CHAPTER : II

MATHEMATICAL GRAPHICS

CHAPTER -II:: MATHEMATICAL GRAPHICS ::

2.1 INTRODUCTION :

Galileo suggested that "Mathematics is the language of God". Later on in seventeeth century, a philosopher rejected this suggestion because there was no way for him to explain fluid dynamics without the aid of differential equations. Some research workers have suggested that the computational psychology[6] is philosophically interesting. But this is also rejected later on. Number crunching or solving mathematical equations is assumed to be the efficient activity of computer. No art or no creativity is supposed to be involved in it. But lot of highly instructive and exciting geometrical shapes can be generated on the screen of the computer under the control of mathematical formula. This branch of computer graphics which deals with the generation of geometrical shapes is known as 'Matnematical Graphics'.

Some of the interesting and appealing envelopes have been developed on the screen of computer by Dr.S.Balchandra Rao[4] . His choice of language was BASIC. 'Spectrum' was the computer on which he executed the program. Though spectrum has low resolution in graphics mode, the patterns look correct (i.e. according to the mathematical equations) and attractive. The following envelopes are developed by Dr.Rao

1 Parabola as an envelope of straight lines.

2 Lemniseateas envelope of circles.

3 Cardioid as envelope of circles.

4 Parabola of safety.

5 Nephroid as envelope of circles.

2.2 A DICTIONARY OF MATHEMATICAL SHAPES :

In this work, a set of programs are developed that produce attractive shapes under the control of mathematical formula.The mathematical shapes have certain parameters that can be varied and change the apperance of the shape e.g. for a sine wave the frequency or amplitude are the variable parameters. These interactive programs demand from user, the required parameters and present the user with the geometrical pattern.

Following mathematical shapes are developed in this chapter.

1 Linear spirals.

2 Concentric lobe patterns.

3 Contra spirals.

4 Amplitude modulation of sine wave.

5 Lissajous figures.

3.3 FEATURES OF THE SHAPE :

The mathematical functions used in this work are mainly 'circular' or harmonic functions. This means that although for example, a spiral formula is used to generate a particular shape, it may well be possible to generate exactly the same shape using Lissajous formula. One of the fascinating features of mathematical graphics is that unless you are an expert mathematician, the outcome of using a certain formula with particular parameters is unknown at the outset. There is literally an infinite number of possible variations.

The area of mathematical graphics is potentially very rewarding. It is rather like composing music. The computer is your musical instrument and the mathematical formulae are the rules of harmony, on which the type or structure of the music depends. You are limited only by your imagination and can experiment freely with the variable parameters of each shape.

Implementation of fuzzy graphs and fuzzy sphere[4] will be an extension of this work. Theorems are deduced for the folding of the fuzzy graphs and fuzzy spheres. They have many applications in a communication switching networks with several subscriber units. Another direction of extension of the work in this chapter is three dimensional manipulations. If you have enough generation time then you can generate pictures that imitate reality to a degree of accuracy that is uncanny. The mathematical shading models are now so accurate that when a three dimensional solid is displayed on a screen of sufficiently high resolution it is almost impossible to tell if the resulting display is a real image, from a colour television camera, or a program generated model. We can draw pseudo three dimensional spheres by drawing ellipses. Here two loops are required. One draws vertically aligned ellipses that get fatter and fatter and end up as a circle. Another loop controls the drawing of horizontally aligned ellipses overlapped on vertically aligned ellipses. The net result is the figure giving impression of a sphere.

2.4 MATHEMATICAL GRAPHICS- APPLICATIONS

Individual mathematical shapes implemented here can be used as motifs in the decorative techniques and they can be combined in any way you choose. One can locate these motifs on wall-paper, fabrics, carpets or porcelain ornamentation. McGregor J. and Watt A.[3] have discussed the seven frieze or band groups and 17 wallpaper groups. These seven frieze groups and 17 wallpaper groups specify all the possible ways in which one can arrange motifs such that the final pattern exhibits certain kind of symmetry.

The first symmetry class of band patterns are with one translation axis and no other type of symmetry. In the second band group, there is a glide reflection plane or mirror glide plane. A glide reflection is a combination of reflection plus translation. In the third symmetry class the motifs possesses axial symmetry of order 2 i.e. the elementary motif coincides with itself after rotation through 180°. Band group 4 involve mirror symmetry planes. In band group 5 mirror plane and translation axis are parallel. Band group 6 is obtained by taking the fourth class and adding a mirror plane parallel to the translation axis. Band group seven has both a vertical axis of symmetry and a transverse mirror plane. Some of these motifs happened to tessellate[3] the plane. This gives rise to a very interesting and attractive overall shape. It is observed that people have always been fascinated by shapes or tiles that fit together to cover a plane surface leaving no gaps between the tiles. Martin Gardner[3] wrote in his column in 'Scientific American' about the polygons that would tessellate the plane. This gives rise to a sudden boost in interest of number of amateur mathematicians into generating descriptions of previously unknown pentagonal shapes which could be used in such tilings (quoted in reference 3). M.C. Escher[3] has also worked on tilings of the plane with shapes that resembled animals and birds. Many traditional decorations and crafts involve tessellations. Appropriate orientation (rotation, translation, scaling etc.) must be applied to individual tile before being connected to other tiles.

Mathematical graphics has received considerable attention as a sophisticated tool for instruction and explorative education in mathematics. This application belongs to the subject area

'Computer Assisted Instruction' (CAI) which is a subbranch of Artifical Intelligence. The application of mathematical graphic in CAI gives rise to a new branch of graphics known as 'Turtle Graphics'[5] . Since 1970 , LOGO group at MIT AI lab is working on 'Turtle Graphics'. This graphics is mainly developed by Seymour, Papert, Harold Abelson, and others at the MIT AI laboratory.[5] This discipline is a dynamic and fertile approach to mathematics that studies geometrical objects and shapes, interms of the exploratory behaviour of mathematical "animals" called turtles. A turtle is an active agent whose movements create traces on the screen that can describe any kind of shape or object.

Some geometrical shapes are very difficult to draw on a paper. But one can simplify the process by observing the sequence of drawing on a screen of a computer. This is possible in fact, if proper delay is set after each iteration in main plotting loop.

2.5 SPIRALS :

If a line rotates in a plane about one of it's ends and if at the same time, a point moves along the line continuously in one direction, the curve traced out by the moving point is called spiral. The point about which the line rotates is called as pole. The general equation of the spiral is given by

$$r= k \theta \qquad 2.1$$

where r and θ are radius and angle respectively.

