0,0);
}/** BitBlt_full **/


void bitblt(char flag,int x,int y,int xsize,int ysize,int x_,int y_)
{
BitBlt(hdcdisplay,x_,y_,xsize,ysize,
       hdctmp1,x,y,SRCCOPY);
}/** bitblt **/


void cleardevice_(char flag,int x,int y,int xsize,int ysize)
{
PatBlt(hdctmp1,x,y,xsize,ysize,bfset[WB].back_);
}/** cleardevice_ **/


COLORREF PALETTE(int color)
{
return RGB(irgb[color].red,irgb[color].green,irgb[color].blue);
}/** PALETTE **/
```

```
void kbhit_(void)
{
MSG msg;

if(PeekMessage(&msg,NULL,0,0,PM_REMOVE)){
TranslateMessage(&msg);
DispatchMessage(&msg);
}
}/** kbhit_ */


LRESULT CALLBACK  wndproc_by_kbhit_(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
{
if(wndproc_filer(hwnd,umsg,wparam,lparam)!=0) return 1;

return DefWindowProc(hwnd,umsg,wparam,lparam);
}/** wndproc_by_kbhit_ **/


int wndproc_filer(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
{
if(umsg==WM_KEYDOWN){
/************************** menu keydowns -> ***************************/
/************************** <- menu keydowns ***************************/

/************************** dialog keydowns -> ***************************/
/************************** <- dialog keydowns ***************************/

if(function==2){
keydowns_f2();
return 1;
}

if(usflag==1) usflag=0;

     if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0) refill=0;
else if(GKS(VK_SHIFT)<0) pauseflag=1;

return 1;
}/**else if(umsg)**/
else if(umsg==WM_SYSKEYDOWN){
}/**else if(umsg)**/
else if(umsg==WM_CLOSE){
if(function==2) charflag=0;
else refill=0;
```

```
return 1;
}/**else if(umsg)**/
else if(umsg==WM_PAINT){
restore_in_PAINT();

return 1;
}/**else if(umsg)**/
else{}

return 0;
}/** wndproc_filer **/


void delay_(long millisecond)
{
long oldtime,nowtime,dtime;
double i=CLOCKS_PER_SEC,j;

j=millisecond;
millisecond=j*(i/1000.);
oldtime=clock();

while(1){
kbhit_();
if(pauseflag==1 && refill==0) {pauseflag=0;refill=1;break;}
if(refill==0) break;

nowtime=clock();dtime=nowtime-oldtime;
if(dtime>=millisecond) break;
if(dtime<0) break;
}
}/** delay_ **/


void beep(long millisecond)
{
Beep(888,millisecond);
}/** beep **/


int fourfloor_fiveceil(double val_d)
{
int val_i,val;

val_i=floor(val_d);
val=(val_d-val_i<0.5)?val_i:val_i+1;
```

```
return val;
}/** fourfloor_fiveceil **/


int ff_fc(double val_d)
{
return fourfloor_fiveceil(val_d);
}/** ff_fc **/


void arrayreset(void)
{
int i,j;

i=0;
while(1){

j=0;
while(1){
pixel[i][j]=0;
j++;
if(j==YRESO) break;
}

i++;
if(i==XRESO) break;
}
}/** arrayreset **/


int putpixel(int nx,int ny,int pcolor)
{
int i,dx,dy;
POINT vertex[7];

if(nx<0 || ny<0 || nx>RESO-1 || ny>yt) return 0;

dx=ff_fc(X0+nx*1.0*PIXSIZE-ny*0.5*PIXSIZE);
dy=ff_fc(Y0+ny*(sqrt(3)/2)*PIXSIZE);

i=0;
while(1){
vertex[i].x=hexagon_x[i]+dx;
vertex[i].y=hexagon_y[i]+dy;
i++;if(i==6) break;
```

```
}

vertex[6].x=vertex[0].x;
vertex[6].y=vertex[0].y;

if(pcolor==15)
hpen=CreatePen(PS_SOLID,1,PALETTE(9));
else
hpen=CreatePen(PS_SOLID,1,PALETTE(15));

/*if(nx==0 || ny==0 || nx==RESO-1 || ny==RESO-1) pcolor=12;*/
hbrush=CreateSolidBrush(PALETTE(pcolor));

SelectObject(hdcdisplay,hpen);
SelectObject(hdcdisplay,hbrush);

Polyline(hdcdisplay,vertex,6+1);
Polygon(hdcdisplay,vertex,6);

SelectObject(hdctmp1,hpen);
SelectObject(hdctmp1,hbrush);

Polyline(hdctmp1,vertex,6+1);
Polygon(hdctmp1,vertex,6);

DeleteObject(hbrush);
DeleteObject(hpen);

pixel[nx][ny]=pcolor;

return 0;
}/** putpixel **/


void check_rcount(void)
{
int i,j,k,m,n,dx,dy,Li,Lj;

Li=1;Lj=2;

/*999*/
for(j=0;j<Lj;j++)
for(i=0;i<Li;i++){
dx=(RESO-1)*i;
dy=(RESO-1)*j;
```

```
/* left */
if(pixel[dx][dy]!=0 && pixel[dx+RESO-1][dy+RESO-1]!=0 && pixel[dx][dy+RESO-1]!=0){
for(k=0;k<VGACOLORS;k++)
cc_sum[k]=0;

for(m=0;m<RESO;m++)
for(n=0;n<RESO;n++) {if(m>=n) cc_sum[pixel[dx+n][dy+m]]++;}

if((cc_sum[9]==cc_sum[10])&&(cc_sum[9]==cc_sum[11])){
printf(" i=%d j=%d left\n",i,j);

for(k=0;k<VGACOLORS;k++)
if(cc_sum[k]) printf(" cc%2d:%ld\n",k,cc_sum[k]);
}
}

/* right */
if(pixel[dx][dy]!=0 && pixel[dx+RESO-1][dy+RESO-1]!=0 && pixel[dx+RESO-1][dy]!=0){
for(k=0;k<VGACOLORS;k++)
cc_sum[k]=0;

