2023

【海報論文】

■一般組 □研究生組

□大學組

11th 台灣 TAIWAN 工業與應用數學會年會

TWSIAM Annual Meeting

Visualization of Pi with Python

Peng-Jen Lai (賴鵬仁), ***Department of Mathematics, National Kaohsiung Normal University, Taiwan***
(t1673Lai@gmail.com)

Abstract

In the movie "Contact" we saw some frames showed beautiful visualization of Pi. We try to simulate similar visualization of Pi with Python. We survey the formula of Pi at first. Then we draw the visualization figure of Pi by tracing Python turtle along the digital order of Pi. We found that on an ordinary notebook computer combined with Python we can simulate up to 1000 digits of Pi within 20 seconds.

It means that even for a collage student or a higher school student, one can render similar beautiful visualization of Pi as in the movie "Contact" by common used desktop or laptop combined with free shareware Python with satisfied computation efficiency.

Problem description

Mathematicians tried to approximate the value of Pi for thousands of years. Before the computer era, the series for arctangent (In 1671, James Gregory, and independently, Leibniz in 1673, discovered the Taylor series expansion for arctangent , is sometimes called Gregory's series (格雷戈里) or the Gregory–Leibniz series) was used by Madhava to estimate π to 11 digits around 1400. 1699 England mathematician Abraham Sharp use Sharp formula (夏普公式) to calculate Pi up to 70 digitals. In 1706, John Machin used the Gregory–Leibniz series to produce an algorithm that converged faster (馬欽公式). Machin calculated up to 100 digits of π with this formula [1, 2].

The approximation of Pi with the aid of computer began around the mid-20th when mathematicians John Wrench and Levi Smith reached 1,120 digits in 1949 [Wiki]. One import development is the invention of iterative method for calculating Pi. The iterative algorithms very different to infinite series method were independently published by Eugene Salamin and Richard Brent round 1975. This kind of algorithms were preferred after 1980 for faster efficiency than infinite series algorithms. But the rapid convergence causes a tradeoff: the iterative algorithms require significantly more memory than infinite series, this is a popular topic discussed in IT that use spaces to tradeoff time! [2]

Two algorithms were invented in 1995 thought as the most novel idea in computing π. They are called spigot algorithms. The BBP digit extraction algorithm, was discovered in 1995 by Simon Plouffe is the most famous one with producing any individual hexadecimal digit of π without calculating all the preceding digits with formula as Pi=sum_{k=0}^inf 1/ 16^k (4/(8k+1)-2/(8k+4)- 1/(8k+5) - 1/(8k+6)) [1, 2, 3].

Recently the another approach is using data analysis to unveil the mystery of Pi by exploring if there is some pattern hidden in the digits sequence of Pi [4]. The most attracting figures showing in the movie "Contact" were presented by the visual artist Martin Krzywinski and Cristian Ilies Vasile. Krzywinski (who makes every year Pi Day art, which can be graphics, words or music) [5]. We try to rebuild the work of artist Martin Krzywinski and Cristian Ilies Vasile. Krzywinski and found it is not so easy but also not very difficult to rebuild the similar exploring Pi figure with Python and usual notebook computer.

Results and discussion

The following figure 1 is the first trial version by drawing the locus of a Python turtle which tracing the digits sequence of Pi.

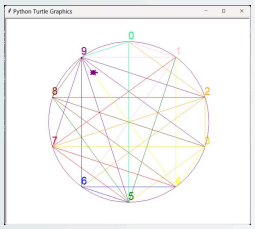

Figure. 1

In the following figure 2, we try the second version by letting the Python turtle every time go forward into the target region with randomly choosing location instead of a specific clock number location. This will spread out the tracing lines.

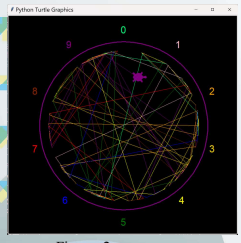

Figure. 2

In figure 3, we try the third version by coloring the arcs on the outside circle with corresponding clock number and simulate up to 1000 digits of pi.

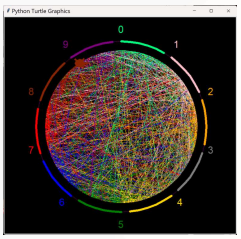

Figure. 3

In figure 4, we get the final version by adding a dynamic table counting the accumulation of every digits and simulate up to 1000 digits of pi with time span about 19 seconds.