Radius vector of a spiral is the line joining any point on the curve with the pole. Angle between radius vector and the line in it's initial position is called the vectorial angle. Each complete revolution of the curve is termed as the convolution. A spiral may make any number of revolutions before reaching the pole. Spiral is one of the basic model for the growth of biological forms.

2.5.1 ARCHIMEDEAN SPIRAL[1] :

It is a curve traced out by a point moving in such a way that it's movement towards or away from the pole is uniform with the increase of the vectorial angle from the straight line.

The use of this pattern is made in teeth profiles of helical gears, profiles of coms etc.

2.5.2 LOGARITHMIC OR EQUIANGULAR SPIRALS[1]:

In a logarithmic spirals the ratio of the lengths of consecutive radius vectors enclosing equal angles is always constant. In other words, the values of vectorial angles are in arithmatic progression and the corresponding values of radius vector are in geometrical progression. The logarithmic spirals are also calld as equiangular spiral because of it's property that the angle which the tangent of any point on the curve makes with it's radius vector at that point is constant.

Archimedean spiral is implemented in this work. The x and y cooordinates of Archimedean spirals are defined as,

$$x = R_x \theta \ \mathrm{Cos}\ (\theta)$$
$$y = R_y \ \theta \ \mathrm{Sin}\ (\theta) \qquad 2.2$$

where R_x and R_y are the radii of the spiral.

If R_x and R_y are not equal then plot of formulae of equation (2.2) enables the user to get an increasing elliptical shape If $R_x = R_y$ then circular spirals are produced.

2.5.3 CONCENTRIC OR LINEAR SPIRALS :

The basic equation used to draw concentric or linear spirals are

$$x = r \ \mathrm{Cos}\ (\mathrm{ang})$$
$$y = r \ \mathrm{Sin}\ (\mathrm{ang}) \qquad 2.3$$

where r = radius

The radius is modulated by some factor. Basically the equations are specifically for circles. But increasing the radius of circle produces spirals. Incrementing the angle by some factor will rotate the spirals and produce this pattern.

2.5.4 PROGRAM DESCRIPTION :

By using include directive, the contents of the necessary files are included in the program. Considering the screen coordinates, origin of the pattern is defined at the centre of the screen. The initial radius is defined by a constant R .A function is defined for plotting on the screen by the statement void (float, float).

In the declaration module of the program, all the declarations of the variables used in program is done. In the 'get info' module graphics mode is initialized.

In the main loop radius r of the pattern is increased by an equal increment "INC" for each angular increment of 60°. In the main loop each time x and y cordinates are updated. Also message "SPIRAL" is displayed on the screen as soon as the drawing completes. Function for plotting is given at the end of the program. For plotting aspect ratio of the screen is considered.

Provision for taking the hardcopy of the pattern is kept, Pressing the key,'P' will give hardcopy. For continuation of the pattern, just press any key. To exit you have to press the key 'q'. Refer to listing of program on page number 45.

2.6 CONCENTRIC LOBE PATTERN :

More realistic and beautiful designs can be produced by modifying the program for linear spirals. This pattern looks like the flower with petals. After careful observation of flower, you will see that the petals are arranged in a spiral like pattern around the centre.

Here the basic principle is the same i.e. modulation of radius of circle.

The formula used for modulation of radius is,

$$r = R \times \text{Cos } (i \times \frac{\pi}{180} \times f_m) \qquad 2.4$$

where r = new radius

R = Initial radius

f_m = modulation factor.

i = local variable in the loop.

Rest of the formulae are same as that of the linear spiral programs.

2.6.1 PROGRAM DESCRIPTION :

With comparison to linear spiral program, the only additional thing in the declaration module is a declaration of pointer which is necessary for allocating the memory. In the 'get info' module of the program, major initialization of the graphics mode is done and it asks user, the modulation factor. For the modulation factor less than or greater than 1, different type of patterns are produced. This modulation factor is related to the variable repfactor which decides the number of lobes of the pattern.

In the 'initial' module the calculation of 'repfactor' and 'step' is done. There are three cases for the modulation factor (f_m), greater than 1, between 0.5 and 1, less than 1. When modulation factor is greater than 1 the repfactor has assigned the vlaue 377 (considering the coordinates of the screen) and 'Step' is taken as inverse of modulation factor. For modulation factor greater than 0.5 and less than 1, the 'repfactor' is equal to

$$\text{repfactor} = 377 \text{ X modulation factor X } 10 \qquad 2.5$$

and step is taken as unity.

For modulation factor less than 0.5, repfactor is taken as,

$$\text{refactor} = 377 \text{ X } (1-f_m) \text{ X } 10 \qquad 2.6$$

and step is taken as unity.

Calculation of step is clearly made on the basis of convenience. It decides the density of the pattern. For modulation factor greater than 1, there is a possibility of much dense and unattractive pattern. So it is reduced, below to $1/f_m$.

In the main loop the major implementation is modulation of radius for which equation (2.4) is used. 'i' is the floating variable used for angle and is varied from 0 to 'repfactor' by the value of step.

In the 'out' module of the program a message "**MODULATION OF RADIUS**" is displayed on the screen. The value of the modulation factor is also displayed for establishing the relation of modulation factor value with the pattern produced. Options for exiting and printing are also displayed on the screen at proper locations. Last part of the program is the function for plotting. Refer to listing of program on page number 47.

2.7 SPIRALS AND COUNTERSPIRALS (Contra-rotating spirals)

Contra-rotating spirals have a valuable significance in nature.Contra-rotating spirals are often come across in nature as seed patterns, for example in fir cones and sunflowers. Actually the natural pattern has different number of clockwise and counterclockwise spirals. Commonly these are, 8, 13 or 13,21 or 21,34 and these pairs of numbers are terms in a mathematical series, known as the Fibonacci series.

2.71 SOFTWARE IMPLEMENTATION :

The program resembles the linear spirals except for some differences in main loop. The radius is modulated by factor 'inc'. First a spiral is generated with following set of equations,

$$\begin{aligned} x &= r \text{ Cos } (ang) \\ y &= r \text{ Sin } (ang) \end{aligned} \tag{2.7}$$

Then a connecting antispiral is generated by taking the value of angle same as that of the spiral but with negative sign i.e.

$$x = r \text{ Cos } (- ang)$$
$$y = r \text{ Sin } (- ang) \qquad 2.8$$

The usual things of updating x and y coordinate values and plotting are done as in the other programs.

The 'out' module gives an output the measage as 'Spiral and counter Spiral' on the screen, after the completion of the drawing. It also outputs other options. It is very exciting to watch the building pattern.
Refer to listing of program on page number 50.

