用 Python Turtle Graphics (龜作圖) 及動畫 來輔助學習 牛頓力學 及運動學

作者： 呂可名，台北市立 麗山高級中學
指導老師： 呂仁園，長庚大學 資訊系

摘要：

在這個專題研究中，我們打算運用 Python 以及 其龜作圖 (turtle) 模組，撰寫一套動畫程式，來輔助學習高中階段所學到的牛頓力學中與運動有關的部分，包含平拋運動、曲線運動、圓周運動、二體運動以及三體運動等等。

正文:

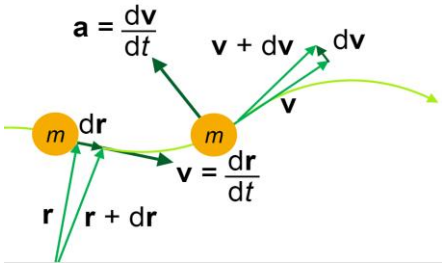
自從去年暑假，我開始接觸 Python，正式學習一種程式語言，我發現 Python 很簡單就可以學起來，特別是 Python 裡面有個 turtle 作圖功能，與我以前國中時代的 Scratch 程式比起來，更為精細，能控制的東西很多，好像比較方便表達更複雜的作圖行為。

過了不久，我也正式進入高中，成為高一新生，在高一的物理課程中，重新複習了國中時代就學過的牛頓力學，像是位置、速度、加速度、圓周運動、行星的軌道運動等等。有一天，我突然有個想法，是否我可以把 Python 和物理連在一起，把那些物理學上的運動現象，自己透過電腦程式把它們呈現出來。

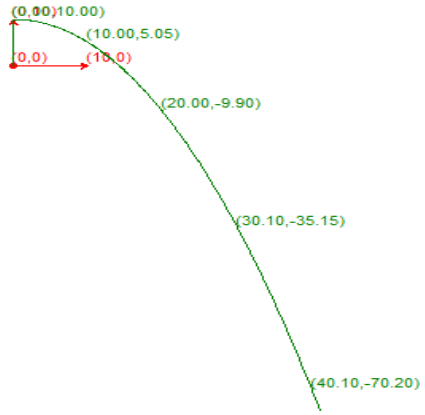
我就以高中物理教科書為起點，開始參考網路上有關這些物理觀念的說明，特別是 wikipedia 上的文章，整理出一些相關知識，指導老師協助我在 python 標準函式庫中找到了 一個純用 python 撰寫的程式 ”planet_and_moon.py”，就是在做太陽、地球、月球的運轉，我覺得很有價值，就把它特別閱讀清楚，一開始不懂的地方雖然很多，特別是有些特殊的龜作圖指令，我也得一一克服，還有幾個較難的 Python 語法，像是「類別」(class)的用法也還不大了解，藉由參加 Shally 老師帶領的程式俱樂部，我有機會與其他更厲害的人請教、討論，因此就越來越能理解，從中學到很多新的程式語法及用法。因此，我也把這個程式相關的重點和原理在此整理一下並呈現出來。

接著我就嘗試自己來寫程式了，我打算依序寫出以下這幾種運動的動畫程式，包含平拋運動、圓周運動、二(星)體運動以及三(星)體運動等等。我會先把我在課本上以及網路上的資訊整理下來，之後再列出我的程式碼，最後是執行它們，觀看成果。

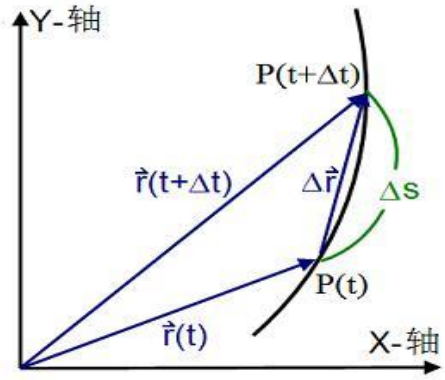
1. 位置、速度、加速度:



