ECEC-301 Project 2: Spirograph

Nathan Schomer

Part 1 – Introduction

This Python module allows users to create and draw virtual spirographs with custom parameters. Spirograph toys allow people to draw periodic roulette curves using a device similar to the one seen in Figure 1.

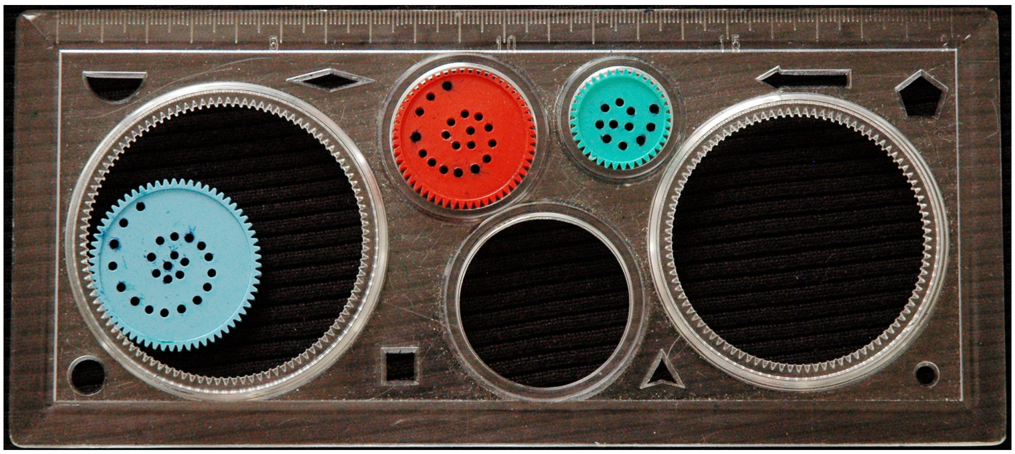

Figure 1. Example of a Spirograph Toy

They consist of a geared wheel nested within an outer circle. The user then places a drawing utensil within one of the holes. As the user rotates their pen in a circular motion, the inner circle will rotate as it moves around the circumference of the outer circle. The resulting shape is a function of several parameters and thus, can be described mathematically.

Part 2 – Mathematical Background

A spirograph can be represented geometrically which can be seen in Figure 2 (image source: Dr. James Shackleford).

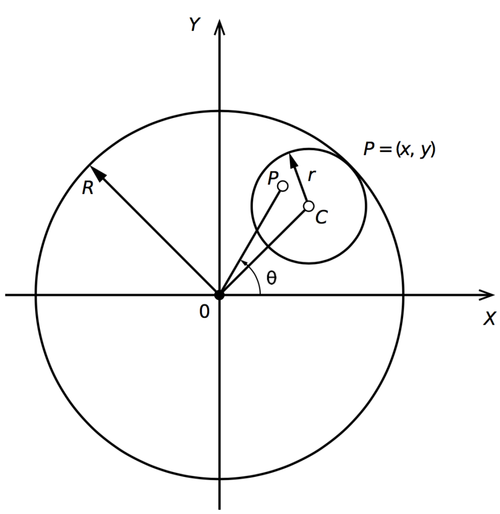

Figure 2: Geometry of the spirograph problem.

Variables

P Cartesian coordinate location of pen
R radius of large circle
r radius of small circle
C distance between center points of circles
θ polar angle location of pen

Additional Parameters

k = r / R ratio of radii
d = distance from P to C
l = d / r ratio of distance from P to C to radius of small circle

The additional parameters are necessary to the mathematical representation of a spirograph. Variable k is the ratio between the radii of the large circle and the small circle. Variable d is the distance from P, the pen's location to C, the center of the small circle. Lastly, variable l is the ratio between d and the radius of the small circle.

With this set of spirograph parameters, it's simple to calculate pen position over the domain θ = (0 : 360) in degrees. However, the period of the spirograph's shape may be greater that a single rotation of 360 degrees. Therefore, the period must be calculated. The period of the shape can be calculated by first finding the greatest common divisor of the radii of both circles. This is the greatest positive integer that evenly divides both r and R. The period is then equal to r divided by the greatest common divisor. The domain is now θ = (0 : 360*period).