2.8 MODULATION OF SINE WAVE :

Sine function is given by,

$$y = \text{Sin } \partial \qquad 2.9$$

The variable parameters in this function are the amplitude or height of the wave and the frequency or the rate of modulation. The equation depicting the variable parameters is,

$$y = S \text{ Sin } (f. \, \theta) \qquad 2.10$$

where S = amplitude of the wave

f = frequency of the wave

2.8.1 AMPLITUDE MODULATION :

Sine wave is a periodic function i.e. it repeats itself with certain frequency. If S, in the equation (2.10) is replaced by constant, with a function, then the amplitude of the sine wave will vary. So replacing S, with another sine wave, the equation (2.10) becomes,

$$y = S\ \mathrm{Sin}\ (f\ 1\ \theta)\ \mathrm{Sin}\ (f\ 2\ \theta) \qquad 2.11$$

2.8.2 SIMULATION OF AMPLITUDE MODULATION :

The 'get info' module of the program prompts the user to give modulation factor and time factor. Time factor indicates the rate of modulation, while the modulation factor indicates the amount of modulation.

The 'initial' module of the program calculates the value of step which is further used in the program. If modulation factor is greater than or equal to 1 (i.e. excessive) then the step is inverse of the modulation factor. For all other cases the step is equal to one.

The function of "MAIN" loop is to plot the pattern. The pattern of amplitude modulation starts from the extreme left side of the screen and ends at the rignt side. The basic equations used are as follows,

$$\begin{aligned} x &= i \\ y &= r \sin (ang) \end{aligned} \qquad 2.12$$

In the first iteration x is zero and increases by step (the value of which depends upon the modulation factor) in each iteration . The maximum value of 'i' is chosen by considering screen coordinate. The priciple of radius modulation is also used here. The equation for radius is

$$r = R \text{ Cos } (t \times i \times \frac{\pi}{180} \times f_m) \qquad 2.13$$

where R = initial radius

t= time factor.

i = local variable of the main loop.

f_m = modulation factor.

So amplitude of equation will depend on the modulation factor and the rate of which depends on t.

The value of 'ang' in equation (2.12) is given by,

$$ang = t \times i \times \frac{\pi}{180} \qquad 2.14$$

So there is no effect of modulation factors on equation (2.12) but time factor has some effect.

The 'out' module of the program displays the message 'MODULATION OF SINE WAVE" and also gives the modulation factor entered by the user. And also it displays how to quit and take the hardcopy of the pattern.

Refer to listing of program on page number 52.

2.9 LISSAGJOUS PATTERNS :

In some foregin languages, lissajous means "pretty picture". This amazing graphics program transforms PC into a laser light show. Two sine waves of equal frequency produce a Lissajous pattern which may be a straight line, an ellipse or a circle depending on the phase, amplitude and freauencv of two siqnals. Mathematicallv it can be exoressed as.

$$x = S1 \ \mathrm{Sin} \ (\theta)$$
$$y = S2 \ \mathrm{Sin} \ (O + \theta) \qquad 2.15$$

where o is the order of the oattern

making o equal to 0,1.....6 and controlling produces various patterns e.g. if O=1 , the freauency of the sine waves is eaual and now if has assigned a value of 90° then circle results. Lissagious figure appears on the screen of C.R.O. when sinusoidal voltages are simultaneously applied to horizontal and vertical plates A numher of conclusions can he drawn from these patterns.

Now, if for θ = 90 and varying the freauencies then the equation we get,

$$x = S1 \ \mathrm{Sin} \ (o \ 1 \ \theta) \qquad 2.16$$
$$y = S2 \ \mathrm{Cos} \ (o \ 2 \ \theta)$$

Now making S itself a sine function, an' astonishing variety of patterns result

2.9.1 PROGRAMME IMPLEMENTATION :

The 'get info' module of the program first displays the equations implemented in the program. This facility is intentionally kept so that it would be easier for a user to choose the parameters properly. There are four parameters to

12345
A

choose. If they are not chosen properly the major portion of the pattern is formed outside the screen and only lines appear on the screen. The module also prompts the user to give the required parameters to form the pattern. The following parameters are to be provided by the user viz. radius modulation factor, x modulation factor, frequency ratio and phase.

The 'initial' module of the program calculates repfactor (which decides the number of lobes) and step value (the separation between lobes). There are four distinct cases. The modulation factor is taken as 377 (value depends on the screen coordinates) and step value is inverse of radius modulation factor multiplied by frequency ratio. For radius modulation factor greater than 0.5.

repfactor = 377 X radius modulation fact. X 10 2.17

and step = 1

If modulation factor is less than 0.5 then

repfactor = 377 X (1- radius modulation factor) X 10 2.18

and step remains to be unity.

A special case for zero radius modulation factor is treated here. The repfactor for this case is taken as 380 and the step is taken as inverse of the frequency ratio.

The following equations are implemented in main loop module, x= radius X modulation factor along X axis X cosine (angle)

y= radius X Sine (frequenccy ratio X angle + 2.19

phase difference)

radius = constant radius X cosine (modulation factor X angle)

and

$$\text{angle} = i \times \frac{\pi}{180} \qquad 2.20$$

From i equal to zero upto repfactor the loop executes incrementing i by 'step' each time. Plotting of x and y coordinates and updating of x and y is also done in main loop .

Last part of the program is the plot function which plots the pixel considering aspect ratio of the screen.
Refer to listing of program on page number 56.

2.10 POSSIBLE FUTURE EXTENSIONS :

Continuing the same line, one can implement cardioids squirals, cycloids

The general equation of the cardioid is

$$r = a\ (1- \text{Cos}\ (o\ \phi) \qquad 2.21$$

where r is the radius, 0 is the order and is the angle.

By rotating and shrinking the square shapes an effect of spiral can be produced. This pattern is called squiral.

A cycloid is the curve traced out by a point on a circle as it rolls along another curve. One can obtain a cycloid by rolling a circle along a straight line. Cycloid gives astonishing variety of patterns, like the lissajous pattern. In lissajous pattern generator , there is less control over the design of the pattern. But in case of cuycloid, you can control the design of the pattern very successfully.