for(m=0;m<RESO;m++)
for(n=0;n<RESO;n++) {if(m<=n) cc_sum[pixel[dx+n][dy+m]]++;}

if((cc_sum[9]==cc_sum[10])&&(cc_sum[9]==cc_sum[11])){
printf(" i=%d j=%d right\n",i,j);

for(k=0;k<VGACOLORS;k++)
if(cc_sum[k]) printf(" cc%2d:%ld\n",k,cc_sum[k]);
}
}
}/**for(i)**/
}/** check_rcount **/


void field_rect(int x,int y,int dx,int dy)
{
int i,j;

fieldflag=1;

for(j=y;j<y+dy;j++)
for(i=x;i<x+dx;i++){
putpixel_(i,j,15);
}
```

```
fieldflag=0;
}/** field_rect **/


void field(void)
{
field_rect(0,0,RESO,2*RESO-1);
}/** field **/


void putpixel_(int nx,int ny,int pcolor)
{
char flag;

putpixel(nx,ny,pcolor);
if(fieldflag) return;
/*return;*/

if(nx==0 && ny==RESO-1)           flag=0;
else if(nx==RESO-1 && ny==RESO-1) flag=0;
else if((nx==0 && ny==0) || (nx==0 && ny==yt))           flag=1;
else if((nx==RESO-1 && ny==0) || (nx==RESO-1 && ny==yt)) flag=1;

else if(ny==nx) flag=0;                     /* (1-24)1-4 */
else if(ny==RESO-1) flag=0;                 /* (13-24)3-4 */
else if(ny==nx+RESO-1) flag=0;              /* (13-4)3-6 */
else if(nx>=1 && nx<=RESO-2 && ny==0)       flag=1;  /* (5)7 */
else if(nx>=1 && nx<=RESO-2 && ny==yt)      flag=1;  /* (6)8 */
else if(nx==0 && ny>=1 && ny<=RESO-2)       flag=2;  /* (7)9 */
else if(nx==0 && ny>=RESO && ny<=yt-1)      flag=2;  /* (8)11 */
else if(nx==RESO-1 && ny>=1 && ny<=RESO-2)  flag=2;  /* (9)10 */
else if(nx==RESO-1 && ny>=RESO && ny<=yt-1) flag=2;  /* (10)12 */
else flag=-1;

if(flag==0){
}
else if(flag>0){
if(flag==1) {ny=yt-ny;}
else        {ny=yt-ny;}

putpixel(nx,ny,pcolor);
}

/*fprintf_("pp",flag,nx,ny,-1,-1);*/
}/** putpixel_ **/
```

```
void putpixel_2(int nx,int ny,int pcolor)
{
char flag;

if(nx==0 && ny==RESO-1)           flag=0;
else if(nx==RESO-1 && ny==RESO-1) flag=0;
else if((nx==0 && ny==0) || (nx==0 && ny==yt))           flag=1;
else if((nx==RESO-1 && ny==0) || (nx==RESO-1 && ny==yt)) flag=1;

else if(ny==nx) flag=0;                     /* (1-24)1-4 */
else if(ny==RESO-1) flag=0;                 /* (13-24)3-4 */
else if(ny==nx+RESO-1) flag=0;              /* (13-4)3-6 */
else if(nx>=1 && nx<=RESO-2 && ny==0)       flag=1;  /* (5)7 */
else if(nx>=1 && nx<=RESO-2 && ny==yt)      flag=1;  /* (6)8 */
else if(nx==0 && ny>=1 && ny<=RESO-2)       flag=2;  /* (7)9 */
else if(nx==0 && ny>=RESO && ny<=yt-1)      flag=2;  /* (8)11 */
else if(nx==RESO-1 && ny>=1 && ny<=RESO-2)  flag=2;  /* (9)10 */
else if(nx==RESO-1 && ny>=RESO && ny<=yt-1) flag=2;  /* (10)12 */
else flag=-1;

X_=nx;Y_=ny;

if(flag==0){
}
else if(flag>0){
if(flag==1) {ny=yt-ny;}
else        {ny=yt-ny;}

X_=nx;Y_=ny;
}
}/** putpixel_2 **/


int getpixel_(int x,int y,int nx,int ny)
{
int flag;

/*if(nx<0 || ny<0 || nx>RESO-1 || ny>yt) return 0;*/

     if(x==0 && y==0)                       flag=1;
else if(x==RESO-1 && y==0)                  flag=2;
else if(x==0 && y==yt)                      flag=5;
else if(x==RESO-1 && y==yt)                 flag=6;
else if(x==0 && y==RESO-1)                  flag=3;
else if(x==RESO-1 && y==RESO-1)             flag=4;
```

```
else if(x>=1 && x<=RESO-2 && y==0)           flag=7;
else if(x>=1 && x<=RESO-2 && y==yt)          flag=8;
else if(x==0 && y>=1 && y<=RESO-2)           flag=9;
else if(x==RESO-1 && y>=1 && y<=RESO-2)      flag=11;
else if(x==0 && y>=RESO && y<=yt-1)          flag=10;
else if(x==RESO-1 && y>=RESO && y<=yt-1)     flag=12;
else flag=0;

X=nx;Y=ny;
jmpflag=0;
/*goto end;*/

if(flag==5){
     if(nx==x-1 && ny==y-1) {X=0+1;Y=0+1;jmpflag=2;}
else if(nx<0 || ny>yt) return 0;
}
else if(flag==2){
     if(nx==x+1 && ny==y+1) {X=RESO-2;Y=2*RESO-3;jmpflag=3;}
else if(nx>RESO-1 || ny<0) return 0;
}
else if(flag==1){
if(nx<0 || ny<0/* || nx>RESO-1 || ny>yt*/) return 0;
}
else if(flag==6){
if(/*nx<0 || ny<0 || */nx>RESO-1 || ny>yt) return 0;
}
else if(flag==3){
if(nx<0) return 0;
}
else if(flag==4){
if(nx>RESO-1) return 0;
}
else if(flag==7){
if(ny==-1) {X=nx;Y=yt-1;jmpflag=1;}
}
else if(flag==8){
if(ny==yt+1) {X=nx;Y=1;jmpflag=1;}
}
else if(flag==9){
if(nx==-1) {X=1;Y=yt-ny;jmpflag=2;}
}
else if(flag==11){
if(nx==RESO) {X=RESO-2;Y=yt-ny;jmpflag=2;}
}
else if(flag==10){
if(nx==-1) {X=1;Y=yt-ny;jmpflag=2;}
```