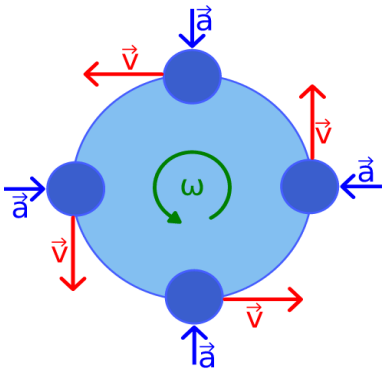
2. 平拋運動:



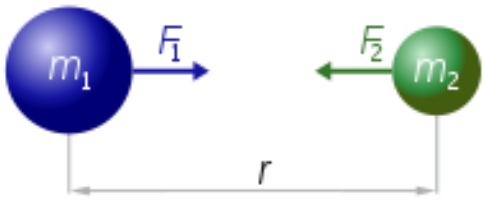
3. 一般的二維曲線運動:



4. 圓周運動 (含簡諧運動):



5. 牛頓萬有引力定律:

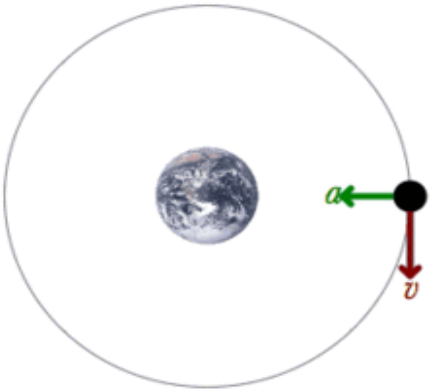


$$F_1 = F_2 = G\frac{m_1 \times m_2}{r^2}$$

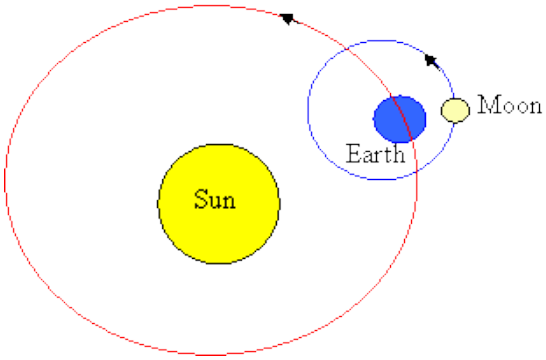
$$a_1 = G\frac{m_2}{r^2}$$

$$\mathbf{a}_1 = G\frac{m_2}{r_{21}^2}\hat{\mathbf{r}}_{21}$$

6. 二體運動:

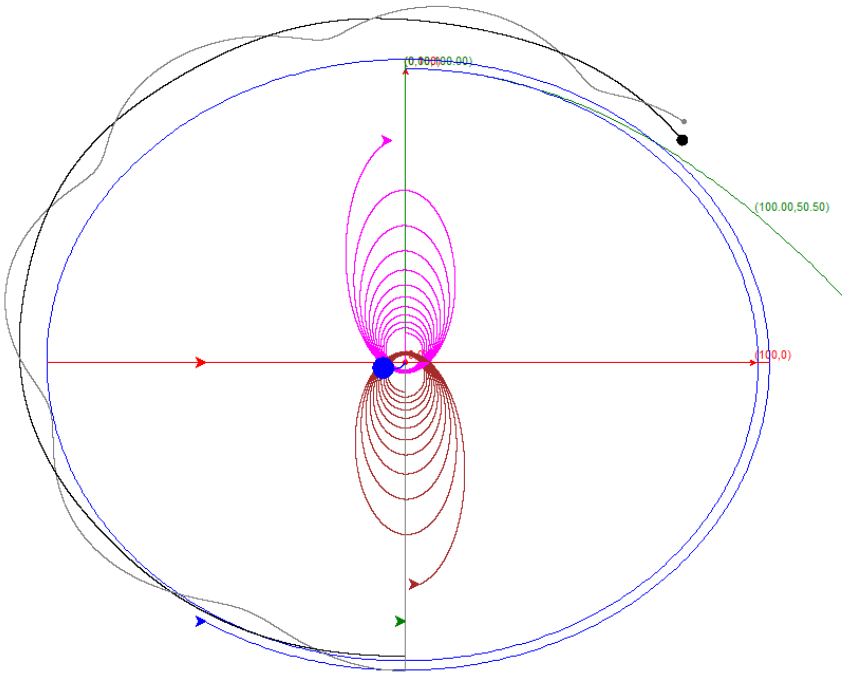


7. 三體運動:



我在老師的指導下逐一完成為以上各種運動撰寫 Python 程式碼，並實際運行無誤，從中也可以更改一些參數，來試玩一下，有些參數要是要經過一段時間的實驗，才會得到比較好的運轉效果，每次改過參數還需要重跑程式，雖有些不便，但隨後能直接看到改變後的結果，覺得很有趣。

目前我還不大會寫圖形使用者介面(GUI)，未來將朝向這方面來學習一下。以上這些程式目前先以附錄方式列於本文後面以供參考，執行的結果的螢幕截圖如下。希望有機會在 Pycon 會議上發表這些學習成果。

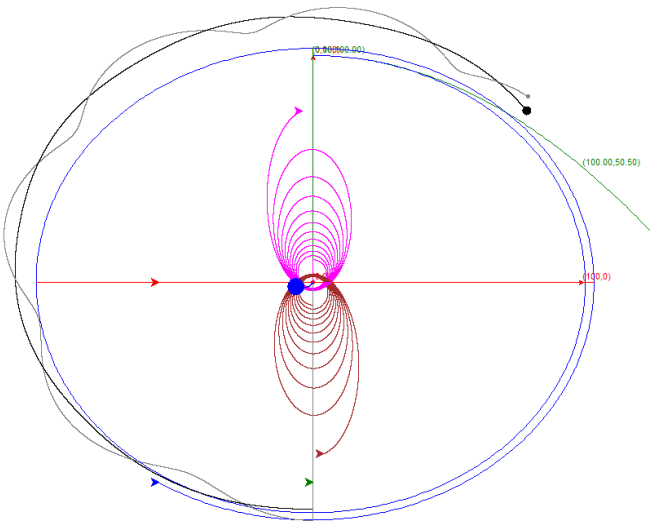


參考資料

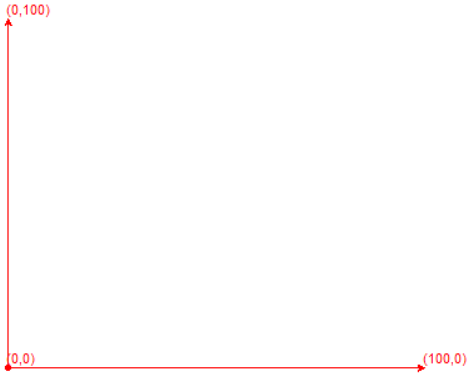
1. https://docs.python.org/3.4/library/turtle.html
2. http://zh.wikipedia.org/wiki/牛頓萬有引力定律
3. http://zh.wikipedia.org/wiki/牛頓運動定律
4. http://zh.wikipedia.org/wiki/克卜勒定律
5. http://zh.wikipedia.org/wiki/平拋運動
6. http://zh.wikipedia.org/wiki/曲線運動
7. http://zh.wikipedia.org/wiki/圓周運動
8. http://zh.wikipedia.org/wiki/簡諧運動
9. http://zh.wikipedia.org/wiki/二體問題
10. http://zh.wikipedia.org/wiki/三體問題
11. http://zh.wikipedia.org/wiki/混沌理論