One can use the random number generation function, to draw the mathematical shapes with random position, random size and random colour.These shapes sometimes, appear in the form of natural patterns such as rosettes. But a totally different technique of fractls is used for drawing natural scenery. Third chapter of this dissertation deals with such techniques.

```
 /*LINEAR SPIRAL  PROGRAM LISTING piral.c*/
 /* DATE:  9-6-1994        */
/*INCREMENTING  THE  RADIUS OF CIRCLE  PRODUCES*/
/*   SPIRALS.ROTATING  THE SPIRALS  WILL  PRODUCE
/*   THIS DESIGN.                             */

/*  ++++++++ PROGRAM  STARTS  HERE +++++++++++*/

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <graphics.h>
#include <math.h>


#define R 100
#define XC 350
#define YC 240


 /*   functions  MODULE*/
void plot(float ,float );

double pi;

int xasp;
int yasp;

main()
æ
  float x,y;
  float r;
  float fm;
  float temp;
  double ang;
  double step;
  float inc;

  int drive = HERCMONO;
  int mode = DETECT;

  float i;

  pi = 22/7;

  inc = .6;

 /* get info  MODULE*/

 initgraph(&drive,&mode,"");

 getaspectratio(&xasp,&yasp);
```

```
/******* MAIN  LOOP   STARTS HERE ********* */

for(step = 0; step < 360 ; step += 60)
    for(i=0,r=0 ; i<377 ; i++)
       æ
          ang = (i+step) * pi/180;

          r  =  r + inc;
              /* modulation of radius */

          x = r*cos(ang);
          y = r*sin(ang);

          x = x + XC;
          y = y + YC;

          plot(x,y);
         /*plots considering aspectratio */

         å
         /* OUT MODULE */

        outtextxy(200,10,"SPIRAL");

        getch();
        closegraph();
         å


       /********** main ends *************/


    /*____ ploting the pixel____________   */

      void plot(float p,float q)
       æ
         q = q * (float)xasp/yasp;
         q = q * 0.95;
         putpixel(p,q,1);
       å
    /*LISTING OF SPYrograms!    SPY.H 8-6-1994*/
    /*  MODULATING  THE  RADIUS OF CIRCLE  */
    /*   THE  PROGRAM WILL ASK THE USER TO  GIVE  THE
       MODULATION  FACTOR.  ACCORDING TO  THE  FACTOR
       GIVEN  PATTERNS  ARE PRODUCED FOR  THE  FACTOR
       LESS THAN OR GREATER THAN 1 DIFFERENT TYPES OF
       PATTERNS WILL BE FORMED*/

/*   ████████  PROGRAM STARTS  HERE  ██████   */

    #include <stdio.h>
    #include <conio.h>
    #include <stdlib.h>
    #include <graphics.h>
    #include <math.h>
    #include "gprnt.h"

    #define R 100
    #define XC 350
    #define YC 240


    /*   functions  */
     void plot(float ,float );

          double pi;
          int xasp;
          int yasp;

      main()
        æ
           float x,y;
           float xcen,ycen;
           float r;
           float fm;
           float temp;
           float dm;
           double ang;
           int repfactor;
           double step;
           char *fchar;
           int dec=2;
           int sign=1;

           int drive = HERCMONO;
           int mode = DETECT;
           char op;
           float i;

           pi = 22/7;
           fchar = malloc(10);

           /* get info  module*/
           doæ
          clrscr();
          gotoxy(20,10);

     printf("give modulation factor :  ");
     scanf("%f",&fm);

     fchar = fcvt(fm,4,&dec,&sign);

     initgraph(&drive,&mode,"");

     getaspectratio(&xasp,&yasp);

      /*  initial    */

      if(fm >= 1)  æ
      repfactor = 377;
      step = (double)1/fm;
                 å
      elseæ
      step = 1;

      if(fm > 0.5) repfactor = 377 * fm * 10 ;
        else  repfactor = 377 * (1 - fm) * 10 ;
           å


/******* MAIN  LOOP   STARTS HERE ******** */
       for(dm = 1; dm < 1.5; dm = dm + .1)
        for(i=0 ; i<repfactor ; i = i + step)
          æ
             ang = i * pi/180;

                  temp = cos(ang*fm);

                    r  =  R * temp  *  dm;
                    /* modulation of radius */

                    x = r*cos(ang);
                    y = r*sin(ang);

                    x = x + XC;
                    y = y + YC;

                    plot(x,y);
           /* ploats considering aspectratio */

             å
             /* for */

      outtextxy(200,10,"MODULATION OF RADIUS");
      outtextxy(200,30,"Modulation factor:");
      outtextxy(200,310,"q - QUIT    p - PRINT ");
      outtextxy(460,30,fchar);
```

```
  op = getch();
  closegraph();

  å while (op != 'q');

   å


/**************** main ends *************/


/* ploting the pixel_______  */

  void plot(float p,float q)
      æ
        q = q * (float)xasp/yasp;
        q = q * 0.95;
        putpixel(p,q,1);
      å

   /*├├├├├├├├├PROGRAM ENDS HERE ├├├├├├├  */
```

```
/*PROGRAM LISTING OF CPRL.C          */
/* THIS PROGRAM PRODUCES CONTRA-SPIRALS          */
/*INCREMENTING THE RADIUS OF CIRCLE PRODUCES SPIRALS*/
/*ROTATING SUCH SPIRALS IN TWO DIRECTIONS WILL PRODUCE
/*THIS DESIGN.              */


  /* ++++++++++++++++++++++++++++++++++++++++++++++++++*/

   #include <stdio.h>
   #include <conio.h>
   #include <stdlib.h>
   #include <graphics.h>
   #include <math.h>
   #include "gprnt.h"

   #define R 100
   #define XC 350
   #define YC 240

   /*   functions  */
   void plot(float ,float );

   double pi;

   int xasp;
   int yasp;

   main()
   æ
      float x,y;
      float r;
      float fm;
      float temp;
      double ang;
      double step;
      float inc;        /* radius deciding factor */
      char op;

      int drive = HERCMONO;
      int mode = DETECT;

      float i;

      pi = 22/7;

      inc = .4;

      /* get info  */

      initgraph(&drive,&mode,"");

      getaspectratio(&xasp,&yasp);
```

```
        /******* MAIN  LOOP   STARTS HERE **********/
          for(step = 0; step < 362 ; step += 30)
       for(i=0,r=0 ; i<387 ; i++)
        æ
             ang = (i+step) * pi/180;

                  r = r + inc;
                   /*modulation of radius*/

                         x = r*cos(ang);
                         y = r*sin(ang);

                         x = x + XC;
                         y = y + YC;

                         plot(x,y);
                       /*plots considering aspect ratio

                         ang = -ang;
                         x = r*cos(ang);
                         y = r*sin(ang);

                         x = x + XC;
                         y = y + YC;

                         plot(x,y);

                        å
                         /* out module */

                outtextxy(200,10,"SPIRAL & COUNTERSPIRAL");
                outtextxy(200,330,"P - PRINTOUT q -QUIT ");

              doæ
                 op = getch();
                  if( (op == 'p')øø(op == 'P') )
                    ghardpict(50,20,550,290);
                     åwhile(op != 'q');
                     closegraph();
                        å


/************************* main ends *************/


 /*____ ploting the pixel________________________  */

     void plot(float p,float q)
             æ
             q = q * (float)xasp/yasp;
             q = q * 0.95;
             putpixel(p,q,1);
             å
```

```
/* PROGRAM LISTING OF MSINE.C  */
/* PROGRAM TO DRAW THE ENVELOPE OF AMPLITUDE  MODULAT
   ED  WAVE. MODULATING  THE  RADIUS OF  CIRCLE.  THE
   PROGRAM WILL ASK THE USER TO GIVE THE  MODUKLATION
   FACTOR.ACCORDING TO THE FACTOR GIVEN PATTERNS  ARE
   PRODUCED.FOR THE FACTOR LESS THAN OR GREATER  THAN
   1 DIFFERENT TYPES OF PATTERNS WILL BE FORMED. */
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <graphics.h>
#include <math.h>


 #define R 100
 #define XC 0
 #define YC 240


 /*   functions   */
 void plot(float ,float );

 double pi;

 int xasp;
 int yasp;

 main()
  æ
   float x,y;
   float xcen,ycen;
   float r;
   float fm;
   float temp;
   double ang;
   int repfactor;
   double step;
   char *fchar;
   int dec=2;
   int sign=1;
   float t;
   char op;

   int drive = HERCMONO;
   int mode = DETECT;

   float i;

   pi = 22/7;
   fchar = malloc(10);

   /* get info  */

   doæ
     clrscr();
```

```
    gotoxy(20,10);
    printf("give modulation factor :  ");
    scanf("%f",&fm);

    gotoxy(20,12);
    printf("give time factor :  ");
    scanf("%f",&t);

    fchar = fcvt(fm,4,&dec,&sign);

    initgraph(&drive,&mode,"");

    getaspectratio(&xaspp,&yasp);

       /*  initial    */

    if(fm >= 1)  æ
        repfactor = 377;
        step = (double)1/fm;
                 å
            elseæ
              step = 1;
           if(fm > 0.5) repfactor = 377 * fm * 10 ;
            else
                repfactor = 377 * (1 - fm) * 10 ;
                 å


   /* MAIN  LOOP   STARTS HERE *********** */
        for(i=0; i<700 ; i = i + step)
           æ
            ang = t * i * pi/180;

               temp = cos(ang*fm);

               r = R * temp;
                  /* modulation of radius */

                x = i;
                y = r*sin(ang);

                x = x + XC;
                y = y + YC;

                plot(x,y);
                /* plots considering aspectratio*/
                  å
             /* out */

outtextxy(200,10,"MODULATION OF SINEWAVE");
outtextxy(200,30,"Modulation factor:");
outtextxy(20,340,"PRESS q to QUIT");
outtextxy(460,30,fchar);
```

```
if(fm == 0) æ
 repfactor = 380;
 step = (double)1/n ;

  å

/******* MAIN  LOOP STARTS HERE **************** */
for(i=0 ; i<repfactor ; i = i + step)
 æ
        ang = i * pi/180;

        temp = cos(ang*fm);

        r = R * temp;  /* modulation of radius */

        x = s * r*cos(ang);
        y = r*sin(n*ang + phase);

        x = x + XC;
        y = y + YC;

        plot(x,y); /* plots considering aspectratio*/

    å
    /* for */

outtextxy(270,10,"LISSAJOS PATTERN");
outtextxy(100,30,"   X   =  radius   x
  modulation factor along X axis  x   cosine(angle)");
outtextxy(100,40,"   Y   =  radius   x
  sine(frequency ratio * angle   +  phase diff)");
outtextxy(100,50,"radius =  Constant radius   x
  cosine( modulation factor * angle) ");


outtextxy(20,310,"q - QUIT            p - PRINT
          any Key to continue");
/*  outtextxy(200,30,"Modulation factor:");
outtextxy(460,30,fchar);  */


op = getch();
if(op == ´p´) ghardpict(1,1,600,300);
closegraph( );

å while(op!=´q´);


å

  /************************** main ends ************/
```