```
}
else if(flag==12){
if(nx==RESO) {X=RESO-2;Y=yt-ny;jmpflag=2;}
}

/*if(nx<0 || ny<0 || nx>RESO-1 || ny>RESO-1)
fprintf_("gp",flag,x,y,nx,ny);*/

end:
return pixel[X][Y];
}/** getpixel_ **/
#if 0
/* Figure 6 */
int getpixel_(int x,int y,int nx,int ny)
{
int flag;

/*if(nx<0 || ny<0 || nx>RESO-1 || ny>yt) return 0;*/
/*if(nx<0 || ny<0 || nx>xt || ny>yt) return 0;
if(nx>RESO-1 && ny>RESO-1) return 0;*/

     if(x==0 && y==0)           flag=1;
else if(x==RESO-1 && y==0)      flag=2;
else if(x==xt && y==0)          flag=3;
else if(x==0 && y==RESO-1)      flag=4;
else if(x==RESO-1 && y==RESO-1) flag=5;
else if(x==xt && y==RESO-1)     flag=6;
else if(x==0 && y==yt)          flag=7;
else if(x==RESO-1 && y==yt)     flag=8;

else if(x==RESO-1 && y>RESO-1 && y<yt) flag=15;
else if(x>RESO-1 && x<xt && y==RESO-1) flag=11;
else if(x>0 && x<RESO-1 && y==yt)      flag=12;
else if(x==xt && y>0 && y<RESO-1)      flag=16;
else if(x>0 && x<RESO-1 && y==0)       flag=9;
else if(x>RESO-1 && x<xt && y==0)      flag=10;
else if(x==0 && y>0 && y<RESO-1)       flag=13;
else if(x==0 && y>RESO-1 && y<yt)      flag=14;

else flag=0;

X=nx;Y=ny;
jmpflag=0;
/*goto end;*/

if(flag==2){
```

```
if(ny<0) return 0;
}
else if(flag==4){
if(nx<0) return 0;
}
else if(flag==5){
if(nx>RESO-1 && ny>RESO-1) return 0;
}
else if(flag==1){
     if(nx==x && ny==y-1) {X=xt;Y=0+1;jmpflag=2;}
else if(nx<0 || ny<0) return 0;
}
else if(flag==3){
     if(nx==x+1 && ny==y+1) {X=0;Y=yt-1;jmpflag=2;}
else if(nx==x+1 && ny==y) {X=0+1;Y=0+1;jmpflag=2;}
else if(nx>xt || ny<0) return 0;
}
else if(flag==7){
     if(nx==x-1 && ny==y-1) {X=0+1;Y=0+1;jmpflag=3;}
else if(nx==x-1 && ny==y) {X=0+1;Y=0;jmpflag=3;}
else if(nx<0 || ny>yt) return 0;
}
else if(flag==8){
     if(nx==x && ny==y+1) {X=y-1;Y=x-1;jmpflag=1;}
else if(nx>RESO-1 || ny>yt) return 0;
}
else if(flag==6){
     if(nx==x && ny==y+1) {X=y-1;Y=x-1;jmpflag=1;}
else if(nx>RESO-1 || ny>yt) return 0;
}
else if(flag==13 || flag==14){
if(nx==-1) {X=1;Y=yt-ny;jmpflag=3;}
}
else if(flag==9 || flag==10){
if(ny==-1) {X=xt-nx;Y=1;jmpflag=2;}
}
else if(flag==11){
     if(nx==x && ny==y+1) {X=y-1;Y=x-1;jmpflag=1;}
else if(nx==x+1 && ny==y+1) {X=y-1;Y=x;jmpflag=1;}
}
else if(flag==15){
     if(nx==x+1 && ny==y+1) {X=y;Y=x-1;jmpflag=1;}
else if(nx==x+1 && ny==y) {X=y-1;Y=x-1;jmpflag=1;}
}
else if(flag==12){
     if(nx==x && ny==y+1) {X=y-1;Y=x-1;jmpflag=1;}
```

```
else if(nx==x+1 && ny==y+1) {X=y-1;Y=x;jmpflag=1;}
}
else if(flag==16){
     if(nx==x+1 && ny==y+1) {X=y;Y=x-1;jmpflag=1;}
else if(nx==x+1 && ny==y) {X=y-1;Y=x-1;jmpflag=1;}
}

/*if(nx<0 || ny<0 || nx>RESO-1 || ny>RESO-1)
fprintf_("gp",flag,x,y,nx,ny);*/

end:
return pixel[X][Y];
}/** getpixel_ **/
#endif
#if 0
/* Figure 8 */
int getpixel_(int x,int y,int nx,int ny)
{
int flag;

/*if(nx<0 || ny<0 || nx>RESO-1 || ny>yt) return 0;*/
/*if(nx<0 || ny<0 || nx>xt || ny>yt) return 0;*/

     if(x==RESO-1 && y==0)  flag=2;
else if(x==0 && y==RESO-1)  flag=5;
else if(x==xt && y==RESO-1) flag=8;
else if(x==xtm && y==yt)    flag=11;
else if(x==0 && y==0)       flag=1;
else if(x==xtm && y==0)     flag=3;
else if(x==0 && y==yt)      flag=9;
else if(x==RESO-1 && y==yt) flag=10;
else if(x==xt && y==yt)     flag=12;
else if(x==xt && y==0)      flag=4;

else if(x>0 && x<RESO-1 && y==0)    flag=13;
else if(x>RESO-1 && x<xtm && y==0)  flag=14;
else if(x>RESO-1 && x<xtm && y==yt) flag=17;
else if(x>xtm && x<xt && y==yt)     flag=18;
else if(x==0 && y>0 && y<RESO-1)    flag=19;
else if(x==0 && y>RESO-1 && y<yt)   flag=20;
else if(x==xt && y>0 && y<RESO-1)   flag=21;
else if(x==xt && y>RESO-1 && y<yt)  flag=22;
else if(x>xtm && x<xt && y==0)      flag=15;
else if(x>0 && x<RESO-1 && y==yt)   flag=16;