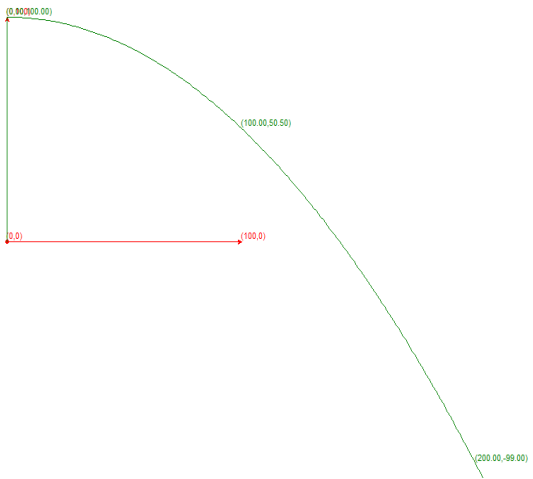
附錄: 程式列表及說明

```
'''
cvryGrav001.py

用 Python 龜作圖 及動畫
來輔助學習 牛頓力學 及運動學

作者：                          呂可名，台北市立 麗山高級中學
共同作者及指導老師：    呂仁園，長庚大學 資訊系

摘要：

在這個專題研究中，
我們打算運用 Python 以及 其龜作圖 (turtle)模組，
撰寫一套動畫程式，
來輔助學習高中階段所學到的牛頓力學中與運動有關的部分，

包含：

自由落體(平拋運動)、
圓周運動(含簡諧運動)、
雙(星)體運動 以及
三(星)體運動 等等。

renyuan, 2015/02/28
calvin, 2015/03/22

'''

#
# 本程式運用 turtle 模組中的
# Turtle 類別 來協助畫圖，
# Vec2D  類別 來協助 2 維向量 的計算
#
from turtle import  Turtle, Vec2D

def main():

    畫坐標軸()
    自由落體()
    圓周運動()
    雙星運動()
    三星運動()

def 畫坐標軸():

    u= Turtle()

    # 設座標系統
```

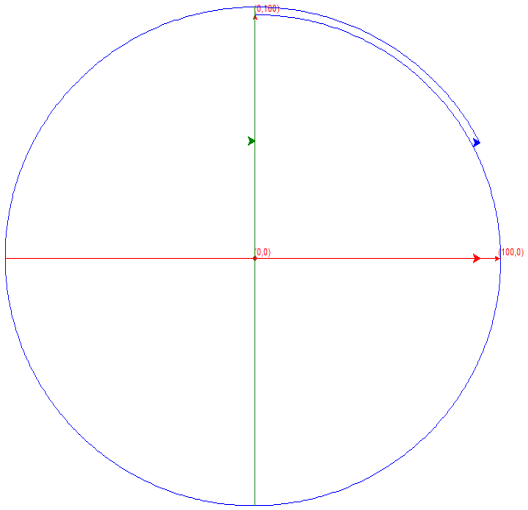


```
    boundary= B= 100 # 本世界座標邊界值
    u.getscreen().setworldcoordinates(-B,-B,+B,+B)

    # 調整龜筆大小及顏色準備畫圖
    u.shapesize(.5)
    u.color('red')

    # 畫橫軸
    u.home()
    u.setheading(0)
    u.goto(100,0)
    u.stamp()
    u.write('(%.0f,%.0f)'%u.pos())

    # 畫縱軸
    u.home()
    u.setheading(90)
    u.goto(0,100)
    u.stamp()
    u.write('(%.0f,%.0f)'%u.pos())


    # 畫原點
    u.home()
    u.shapesize(.2)
    u.shape('circle')
    u.stamp()
    u.write('(%.0f,%.0f)'%u.pos())

    # 功成身退
    u.hideturtle()

def 自由落體():

    z= Turtle()
    z.color('green')

    z.s=  Vec2D(0,1) *100    # 初位置，隨便給。
    z.a= -z.s                # 初加速度，讓它向下。

    v0=  100                 # 隨便給個初速度
    z.v=  Vec2D(1,0) *v0     # 初速度，水平方向。

    t=   0  # 初時間
    dt= .01 # 微小的時間遞增量
    T=  10  # 末時間

    while t < T:

        # 龜作圖在此
        z.goto(z.s)
```