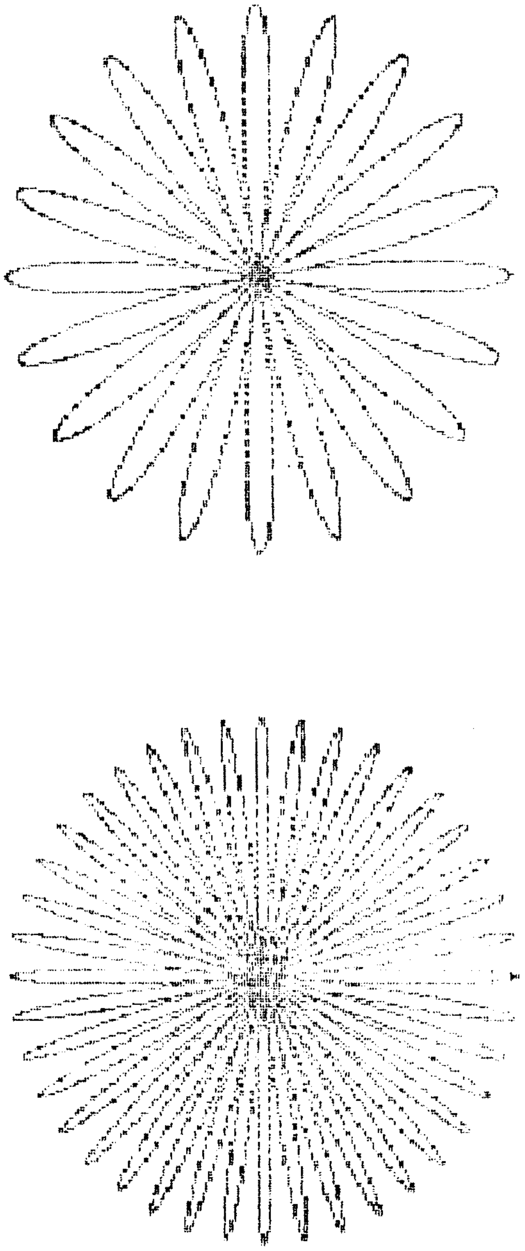
```
/*_____ ploting the pixel____________________________ */

 void plot(float p,float q)
  æ
     q = q * (float)xasp/yasp;
     q = q * 0.95;
     putpixel(p,q,1);
   å
```

```
/*PROGRAM TO DRAW PRETTY LISSAJOUS PATTERNS LISSO.C  */
/*MODULATING  THE  RADIUS OF CIRCLE          */
/*THE PROGRAM WILL ASK THE USER TO GIVE THE MODUKLATION*/
/*FACTOR. ACCORDING TO THE FACTOR GIVEN PATTERNS ARE  */
/*PRODUCED. FOR THE FACTOR LESS THAN OR GREATER THAN 1 */
/*DIFFERENT  TYPES  OF PATTERNS  WILL  BE  FORMED.    */

   /*╚╚╚╚╚╚╚PROGRAM STARTS HERE ╚╚╚╚╚╚  */
   #include <stdio.h>
   #include <conio.h>
   #include <stdlib.h>
   #include <graphics.h>
   #include <math.h>
   #include "gprnt.h"

   #define R 100
   #define XC 350
   #define YC 240


           /*   functions   */
   void plot(float ,float );

   double pi;

   int xasp;
   int yasp;

   main()
   æ
     float x,y;
     float xcen,ycen;
     float r;
     float fm;
     float temp;
     double ang;
     int repfactor;
     double step;
     char *fchar;
     int dec=2;
     int sign=1;
     float n,phase,s;
     char op;

     int drive = HERCMONO;
     int mode = DETECT;

     float i;

     pi = 22/7;
     fchar = malloc(10);
```