else flag=0;
```

```
X=nx;Y=ny;
jmpflag=0;
/*goto end;*/

if(flag==2){
if(ny<0) return 0;
}
else if(flag==5){
if(nx<0) return 0;
}
else if(flag==8){
if(nx>xt) return 0;
}
else if(flag==11){
if(ny>yt) return 0;
}

else if(flag==1){
     if(nx==x && ny==y-1) {X=xtm;Y=0+1;jmpflag=2;}
else if(nx==x-1 && ny==y-1) {X=xtm+1;Y=0+1;jmpflag=2;}
else if(nx==x-1 && ny==y) {X=xtm+1;Y=0;jmpflag=2;}
}
else if(flag==3){
     if(nx==x && ny==y-1) {X=0;Y=0+1;jmpflag=2;}
else if(nx==x-1 && ny==y-1) {X=0+1;Y=0+1;jmpflag=2;}
}
else if(flag==9){
     if(nx==x-1 && ny==y-1) {X=0+1;Y=0+1;jmpflag=4;}
else if(nx==x-1 && ny==y) {X=0+1;Y=0;jmpflag=4;}
else if(nx==x && ny==y+1) {X=xtm;Y=0+1;jmpflag=1;}
else if(nx==x+1 && ny==y+1) {X=xtm+1;Y=0+1;jmpflag=1;}
}
else if(flag==10){
     if(nx==x && ny==y+1) {X=xt;Y=yt-1;jmpflag=3;}
else if(nx==x+1 && ny==y+1) {X=xt-1;Y=yt-1;jmpflag=3;}
}
else if(flag==12){
     if(nx==x && ny==y+1) {X=RESO-1;Y=yt-1;jmpflag=3;}
else if(nx==x+1 && ny==y+1) {X=RESO-2;Y=yt-1;jmpflag=3;}
else if(nx==x+1 && ny==y) {X=RESO-2;Y=yt;jmpflag=3;}
}
else if(flag==4){
     if(nx==x+1 && ny==y+1) {X=xt-1;Y=yt-1;jmpflag=4;}
else if(nx==x+1 && ny==y) {X=RESO;Y=yt;jmpflag=5;}
else if(nx==x && ny==y-1) {X=RESO-1;Y=yt-1;jmpflag=5;}
```

```
else if(nx==x-1 && ny==y-1) {X=RESO-2;Y=yt-1;jmpflag=5;}
}

else if(flag==13 || flag==14){
if(ny==-1) {X=xtm-nx;Y=1;jmpflag=2;}
}
else if(flag==17 || flag==18){
if(ny==yt+1) {X=xtp-nx;Y=yt-1;jmpflag=3;}
}
else if(flag==19 || flag==20){
if(nx==-1) {X=1;Y=yt-ny;jmpflag=4;}
}
else if(flag==21 || flag==22){
if(nx==xt+1) {X=xt-1;Y=yt-ny;jmpflag=4;}
}
else if(flag==15){
     if(nx==x && ny==-1) {X=x-2*(RESO-1);Y=yt-1;jmpflag=5;}
else if(nx==x-1 && ny==-1) {X=x-1-2*(RESO-1);Y=yt-1;jmpflag=5;}
}
else if(flag==16){
     if(nx==x && ny==yt+1) {X=x+2*(RESO-1);Y=1;jmpflag=6;}
else if(nx==x+1 && ny==yt+1) {X=x+1+2*(RESO-1);Y=1;jmpflag=6;}
}

/*if(nx<0 || ny<0 || nx>RESO-1 || ny>RESO-1)
fprintf_("gp",flag,x,y,nx,ny);*/

end:
return pixel[X][Y];
}/** getpixel_ **/
#endif


int random_(int n)
{
int val;

val=(int)((rand()/(RAND_MAX+1.))*n);

return val;
}/** random_ **/


long ftell_mem(int i)
{
return fp_mem[i];
```

```
}/** ftell_mem **/


void fwrite_mem(int i)
{
rtn[i][fp_mem[i]]=s;
fp_mem[i]++;if(fp_mem[i]>asize-1) refill=0;
}/** fwrite_mem **/


void fread_mem(int i)
{
fp_mem[i]--;if(fp_mem[i]<0) fp_mem[i]=0;
s=rtn[i][fp_mem[i]];
}/** fread_mem **/


double getangle(int next_x,int next_y)
{
char sign;
long product,p1,p2;
double val;

if(next_x==last_x && next_y==last_y) {beep(10000);refill=0;return 0;}

product=(next_x-std_x)*(std_y-last_y)-(next_y-std_y)*(std_x-last_x);
if(product==0) sign=0;
else if(product>0) sign=1;
else sign=-1;

if(sign==0) return 0;
else{
product=(next_x-std_x)*(std_x-last_x)+(next_y-std_y)*(std_y-last_y);
p1=(next_x-std_x)*(next_x-std_x)+(next_y-std_y)*(next_y-std_y);
p2=(std_x-last_x)*(std_x-last_x)+(std_y-last_y)*(std_y-last_y);
val=(double)product/(sqrt((double)p1)*sqrt((double)p2));

if(val>1) val=1;
else if(val<-1) val=-1;

return sign*acos(val);
}
}/** getangle **/


void mod_XY(int flag)
```

```
{
if(flag!=1) {putpixel_2(Nxp,Ny,0);x_[1]=X_;y_[1]=Y_;}
if(flag!=5) {putpixel_2(Nxp,Nyp,0);x_[5]=X_;y_[5]=Y_;}
if(flag!=2) {putpixel_2(Nx,Nyp,0);x_[2]=X_;y_[2]=Y_;}
if(flag!=3) {putpixel_2(Nxm,Ny,0);x_[3]=X_;y_[3]=Y_;}
if(flag!=7) {putpixel_2(Nxm,Nym,0);x_[7]=X_;y_[7]=Y_;}
if(flag!=4) {putpixel_2(Nx,Nym,0);x_[4]=X_;y_[4]=Y_;}
}/** mod_XY **/


int nv(int flag,int cc,int plc)
{
int order;