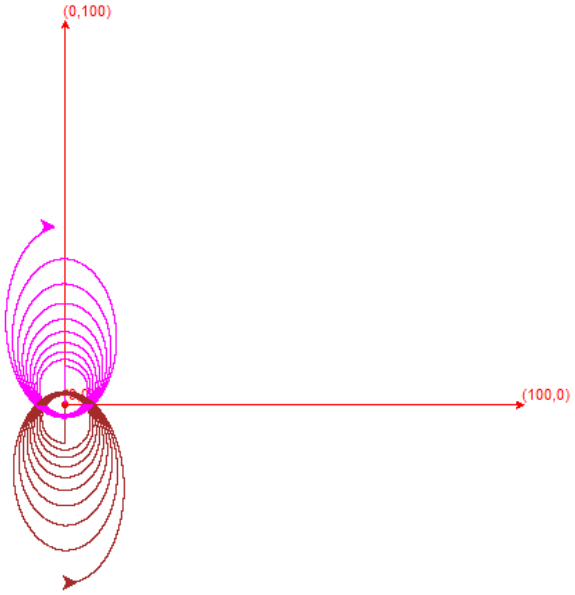




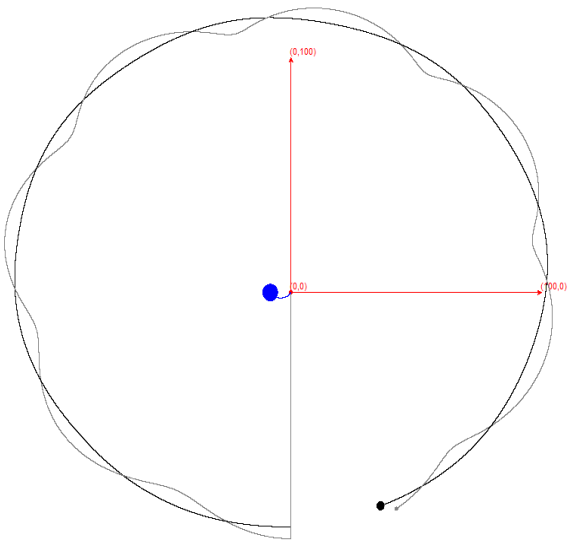
```
        # 每隔一段時間，龜寫一下座標值於幕上。
        if int(t*100)%100==0:
            z.write(z.pos())

        # 牛頓力學(運動學)精華在此：
        z.dv= z.a * dt
        z.ds= z.v * dt

        z.v += z.dv
        z.s += z.ds

        # 時間遞增
        t += dt

def 圓周運動():
    '''
    含簡諧運動
    '''

    z= Turtle()
    z.color('blue')

    z.s=  Vec2D(0,1) *100
    z.a= -z.s

    #
    # 設定圓周運動的必要初速度
    # 好像是 a= v*v/R，就可以是圓周運動。
    # 需要再找一下 理論基礎
    #
    v0= (abs(z.s)*abs(z.a))**.5
    z.v=  Vec2D(1,0) *v0

    '''
    這個 v0 可以 變化看看，
    e.g., v0= 0, 1, 2, 5, 10, 20, 50, 100, 1000, ...
    Vec2D(1,0) 也可以試試看改成 (1,1), (0,1)看看。
    '''

    zx= z.clone() # 用來幫助畫 簡諧運動  x 方向
    zy= z.clone() # 用來幫助畫 簡諧運動  y 方向
    zx.color('red')
    zy.color('green')

    t=   0
    dt= .01
    T=  10

    while t < T:
```



```
        # 龜作圖在此
        z.goto(z.s)             # 圓周運動

        # 順便畫簡諧運動
        zx.goto(z.s[0], 0)      # x 方向的簡諧運動
        zy.goto(    0, z.s[1]) # y 方向的簡諧運動

        # 圓周運動關鍵在此，加速度隨時保持向心。
        z.a= -z.s

        # 牛頓力學(運動學)精華在此：
        z.dv= z.a * dt
        z.ds= z.v * dt

        z.v += z.dv
        z.s += z.ds

        # 時間遞增
        t += dt

def 雙星運動():
    '''
    雙星運動涉及 2 個星體，
    個別質量的差異，
    初速度的設定，
    以及引力常數 G 的倍率，
    該如何設定，
    才能讓雙星之間的運動互繞，
    不會彼此脫離，
    花樣滿多的，
    饒富趣味！
    '''

    G= 1 # 引力常數，本來是 6.67e-11，在此簡化，先設定為 1

    z1= Turtle()
    z1.color('magenta')
    z1.m= 100
    z1.s=  Vec2D(0,1) *10
    z1.v=  Vec2D(1,0) *1

    z2= Turtle()
    z2.color('brown')
    z2.m= 100
    z2.s=  Vec2D(0,-1) *10
    z2.v=  Vec2D(-1,0) *1

    # 雙星運動關鍵在此，加速度根據牛頓重力公式來計算。
    z1.a= G *z2.m *(z2.s-z1.s)*(1/abs(z2.s-z1.s)**3)
    z2.a= G *z1.m *(z1.s-z2.s)*(1/abs(z1.s-z2.s)**3)