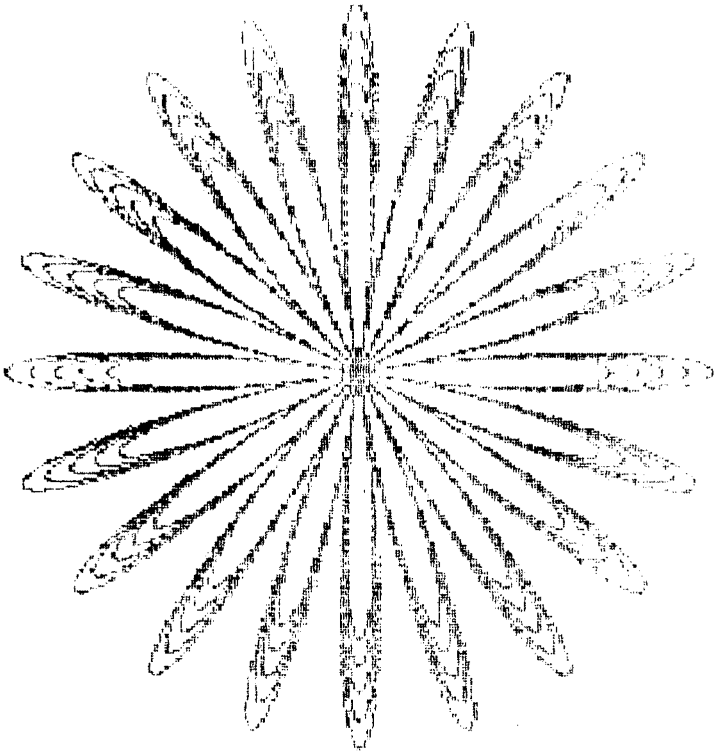
```
  do
  æ
            /* get info  */
clrscr();

gotoxy(30,1);
printf("LISSAJOS PATTERN");
gotoxy(8,3);
printf("X= radius x modulation factor along X axis
          x   cosine(angle)");
gotoxy(8,4);
printf("   Y   =  radius   x   sine(frequency ratio *
      angle   +  phase diff)");
gotoxy(8,5);
printf("radius =  Constant radius   x
          cosine( modulation factor * angle) ");

gotoxy(20,8);
printf("give Radius modulation factor:  ");
scanf("%f",&fm);

gotoxy(20,10);
printf("give x modulation factor     :  ");
scanf("%f",&s);

gotoxy(20,12);
printf("give frequency ratio         :  ");
scanf("%f",&n);

gotoxy(20,14);
printf("give phase                   :  ");
scanf("%f",&phase);


fchar = fcvt(fm,4,&dec,&sign);

initgraph(&drive,&mode,"");

getaspectratio(&xasp,&yasp);

         /*  initial   MODULE */


if(fm >= 1)   æ
 repfactor = 377;
 step = (double)1/(fm*n) ;
  å
         elseæ
                 step = 1;
            if(fm > 0.5) repfactor = 377 * fm * 10 ;
                 else
                  repfactor = 377 * (1 - fm) * 10;
              å
```

```
if(fm == 0) æ
 repfactor = 380;
 step = (double)1/n ;

  å

/******* MAIN  LOOP STARTS HERE **************** */
for(i=0 ; i<repfactor ; i = i + step)
 æ
        ang = i * pi/180;

        temp = cos(ang*fm);

        r = R * temp;  /* modulation of radius */

        x = s * r*cos(ang);
        y = r*sin(n*ang + phase);

        x = x + XC;
        y = y + YC;

        plot(x,y); /* plots considering aspectratio*/

    å
    /* for */

outtextxy(270,10,"LISSAJOS PATTERN");
outtextxy(100,30,"   X   =  radius   x
  modulation factor along X axis  x   cosine(angle)");
outtextxy(100,40,"   Y   =  radius   x
  sine(frequency ratio * angle   +  phase diff)");
outtextxy(100,50,"radius =  Constant radius   x
  cosine( modulation factor * angle) ");


outtextxy(20,310,"q - QUIT              p - PRINT
          any Key to continue");
/*  outtextxy(200,30,"Modulation factor:");
outtextxy(460,30,fchar);  */


op = getch();
if(op == 'p') ghardpict(1,1,600,300);
closegraph();

å while(op!='q');


å

  /************************** main ends ************/
```