if(0) {if(cc==Ca) return 0;else return 1;}

order=1;

while(1){
if(flag==0){  /* for CW */
if(plc==1){
if(order==1){
mod_XY(5);
  if(C1==Ca) {if(x_[1]!=x[5] || y_[1]!=y[5]) return 0;}else if(C1!=0) return 1;}
else if(order==2){
  if(C4==Ca) {if(x_[4]!=x[5] || y_[4]!=y[5]) return 0;}else if(C4!=0) return 1;}
else if(order==3){
  if(C7==Ca) {if(x_[7]!=x[5] || y_[7]!=y[5]) return 0;}else if(C7!=0) return 1;}
else if(order==4){
  if(C3==Ca) {if(x_[3]!=x[5] || y_[3]!=y[5]) return 0;}else if(C3!=0) return 1;}
else if(order==5){
  if(C2==Ca) {if(x_[2]!=x[5] || y_[2]!=y[5]) return 0;}else if(C2!=0) return 1;}
else if(order==6){
  if(C5==Ca) {if(x_[5]!=x[5] || y_[5]!=y[5]) return 0;}else if(C5!=0) return 1;}
}
else if(plc==4){
if(order==1){
mod_XY(1);
  if(C4==Ca) {if(x_[4]!=x[1] || y_[4]!=y[1]) return 0;}else if(C4!=0) return 1;}
else if(order==2){
  if(C7==Ca) {if(x_[7]!=x[1] || y_[7]!=y[1]) return 0;}else if(C7!=0) return 1;}
else if(order==3){
  if(C3==Ca) {if(x_[3]!=x[1] || y_[3]!=y[1]) return 0;}else if(C3!=0) return 1;}
else if(order==4){
  if(C2==Ca) {if(x_[2]!=x[1] || y_[2]!=y[1]) return 0;}else if(C2!=0) return 1;}
else if(order==5){
```

```
  if(C5==Ca) {if(x_[5]!=x[1] || y_[5]!=y[1]) return 0;}else if(C5!=0) return 1;}
else if(order==6){
  if(C1==Ca) {if(x_[1]!=x[1] || y_[1]!=y[1]) return 0;}else if(C1!=0) return 1;}
}
else if(plc==7){
if(order==1){
mod_XY(4);
  if(C7==Ca) {if(x_[7]!=x[4] || y_[7]!=y[4]) return 0;}else if(C7!=0) return 1;}
else if(order==2){
  if(C3==Ca) {if(x_[3]!=x[4] || y_[3]!=y[4]) return 0;}else if(C3!=0) return 1;}
else if(order==3){
  if(C2==Ca) {if(x_[2]!=x[4] || y_[2]!=y[4]) return 0;}else if(C2!=0) return 1;}
else if(order==4){
  if(C5==Ca) {if(x_[5]!=x[4] || y_[5]!=y[4]) return 0;}else if(C5!=0) return 1;}
else if(order==5){
  if(C1==Ca) {if(x_[1]!=x[4] || y_[1]!=y[4]) return 0;}else if(C1!=0) return 1;}
else if(order==6){
  if(C4==Ca) {if(x_[4]!=x[4] || y_[4]!=y[4]) return 0;}else if(C4!=0) return 1;}
}
else if(plc==3){
if(order==1){
mod_XY(7);
  if(C3==Ca) {if(x_[3]!=x[7] || y_[3]!=y[7]) return 0;}else if(C3!=0) return 1;}
else if(order==2){
  if(C2==Ca) {if(x_[2]!=x[7] || y_[2]!=y[7]) return 0;}else if(C2!=0) return 1;}
else if(order==3){
  if(C5==Ca) {if(x_[5]!=x[7] || y_[5]!=y[7]) return 0;}else if(C5!=0) return 1;}
else if(order==4){
  if(C1==Ca) {if(x_[1]!=x[7] || y_[1]!=y[7]) return 0;}else if(C1!=0) return 1;}
else if(order==5){
  if(C4==Ca) {if(x_[4]!=x[7] || y_[4]!=y[7]) return 0;}else if(C4!=0) return 1;}
else if(order==6){
  if(C7==Ca) {if(x_[7]!=x[7] || y_[7]!=y[7]) return 0;}else if(C7!=0) return 1;}
}
else if(plc==2){
if(order==1){
mod_XY(3);
  if(C2==Ca) {if(x_[2]!=x[3] || y_[2]!=y[3]) return 0;}else if(C2!=0) return 1;}
else if(order==2){
  if(C5==Ca) {if(x_[5]!=x[3] || y_[5]!=y[3]) return 0;}else if(C5!=0) return 1;}
else if(order==3){
  if(C1==Ca) {if(x_[1]!=x[3] || y_[1]!=y[3]) return 0;}else if(C1!=0) return 1;}
else if(order==4){
  if(C4==Ca) {if(x_[4]!=x[3] || y_[4]!=y[3]) return 0;}else if(C4!=0) return 1;}
else if(order==5){
  if(C7==Ca) {if(x_[7]!=x[3] || y_[7]!=y[3]) return 0;}else if(C7!=0) return 1;}
```