```



```
    t=   0
    dt= .01
    T=  1000

    z1.getscreen().tracer(100,0) # 讓螢幕更新快一點。

    while t < T:

        # 龜作圖在此
        z1.goto(z1.s)
        z2.goto(z2.s)

        # 雙星運動關鍵在此，加速度根據牛頓重力公式來計算。
        z1.a= G *z2.m *(z2.s-z1.s)*(1/abs(z2.s-z1.s)**3)
        z2.a= G *z1.m *(z1.s-z2.s)*(1/abs(z1.s-z2.s)**3)

        # 牛頓力學(運動學)精華在此：
        z1.dv= z1.a * dt
        z1.ds= z1.v * dt

        z1.v += z1.dv
        z1.s += z1.ds

        z2.dv= z2.a * dt
        z2.ds= z2.v * dt

        z2.v += z2.dv
        z2.s += z2.ds

        # 時間遞增
        t += dt

def 三星運動():
    '''
    三星運動涉及 3 個星體，
    個別質量的差異，通常讓 m1>>m2>>m3
    彼此間的距離 r12, r23, 也讓他們 r12>>r23

    初速度的設定，
    以及引力常數 G 的倍率，
    該如何設定，
    才能讓三星之間的運動互繞，
    不會彼此脫離，
    花樣更多，更有趣！
    '''


    G= 1

    z1= Turtle()
    z1.color('blue')
    z1.m= 100
```



```
    z1.s=  Vec2D(0,1) *0
    z1.v=  Vec2D(1,0) *0

    z2= Turtle()
    z2.color('black')
    z2.m= 1
    z2.s=  Vec2D(0,-1) *100
    z2.v=  Vec2D(-1,0) *1

    z3= Turtle()
    z3.color('gray')
    z3.m= .1
    z3.s=  Vec2D(0,-1) *105
    z3.v=  Vec2D(-1,0) *1.5

    z1.shape('circle');z1.shapesize(1.0)
    z2.shape('circle');z2.shapesize( .5)
    z3.shape('circle');z3.shapesize( .2)

    t=   0
    dt= .01
    T=  1000

    z1.getscreen().tracer(100,0) # 讓螢幕更新快一點。

    while t < T:

        # 龜作圖在此
        z1.goto(z1.s)
        z2.goto(z2.s)
        z3.goto(z3.s)

        #
        # 三星運動關鍵在此，
        # 加速度根據牛頓重力公式來計算。
        # 注意每個星體都受到其他 2 星體的引力作用
        #
        z1.a= G *( z2.m *(z2.s-z1.s)*(1/abs(z2.s-z1.s)**3)
                +z3.m *(z3.s-z1.s)*(1/abs(z3.s-z1.s)**3))

        z2.a= G *( z1.m *(z1.s-z2.s)*(1/abs(z1.s-z2.s)**3)
                +z3.m *(z3.s-z2.s)*(1/abs(z3.s-z2.s)**3))

        z3.a= G *( z1.m *(z1.s-z3.s)*(1/abs(z1.s-z3.s)**3)
                +z2.m *(z2.s-z3.s)*(1/abs(z2.s-z3.s)**3))

        # 牛頓力學(運動學)精華在此：
        z1.dv= z1.a * dt
        z1.ds= z1.v * dt

        z1.v += z1.dv
        z1.s += z1.ds
```

```
305
306         z2.dv= z2.a * dt
307         z2.ds= z2.v * dt
308
309         z2.v += z2.dv
310         z2.s += z2.ds
311
312         z3.dv= z3.a * dt
313         z3.ds= z3.v * dt
314
315         z3.v += z3.dv
316         z3.s += z3.ds
317
318         # 時間遞增
319         t += dt
320
321
322 if   name   =='  main   ':
323     main()
324
```