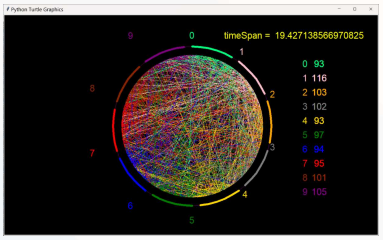

Figure. 4

The complete Python codes

```
# By Prof P-J Lai MATH NKNU 20230502
import turtle
import math
import random
import time
timestartr=time.time()
T = turtle.Turtle()
T.ht()
T.turtlesize(2)
T.shape('turtle')
turtle.tracer(0,0)
turtle.bgcolor("black")
colorList
= {'0':'SpringGreen','1':'pink','2':'orange','3':'grey','4':'gold','5':'green','6':'blue','7':'red','8':'Or
angeRed4','9':'purple',}
rOuter = 280
theta = 360/10
phase = 90
piDiskNumberCoor = []
rInner = rOuter - 30
def coord(r, i, theta, phase):
    return (r*math.cos(math.radians(-i*theta+phase)), r*math.sin(math.radians(-
i*theta+phase)))
for i in range(10):
    r=rInner
    piDiskNumberCoor.append(coord(r, i, 36, 90))
    T.goto( piDiskNumberCoor[i] )
    T.pendown()
    T.dot(5)
    T.penup()
r=rOuter+30
for i in range(10):
    if i ==6:
        r = r - 10
    if i == 5:
        r = r + 30
    if i == 4 or i == 6:
        r = r + 20
    T.penup()
    T.color(colorList[str(i)])
    T.goto( r*math.cos(math.radians(-i*theta+phase)), r*math.sin( math.radians(-
i*theta+phase)) )
    T.pendown()
    T.write(i, align="center",font=('Arial', 25))
T.penup()
r = rOuter
T.goto(coord(r, 5, 36, 90))
T.pendown()
T.pensize(1)
T.circle(r)
T.penup()
T.goto(coord(r, 5, 36, 90))
T.pensize(7)
T.pendown()
for i in range(10):
    T.color(colorList[str((5+i)%10)])
    T.circle(r, -30)
    T.penup()
    T.circle(r, -6)
    T.pendown()
T.penup()
pilntList80=[1, 4, 1, 5, 9, 2, 6, 5, 3, 5, 8, 9, 7, 9, 3, 2, 3, 8, 4, 6, 2, 6, 4,
        3, 3, 8, 3, 2, 7, 9, 5, 0, 2, 8, 8, 4, 1, 9, 7, 1, 6, 9, 3, 9, 9, 3,
        7, 5, 1, 0, 5, 8, 2, 0, 9, 7, 4, 9, 4, 4, 5, 9, 2, 3, 0, 7, 8, 1, 6,
        4, 0, 6, 2, 8, 6, 2, 0, 8, 9, 9]
pilntList=pilntList80
def randomKTheta(k):
    "return a random point coordinates in negiborhood of point k.
    for example, negiborhood of point 1 is between 1 and 2,,,"
    kTheta = math.radians(-k*theta+phase)
    kPlusOneTheta = math.radians(-(k+1)*theta+phase)
    DeltakTheta = -math.radians(theta)
    kThetaRandom = kTheta+random.random()*DeltakTheta
    return kThetaRandom
for i in range(10):
    T.color(colorList[str(i)])
    T.penup()
    T.goto( 400, -50*i+200)
    T.pendown()
    T.write(i, align="center",font=('Arial', 25))
T.penup()
r = rInner
T.pensize(1)
r = rInner
piCount = [0 for n in range(10)]
for i in range(0,len(pilntList)-1):
    randomKThetaTemp=randomKTheta(pilntList[i])
    piCount[pilntList[i]] = piCount[pilntList[i]] + 1
    T.goto( r*math.cos(randomKThetaTemp),
r*math.sin(randomKThetaTemp) ))
    T.pendown()
    T.color(colorList[str(pilntList[i])])
    T.shape("turtle")
    randomKThetaTemp=randomKTheta(pilntList[i+1])
    T.goto(( r*math.cos(randomKThetaTemp),
r*math.sin(randomKThetaTemp) ))
    T.penup()
for i in range(0,len(pilntList)-1):
    T.color(colorList[str(pilntList[i])])
    T.shape("turtle")
    T.goto( 450, -50*pilntList[i]+200)
    T.pendown()
    T.write(pilCount[pilntList[i]], align="center",font=('Arial', 25))
    T.penup()
timeSpan = time.time() - timestartr
print(timeSpan)
T.goto( 200, 300)
T.color("yellow")
T.pendown()
T.write("timeSpan  = ", align="center",font=('Arial', 25))
T.goto( 450, 300)
T.pendown()
T.write(timeSpan, align="center",font=('Arial', 25))
```

Conclusions

After many times try and error, we got the final version by adding a dynamic table counting the accumulation of every digits and simulate up to 1000 digits of pi with time span about 19 seconds. With a non-expensive desktop or laptop and free shareware Python we can teach collage students or high school students to repeat similar beautiful figures of Pi shown in movie "Contact". It takes only about 19 seconds to explore the pattern of Pi up to 1000 digits. This project could be transferred to many seed teachers with a little bit training based on the work of Martin Krzywinski and Cristian Ilies Vasile and our trial .

References

[1] The SIAM100-Digit Challenge.
[2] Pi, https://en.wikipedia.org/wiki/Pi
[3] Adamchik, Wagon, A simple formula for pi, A.M.Monthly 104 (1997), no. 9, 852-855.
[4] Pi is Beautiful – Numberphile, https://youtu.be/NPoj8lk9Fo4.
[5] 2022 pi day, http://mkweb.bcgsc.ca/pi/#projecthome