Pacman Ghost Algorithm

吳智鴻 教授
國立臺中教育大學 數位內容科技學系
EMAIL:CHWU@MAIL.NTCU.EDU.TW
WEBSITE: 120.108.221.55/PROFCHWU/DCTAI

2020/10/07

Pacman ghost algorihtm

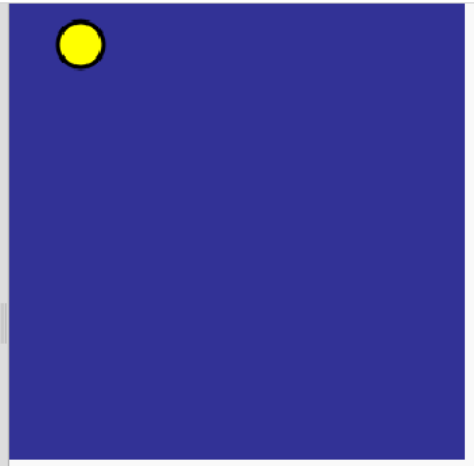
https://www.101computing.net/pacman-ghost-algorithm

Python Code

trinket | Stop | Modules | Share | Remix

main.py

```
#Pacman Ghost Algorithm - www.101computing.net/pacman-ghost-algorithm/
from math import atan, cos, sin
from processing import *

WIDTH=300
HEIGHT=300
pacman_X = 30
pacman_Y = 30
delay = 5

ghost_X = 10
ghost_Y = 10

def setup():
    strokeWeight(3)
    frameRate(20)
    size(300,300)

def moveGhost():
  global ghost_X,ghost_Y,pacman_X,pacman_Y
  fill(255,0,0)
  stroke(255,255,255)

  #Find out the direction (angle) the Ghost needs to move towards
  #Using SOH-CAH-TOA trignometic rations
  opposite=pacman_Y-ghost_Y
  adjacent=pacman_X-ghost_X
```



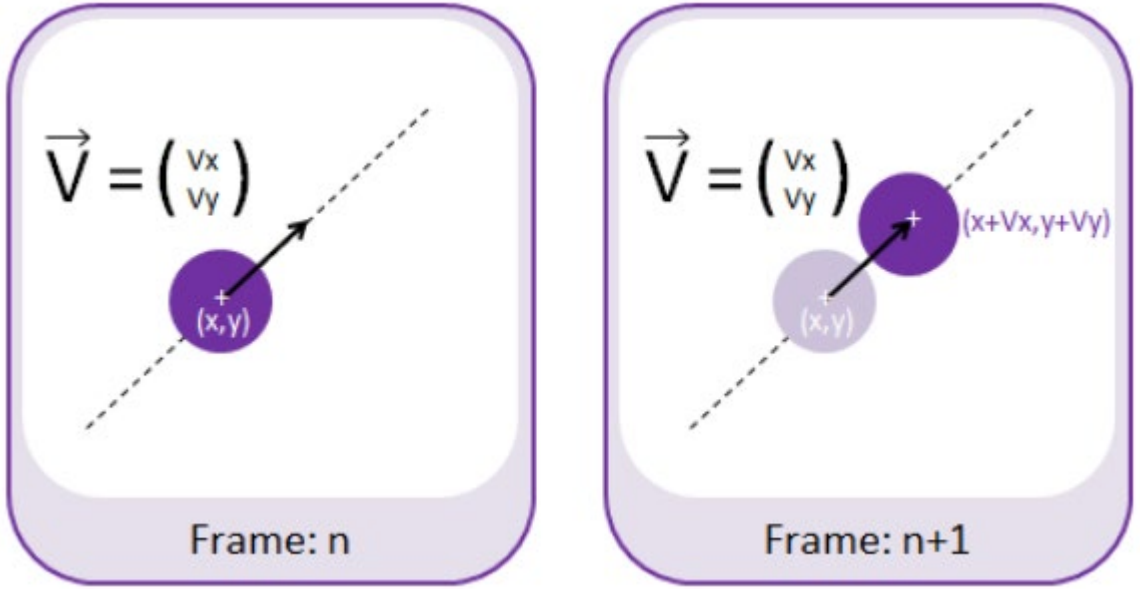
Pacman Ghost Algorithm

In a game of Pacman a specific algorithm is used to control the movement of the ghosts who are chasing (running towards) Pacman.

For this challenge we will assume that ghosts can walk through walls (as ghosts do!). So we will implement an algorithm that is slightly different to the algorithm used in the real game of Pacman where ghosts can only run alongside the corridors of the maze.