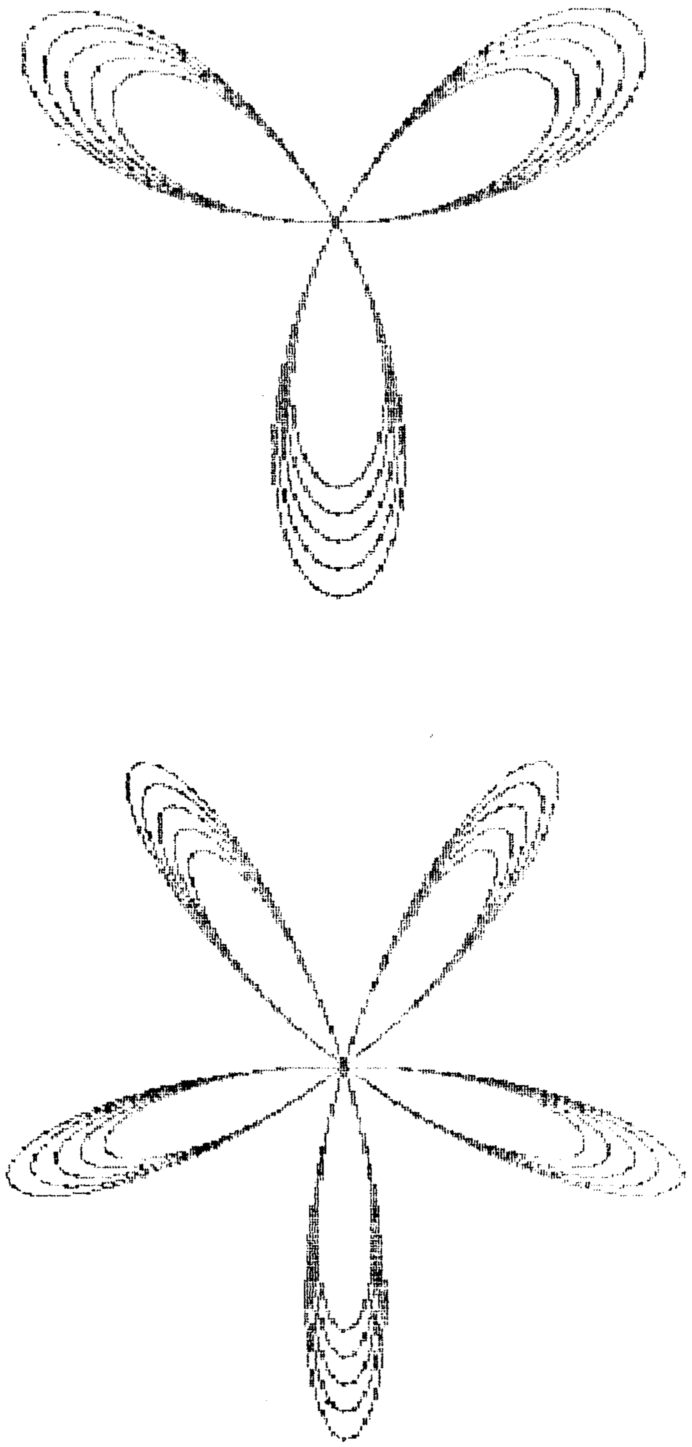
```
/*____ ploting the pixel_______________________ */

 void plot(float p,float q)
  æ
     q = q * (float)xasp/yasp;
     q = q * 0.95;
     putpixel(p,q,1);
   å
```



Printout of output of SPY.C.(A program drawing concentric lobe patterns)