```
else if(order==6){
  if(C3==Ca) {if(x_[3]!=x[3] || y_[3]!=y[3]) return 0;}else if(C3!=0) return 1;}
}
else if(plc==5){
if(order==1){
mod_XY(2);
  if(C5==Ca) {if(x_[5]!=x[2] || y_[5]!=y[2]) return 0;}else if(C5!=0) return 1;}
else if(order==2){
  if(C1==Ca) {if(x_[1]!=x[2] || y_[1]!=y[2]) return 0;}else if(C1!=0) return 1;}
else if(order==3){
  if(C4==Ca) {if(x_[4]!=x[2] || y_[4]!=y[2]) return 0;}else if(C4!=0) return 1;}
else if(order==4){
  if(C7==Ca) {if(x_[7]!=x[2] || y_[7]!=y[2]) return 0;}else if(C7!=0) return 1;}
else if(order==5){
  if(C3==Ca) {if(x_[3]!=x[2] || y_[3]!=y[2]) return 0;}else if(C3!=0) return 1;}
else if(order==6){
  if(C2==Ca) {if(x_[2]!=x[2] || y_[2]!=y[2]) return 0;}else if(C2!=0) return 1;}
}
}/**if(flag)**/
else{  /* for CCW */
if(plc==1){
if(order==1){
mod_XY(4);
  if(C1==Ca) {if(x_[1]!=x[4] || y_[1]!=y[4]) return 0;}else if(C1!=0) return 1;}
else if(order==2){
  if(C5==Ca) {if(x_[5]!=x[4] || y_[5]!=y[4]) return 0;}else if(C5!=0) return 1;}
else if(order==3){
  if(C2==Ca) {if(x_[2]!=x[4] || y_[2]!=y[4]) return 0;}else if(C2!=0) return 1;}
else if(order==4){
  if(C3==Ca) {if(x_[3]!=x[4] || y_[3]!=y[4]) return 0;}else if(C3!=0) return 1;}
else if(order==5){
  if(C7==Ca) {if(x_[7]!=x[4] || y_[7]!=y[4]) return 0;}else if(C7!=0) return 1;}
else if(order==6){
  if(C4==Ca) {if(x_[4]!=x[4] || y_[4]!=y[4]) return 0;}else if(C4!=0) return 1;}
}
else if(plc==5){
if(order==1){
mod_XY(1);
  if(C5==Ca) {if(x_[5]!=x[1] || y_[5]!=y[1]) return 0;}else if(C5!=0) return 1;}
else if(order==2){
  if(C2==Ca) {if(x_[2]!=x[1] || y_[2]!=y[1]) return 0;}else if(C2!=0) return 1;}
else if(order==3){
  if(C3==Ca) {if(x_[3]!=x[1] || y_[3]!=y[1]) return 0;}else if(C3!=0) return 1;}
else if(order==4){
  if(C7==Ca) {if(x_[7]!=x[1] || y_[7]!=y[1]) return 0;}else if(C7!=0) return 1;}
else if(order==5){
```

```
  if(C4==Ca) {if(x_[4]!=x[1] || y_[4]!=y[1]) return 0;}else if(C4!=0) return 1;}
else if(order==6){
  if(C1==Ca) {if(x_[1]!=x[1] || y_[1]!=y[1]) return 0;}else if(C1!=0) return 1;}
}
else if(plc==2){
if(order==1){
mod_XY(5);
  if(C2==Ca) {if(x_[2]!=x[5] || y_[2]!=y[5]) return 0;}else if(C2!=0) return 1;}
else if(order==2){
  if(C3==Ca) {if(x_[3]!=x[5] || y_[3]!=y[5]) return 0;}else if(C3!=0) return 1;}
else if(order==3){
  if(C7==Ca) {if(x_[7]!=x[5] || y_[7]!=y[5]) return 0;}else if(C7!=0) return 1;}
else if(order==4){
  if(C4==Ca) {if(x_[4]!=x[5] || y_[4]!=y[5]) return 0;}else if(C4!=0) return 1;}
else if(order==5){
  if(C1==Ca) {if(x_[1]!=x[5] || y_[1]!=y[5]) return 0;}else if(C1!=0) return 1;}
else if(order==6){
  if(C5==Ca) {if(x_[5]!=x[5] || y_[5]!=y[5]) return 0;}else if(C5!=0) return 1;}
}
else if(plc==3){
if(order==1){
mod_XY(2);
  if(C3==Ca) {if(x_[3]!=x[2] || y_[3]!=y[2]) return 0;}else if(C3!=0) return 1;}
else if(order==2){
  if(C7==Ca) {if(x_[7]!=x[2] || y_[7]!=y[2]) return 0;}else if(C7!=0) return 1;}
else if(order==3){
  if(C4==Ca) {if(x_[4]!=x[2] || y_[4]!=y[2]) return 0;}else if(C4!=0) return 1;}
else if(order==4){
  if(C1==Ca) {if(x_[1]!=x[2] || y_[1]!=y[2]) return 0;}else if(C1!=0) return 1;}
else if(order==5){
  if(C5==Ca) {if(x_[5]!=x[2] || y_[5]!=y[2]) return 0;}else if(C5!=0) return 1;}
else if(order==6){
  if(C2==Ca) {if(x_[2]!=x[2] || y_[2]!=y[2]) return 0;}else if(C2!=0) return 1;}
}
else if(plc==7){
if(order==1){
mod_XY(3);
  if(C7==Ca) {if(x_[7]!=x[3] || y_[7]!=y[3]) return 0;}else if(C7!=0) return 1;}
else if(order==2){
  if(C4==Ca) {if(x_[4]!=x[3] || y_[4]!=y[3]) return 0;}else if(C4!=0) return 1;}
else if(order==3){
  if(C1==Ca) {if(x_[1]!=x[3] || y_[1]!=y[3]) return 0;}else if(C1!=0) return 1;}
else if(order==4){
  if(C5==Ca) {if(x_[5]!=x[3] || y_[5]!=y[3]) return 0;}else if(C5!=0) return 1;}
else if(order==5){
  if(C2==Ca) {if(x_[2]!=x[3] || y_[2]!=y[3]) return 0;}else if(C2!=0) return 1;}
```

```
else if(order==6){
  if(C3==Ca) {if(x_[3]!=x[3] || y_[3]!=y[3]) return 0;}else if(C3!=0) return 1;}
}
else if(plc==4){
if(order==1){
mod_XY(7);
  if(C4==Ca) {if(x_[4]!=x[7] || y_[4]!=y[7]) return 0;}else if(C4!=0) return 1;}
else if(order==2){
  if(C1==Ca) {if(x_[1]!=x[7] || y_[1]!=y[7]) return 0;}else if(C1!=0) return 1;}
else if(order==3){
  if(C5==Ca) {if(x_[5]!=x[7] || y_[5]!=y[7]) return 0;}else if(C5!=0) return 1;}
else if(order==4){
  if(C2==Ca) {if(x_[2]!=x[7] || y_[2]!=y[7]) return 0;}else if(C2!=0) return 1;}
else if(order==5){
  if(C3==Ca) {if(x_[3]!=x[7] || y_[3]!=y[7]) return 0;}else if(C3!=0) return 1;}
else if(order==6){
  if(C7==Ca) {if(x_[7]!=x[7] || y_[7]!=y[7]) return 0;}else if(C7!=0) return 1;}
}
}/**else(flag)**/

order++;if(order==6) return 0;
}

return -1;
}/** nv **/



long get_len(int x1,int y1,int x2,int y2)
{
int dx,dy;

dx=x2-x1;
dy=y2-y1;

return /*sqrt*/(dx*dx+dy*dy-dx*dy);
}/** get_len **/



int check_len(int *nx,int *ny)
{
int i,j,k,m,dx,dy,lr,Li,Lj;
long len[3];