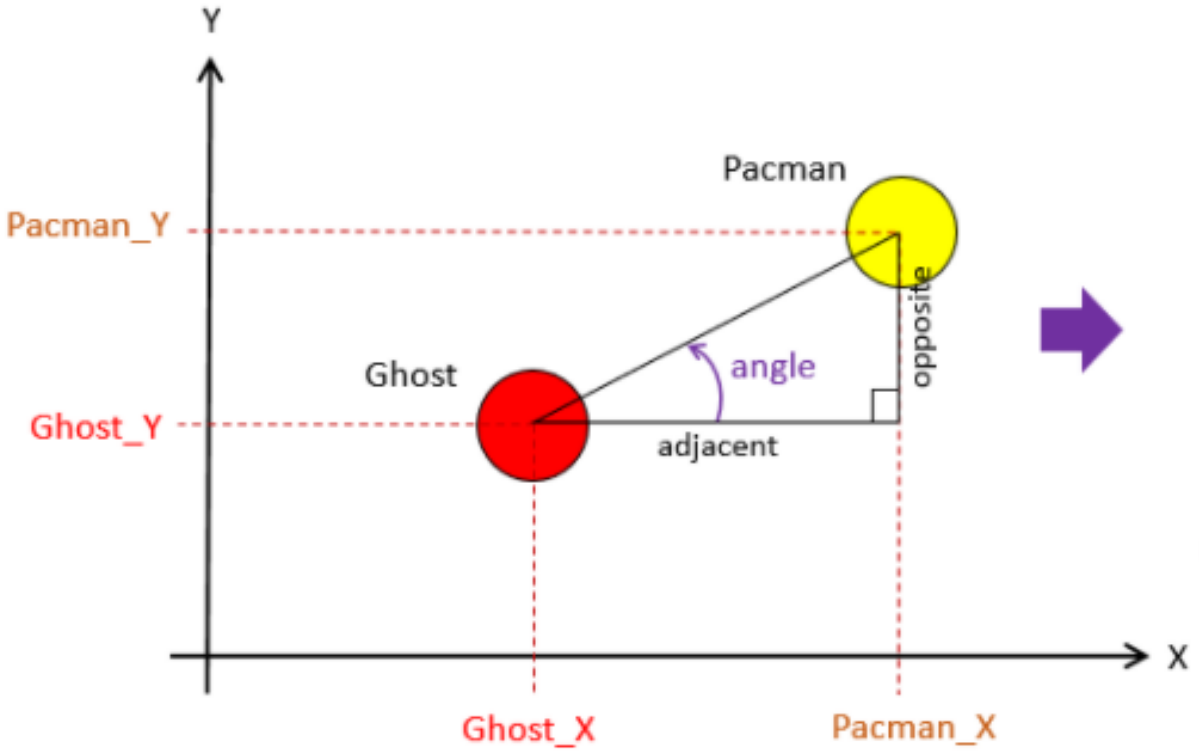
Idea of algorithm

Our algorithm will be used in a frame based game where the sprites (e.g. Pacman, Ghosts) are positioned using (x,y) coordinates. The Pacman movement will be based on the position of the mouse cursor whereas the Ghosts will use a velocity vector (vx,vy) to move/translate between two frames.



Trignometric Ratios: SOH-CAH-TOA

Our algorithm will use the trigonometric ration to find the angle the Ghost needs to head towards to chase Pacman.