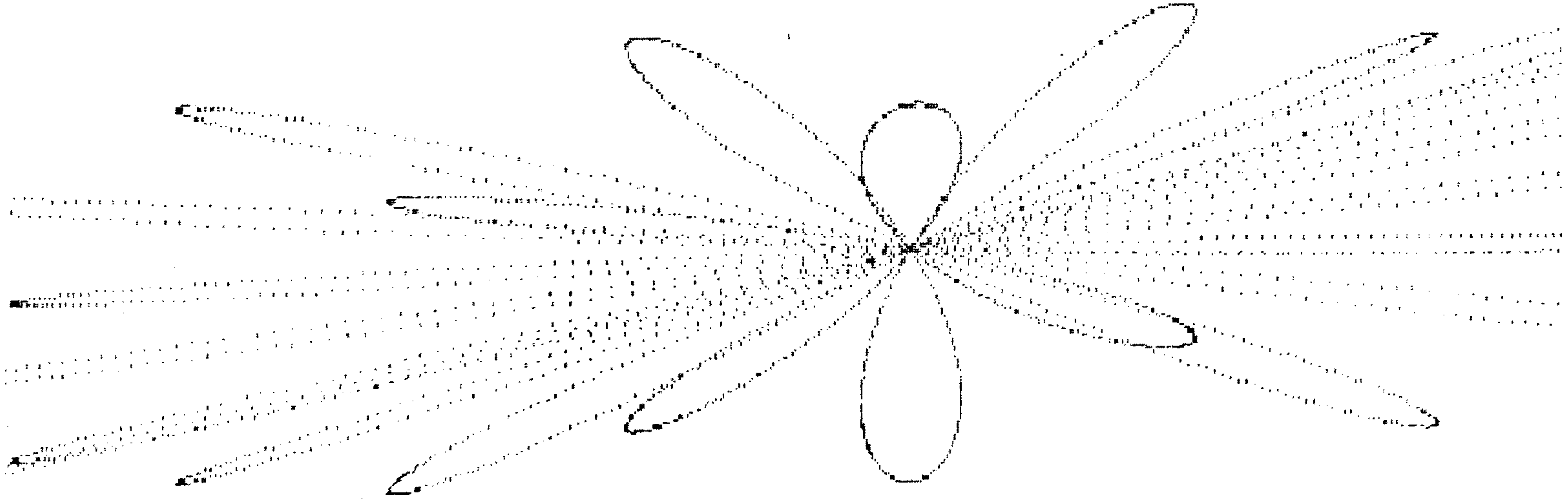


Modulation of spirals with factor 10

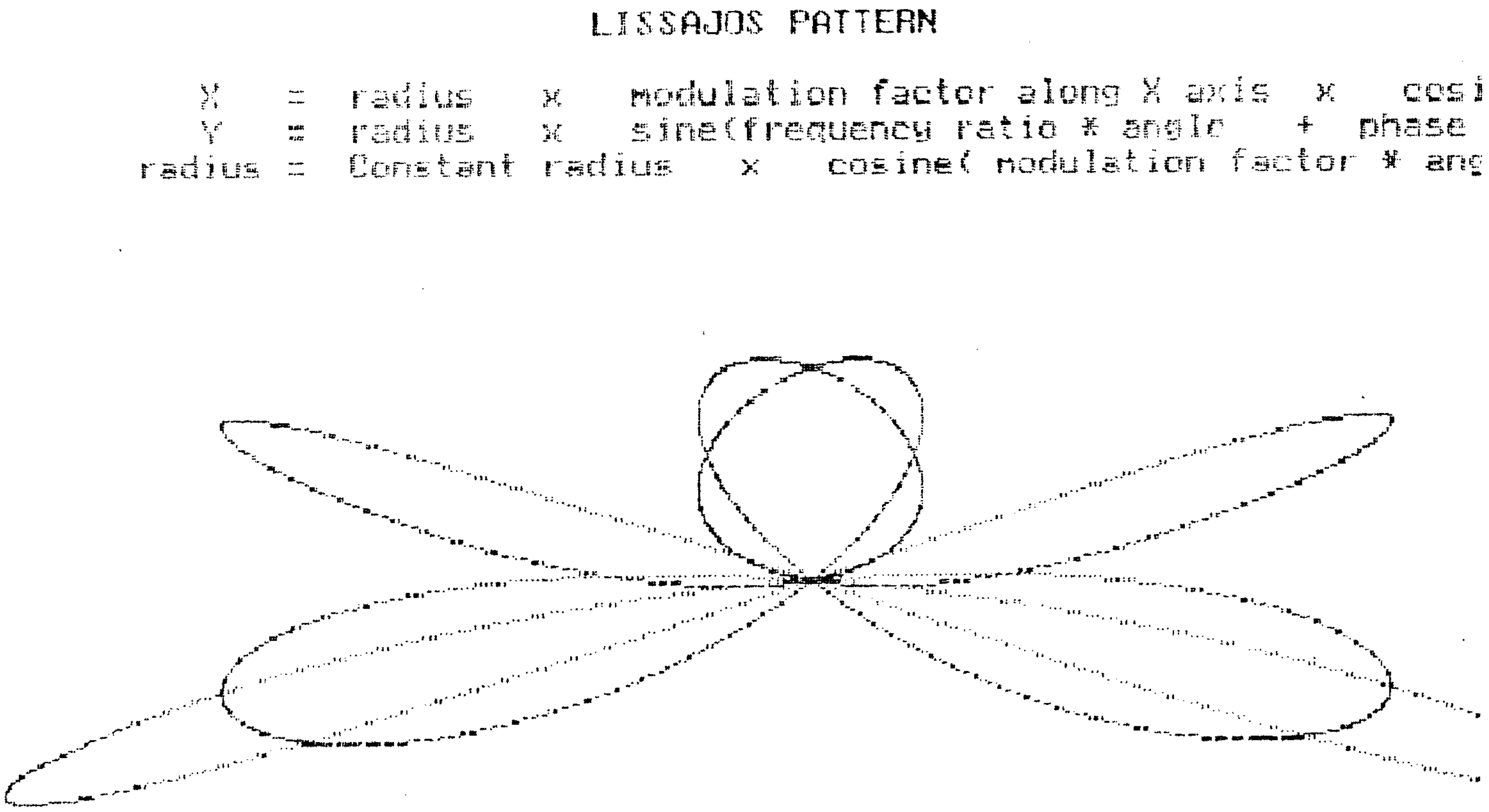


Modulation of spinals with factors 3 and 5

```
                      LISSAJOS PATTERN

     X    =  radius    x   modulation factor along X axis  x   cos
     Y    =  radius    x   sine(frequency ratio * angle   +  phase
radius    =  Constant radius   x    cosine( modulation factor * ang
```

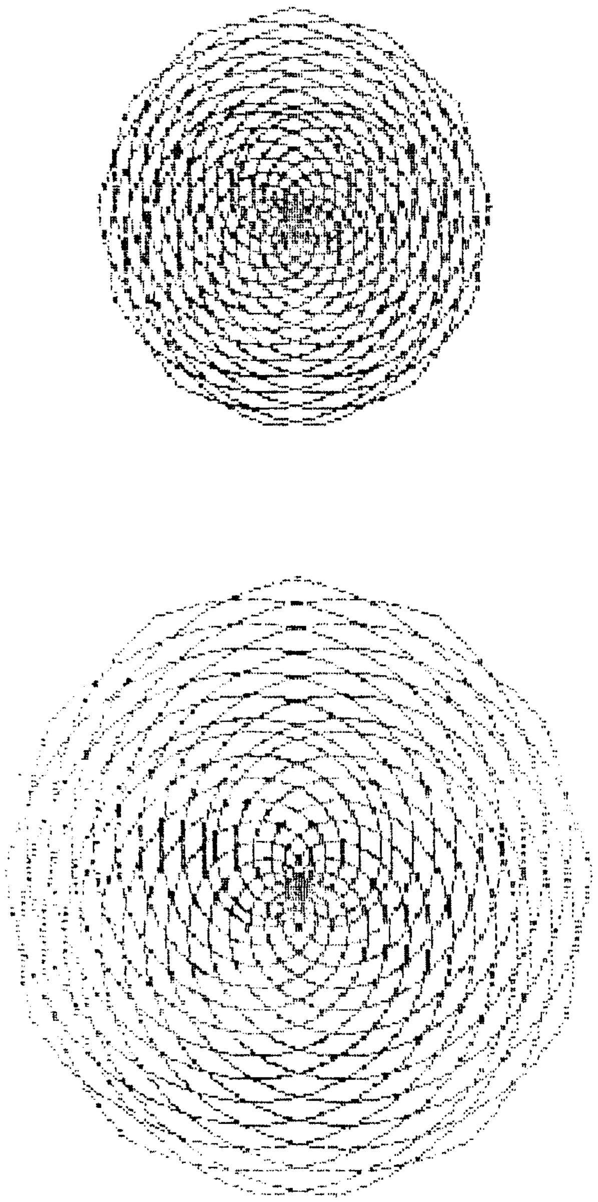


A bu𝐭terfly like pattern generated by Lisso.C(Program Drawing Lissajos pattern)



```
                    LISSAJOS PATTERN

  X    =  radius    x   modulation factor along X axis  x   cosi
  Y    =  radius    x   sine(frequency ratio * angle    +  phase
radius =  Constant radius    x    cosine( modulation factor * ang
```

Another interesting pattern generated by LISSO.C



Spirals and counterspirals (counter rotating spirals)
for different values of INC (0.2 and 0.4) respectively)

REFERENCES

1 Bhatt N.D.,Engineering Drawing (plane and solid Geometry) Charotar Publishing House 1989, pp.121-169.

2 El-Ghoul M,Folding of Fuzzy Graphs and Fuzzy Spheres, Fuzzy Sets and System 58(1993) North-Holland pp.355- 363.

3 McGregor J. and Watt A. The Art of Graphics for the IBM PC, ADDISON-WESLEY Publishing Company 1986 pp.399-422 and pp.207-249.

4 Rao S.B,Envelope Graphics on Your Spectrum, Electronics For You, Software Section June 1990,pp.95-97.

5 Tello E.R.,Turtle Geometry, Encyclopedia of personal computers.

6 Boden M.A., Creativity ad Computers, CURRENT SCIENCE, Vol.64,No.6,25,March 1993 pp.419-433.