Li=1;Lj=2;

/*999*/
```

```
for(j=0;j<Lj;j++)
for(i=0;i<Li;i++){
dx=(RESO-1)*i;
dy=(RESO-1)*j;

/* left */
if(pixel[dx][dy]!=0 && pixel[dx+RESO-1][dy+RESO-1]!=0 && pixel[dx][dy+RESO-1]!=0){
for(k=0;k<CPMAX;k++){
if(nx[k]>=dx && ny[k]<=dy+RESO-1 && ny[k]>=nx[k]-dx+dy) ;
else goto right;
}

lr=0;
goto next;
}

/* right */
right:
if(pixel[dx][dy]!=0 && pixel[dx+RESO-1][dy+RESO-1]!=0 && pixel[dx+RESO-1][dy]!=0){
for(k=0;k<CPMAX;k++){
if(nx[k]<=dx+RESO-1 && ny[k]>=dy && ny[k]<=nx[k]-dx+dy) ;
else goto next_for;
}

lr=1;
goto next;
}

next_for:
;
}/**for(i)**/

return 0;

next:
for(k=0;k<CPMAX;k++){
if(k<CPMAX-1) m=k+1;else m=0;
len[k]=get_len(nx[k],ny[k],nx[m],ny[m]);
}

if(len[0]==len[1] && len[1]==len[2]){
/*printf(" i=%d j=%d\n",i,j);*/
if(RESO%3==1){
if(lr==0) {dx=RESO/3+(RESO-1)*i;          dy=RESO-(RESO/3+1)+(RESO-1)*j;}
else      {dx=RESO-(RESO/3+1)+(RESO-1)*i;dy=RESO/3+(RESO-1)*j;}
```

```
for(k=0;k<CPMAX;k++)
len[k]=get_len(dx,dy,nx[k],ny[k]);

if(len[0]==len[1] && len[1]==len[2]) return 1;
else return 2;
}/**if(RESO%3)**/
else return 1;
}/**if(len[]==len[])**/
else return 2;
}/** check_len **/


int cag_r(void)
{
int i,flag_us,dy;
int flag_[CPMAX],flag_pp[CPMAX],acolor[CPMAX];
int nx[CPMAX],ny[CPMAX],nx_[CPMAX],ny_[CPMAX],nax[CPMAX],nay[CPMAX];
int XDP,YDP,nx_old[CPMAX],ny_old[CPMAX];
int cp,ssize;
int ca,c1,c2,c3,c4,c5,c7;
int nxp,nxm,nyp,nym;
int jmp[8],jp,tmp;
double val,angle;

if(1){
/* u *//*pixel[0][0]=-1;pixel[0][yt]=-1;*/
/* 3 *//*pixel[RESO-1][0]=-1;pixel[RESO-1][yt]=-1;*/
/* 2 */pixel[0][RESO-1]=-1;
/* 1 *//*pixel[RESO-1][RESO-1]=-1;*/
}
if(RESO%3==1){
dy=RESO-1;
/*XDP=RESO/3;YDP=RESO-(RESO/3+1);pixel[XDP][YDP]=-1;*/
XDP=RESO-(RESO/3+1);YDP=RESO/3;pixel[XDP][YDP]=/*12*/-1;
/*XDP=RESO/3;YDP=RESO-(RESO/3+1)+dy;pixel[XDP][YDP]=13;
XDP=RESO-(RESO/3+1);YDP=RESO/3+dy;pixel[XDP][YDP]=14;*/
}

ssize=sizeof(ss);
cp=CPMAX;

acolor[0]=9;acolor[1]=10;acolor[2]=11;
if(CPMAX==12){
                                        acolor[3]=20;acolor[4]=21;acolor[5]=22;
acolor[6]=12;acolor[7]=13;acolor[8]=14;acolor[9]=23;acolor[10]=24;acolor[11]=25;
}
```

```
for(i=0;i<CPMAX;i++){
flag_[i]=1;
}

for(i=0;i<CPMAX;i++)
fp_mem[i]=0;

ca=15;
Ca=ca;

nax[0]=0+2     ;nay[0]=RESO-1-2;
if(combination==0){
}
else{
nax[1]=0+4     ;nay[1]=RESO-1+2;
nax[2]=0+2     ;nay[2]=RESO-1+4;
}

if(CPMAX==12){
}

i=0;
while(1){
if(flag_[i]){                        /* CP_? */
ig=i;

nx[i]=nax[i];ny[i]=nay[i];
putpixel_(nx[i],ny[i],acolor[i]);
}/**if(flag_[i])**/

i++;if(i==CPMAX) break;
}/**while(1)**/

i=0;
while(1){
if(flag_[i]){                        /* CP_? */
nx_[i]=nax[i];ny_[i]=nay[i];
ig=i;

if(combination==0){
/* CW */
}/**if(combination)**/
else{
/* CCW */
if(drn==4) {/* 217 */
```