The pen position may be calculated as:

$$P_x(\theta) = R\left((1-k)cos(\theta) + lkcos\left(\frac{1-k}{k}\theta\right)\right)$$

$$P_y(\theta) = R\left((1-k)sin(\theta) - lksin\left(\frac{1-k}{k}\theta\right)\right)$$

$P_x(\theta)$ is the X coordinate of the pen at θ and $P_y(\theta)$ is the Y coordinate of the pen at θ.

Part 3 – Source Code Overview

The spirograph module is implemented as a Python class with one external function. This module imports Python's math module for calculations and the turtle module for graphical display of the spirograph shapes.

Spirograph.__init__(self, R):

- Creates new spirograph object with an outer radius of R
- Returns nothing

Spirograph.setSmallCircle(self, r):

- Sets radius of the inner circle.
- Returns nothing

Spirograph.setPen(self, l, color):

- Sets color of pen and the pen's radius l.
- Returns nothing

Spirograph.draw(self):

- Draw spirograph using the current small circle and pen settings
- Operation
    - Calculate period
    - Calculate k
    - Calculate Cartesian position of pen for a full period of the shape
    - Set pen to calculated position

Spirograph.clear(self):

- Reset the drawing surface
- Returns nothing

getGCD(num1, num2):

- This function utilized Euclid's algorithm to calculate the greatest common divisor (GCD).
- Euclid's algorithm begins by dividing num1 by num2. Next, num1 is assigned the value of num2 and num2 is assigned the remainder of num1 divided by the previous num2 value (i.e. num1 mod num2). This continues in a loop until num1 mod num2 yields 0. Once the loop has terminated, the GCD is equal to the previous value of num1 mod num2.
- Note- this method is not a member of the Spirograph class

Part 4 – Using the Spirograph Module

Spirograph operations provided by the module include creation of new spirograph objects, setting the small circle radius, setting pen color and distance ratio, draw function, and clear function.

A typical instance of a Spirograph object would be operated as show in Figure 3. This instance was created in the Python interpreter but the same process applies for Python scripts. The resulting image is show in Figure 4.

```
Python 2.7.6 (default, Jun 22 2015, 17:58:13)
[GCC 4.8.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> from spirograph import *
>>> my_spirograph = Spirograph(500)
>>> my_spirograph.setSmallCircle(85)
>>> my_spirograph.setPen(0.65, 'red')
>>> my_spirograph.draw()
>>> my_spirograph.clear()
>>>
```

Figure 3: Example of spirograph module usage.

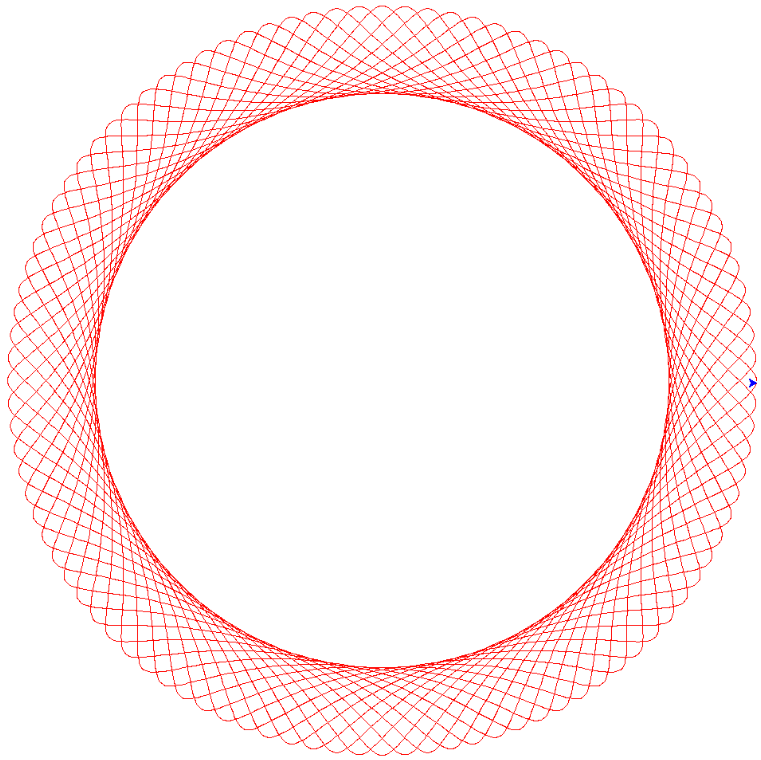

Figure 4: Example of drawing generated with spirograph module.