```
     if(i%3==0) {nax[i]++;}
else if(i%3==1) {nax[i]++;nay[i]++;}
else if(i%3==2) {nay[i]++;}
}
}/**else(combination)**/

nx[i]=nax[i];ny[i]=nay[i];

putpixel_(nx[i],ny[i],acolor[i]);
}/**if(flag_[i])**/

i++;if(i==CPMAX) break;
}/**while(1)**/

/***************************** while(cp) -> ******************************/

flag_us=0;

while(cp){                  /* 2pie/3 */
kbhit_();
if(refill==0) break;

algo=random_(2);

i=0;
while(1){

if(flag_[i]){                       /* CP_? */
ig=i;

nxp=nx[i]+1;nyp=ny[i]+1;nxm=nx[i]-1;nym=ny[i]-1;
c1=getpixel_(nx[i],ny[i],nxp,ny[i]);
x[1]=X;y[1]=Y;jmp[1]=jmpflag;
c2=getpixel_(nx[i],ny[i],nx[i],nyp);
x[2]=X;y[2]=Y;jmp[2]=jmpflag;
c3=getpixel_(nx[i],ny[i],nxm,ny[i]);
x[3]=X;y[3]=Y;jmp[3]=jmpflag;
c4=getpixel_(nx[i],ny[i],nx[i],nym);
x[4]=X;y[4]=Y;jmp[4]=jmpflag;
c5=getpixel_(nx[i],ny[i],nxp,nyp);
x[5]=X;y[5]=Y;jmp[5]=jmpflag;
c7=getpixel_(nx[i],ny[i],nxm,nym);
x[7]=X;y[7]=Y;jmp[7]=jmpflag;

Nx=nx[i];Ny=ny[i];Nxp=nxp;Nyp=nyp;Nxm=nxm;Nym=nym;
C1=c1;C2=c2;C3=c3;C4=c4;C5=c5;C7=c7;
```

```
if((c1==ca)||(c2==ca)||(c3==ca)||(c4==ca)||(c5==ca)||(c7==ca)){
s.xx=nx[i];s.yy=ny[i];s.xx_=nx_[i];s.yy_=ny_[i];fwrite_mem(i);

std_x=nx[i];std_y=ny[i];            /* S */
last_x=nx_[i];last_y=ny_[i];
nx_[i]=nx[i];ny_[i]=ny[i];          /* new b */



if(algo==0){
val=-5;
if((/*c1!=ca*/nv(0,c1,1)==1)&&(c5==ca)){  /* CW (no phase) */
  if((angle=getangle(nxp,nyp))>val) {val=angle;nx[i]=x[5];ny[i]=y[5];jp=jmp[5];}}
if((/*c5!=ca*/nv(0,c5,5)==1)&&(c2==ca)){
  if((angle=getangle(std_x,nyp))>val) {val=angle;nx[i]=x[2];ny[i]=y[2];jp=jmp[2];}}
if((/*c2!=ca*/nv(0,c2,2)==1)&&(c3==ca)){
  if((angle=getangle(nxm,std_y))>val) {val=angle;nx[i]=x[3];ny[i]=y[3];jp=jmp[3];}}
if((/*c3!=ca*/nv(0,c3,3)==1)&&(c7==ca)){
  if((angle=getangle(nxm,nym))>val) {val=angle;nx[i]=x[7];ny[i]=y[7];jp=jmp[7];}}
if((/*c7!=ca*/nv(0,c7,7)==1)&&(c4==ca)){
  if((angle=getangle(std_x,nym))>val) {val=angle;nx[i]=x[4];ny[i]=y[4];jp=jmp[4];}}
if((/*c4!=ca*/nv(0,c4,4)==1)&&(c1==ca)){
  if((angle=getangle(nxp,std_y))>val) {val=angle;nx[i]=x[1];ny[i]=y[1];jp=jmp[1];}}
if(val==-5) /*beep(10000)*/;
}
else{
val=5;
if((/*c1!=ca*/nv(1,c1,1)==1)&&(c4==ca)){  /* CCW (no phase) */
  if((angle=getangle(std_x,nym))<val) {val=angle;nx[i]=x[4];ny[i]=y[4];jp=jmp[4];}}
if((/*c4!=ca*/nv(1,c4,4)==1)&&(c7==ca)){
  if((angle=getangle(nxm,nym))<val) {val=angle;nx[i]=x[7];ny[i]=y[7];jp=jmp[7];}}
if((/*c7!=ca*/nv(1,c7,7)==1)&&(c3==ca)){
  if((angle=getangle(nxm,std_y))<val) {val=angle;nx[i]=x[3];ny[i]=y[3];jp=jmp[3];}}
if((/*c3!=ca*/nv(1,c3,3)==1)&&(c2==ca)){
  if((angle=getangle(std_x,nyp))<val) {val=angle;nx[i]=x[2];ny[i]=y[2];jp=jmp[2];}}
if((/*c2!=ca*/nv(1,c2,2)==1)&&(c5==ca)){
  if((angle=getangle(nxp,nyp))<val) {val=angle;nx[i]=x[5];ny[i]=y[5];jp=jmp[5];}}
if((/*c5!=ca*/nv(1,c5,5)==1)&&(c1==ca)){
  if((angle=getangle(nxp,std_y))<val) {val=angle;nx[i]=x[1];ny[i]=y[1];jp=jmp[1];}}
if(val==5) /*beep(10000)*/;
}

if(jp==1){
ny_[i]=yt-ny_[i];
}
```

```
else if(jp==2){
ny_[i]=yt-ny_[i];
}
else if(jp==3){
ny_[i]=yt-ny_[i];
}

if(1) putpixel_(nx[i],ny[i],acolor[i]);
flag_pp[i]=1;
}/**if(c1,c2,c3,c4)**/
else{
if(ftell_mem(i)==0) {flag_[i]=0;cp--;if(cp==0) break;}
fread_mem(i);
nx[i]=s.xx;ny[i]=s.yy;nx_[i]=s.xx_;ny_[i]=s.yy_;
flag_pp[i]=0;
}/**else(c1,c2,c3,c4)**/
}/**if(flag_[i])**/

i++;if(i==CPMAX) break;
}/**while(1)**/

if(0){
i=0;
while(1){
ig=i;
if(flag_[i]==1 && flag_pp[i]==1) putpixel_(nx[i],ny[i],acolor[i]);

i++;if(i==CPMAX) break;
}/**while(1)**/
}

if(flag_us==0 && flag_[0]==1 && flag_pp[0]==1 && check_len(nx_old,ny_old)>1){
printf(" ?\n");
i=0;putpixel(nx_[i],ny_[i],12);
i=1;putpixel(nx_[i],ny_[i],12);
i=2;putpixel(nx_[i],ny_[i],12);
use_subroop();flag_us=1;
}
}/**while(cp)**/



return 0;
}/** cag_r **/
```