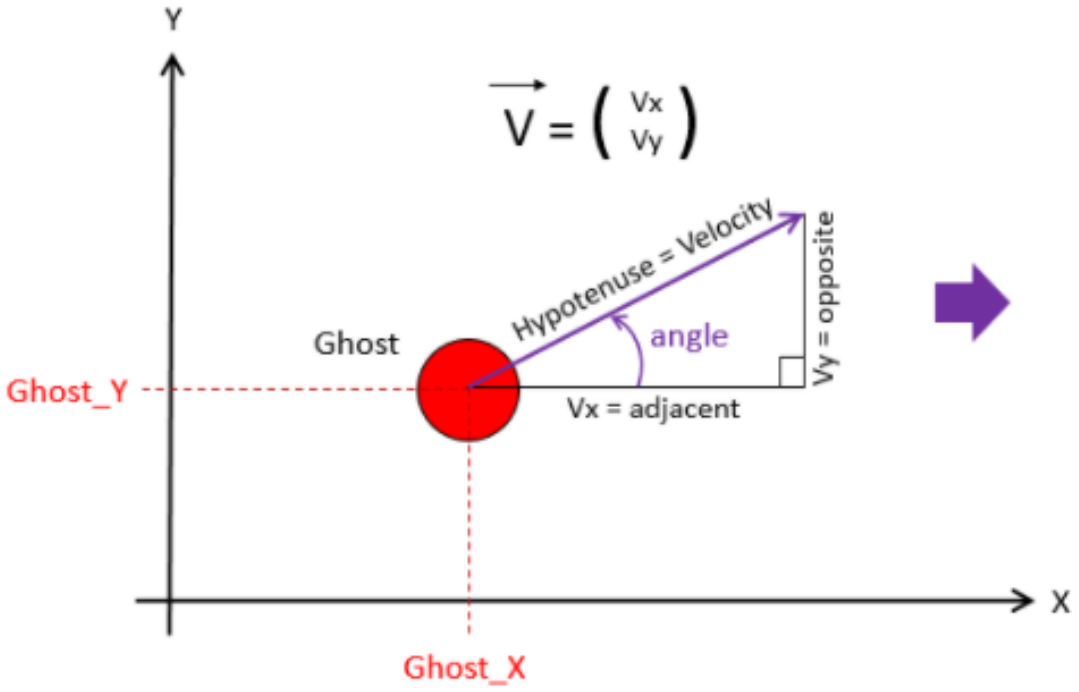


$$\tan(\text{angle}) = \frac{\text{opposite}}{\text{adjacent}}$$

$$\text{angle} = \tan^{-1}\left(\frac{\text{opposite}}{\text{adjacent}}\right)$$

$$\text{angle} = \tan^{-1}\left(\frac{\text{Pacman\_Y} - \text{Ghost\_Y}}{\text{Pacman\_X} - \text{Ghost\_X}}\right)$$

The next step of our algorithm will use this angle to calculate the velocity vector (Vx,Vy) of the ghost:



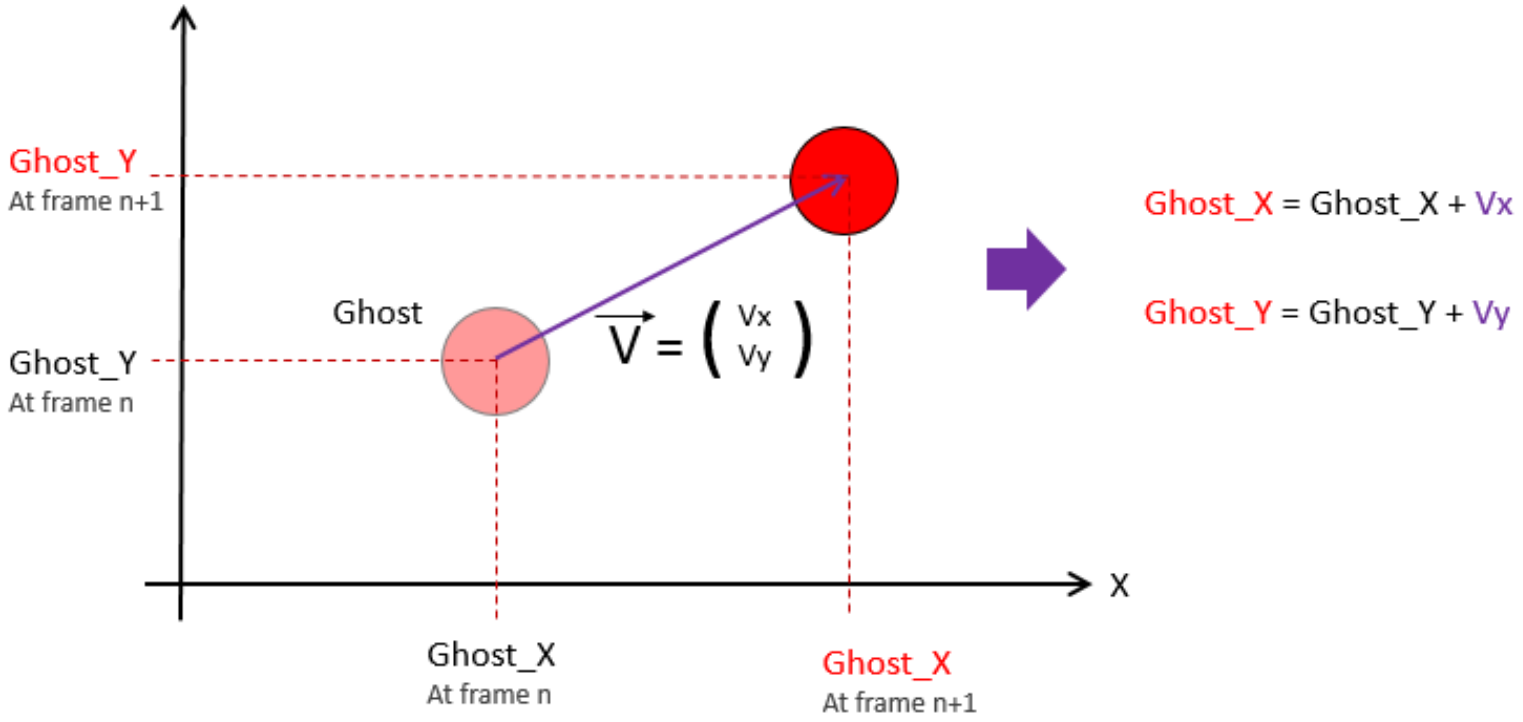
$$\cos(\text{angle}) = \frac{\text{adjacent}}{\text{hypotenuse}} = \frac{Vx}{\text{Velocity}}$$

$Vx = \text{Velocity} \times \cos(\text{angle})$

$$\sin(\text{angle}) = \frac{\text{opposite}}{\text{hypotenuse}} = \frac{Vy}{\text{Velocity}}$$

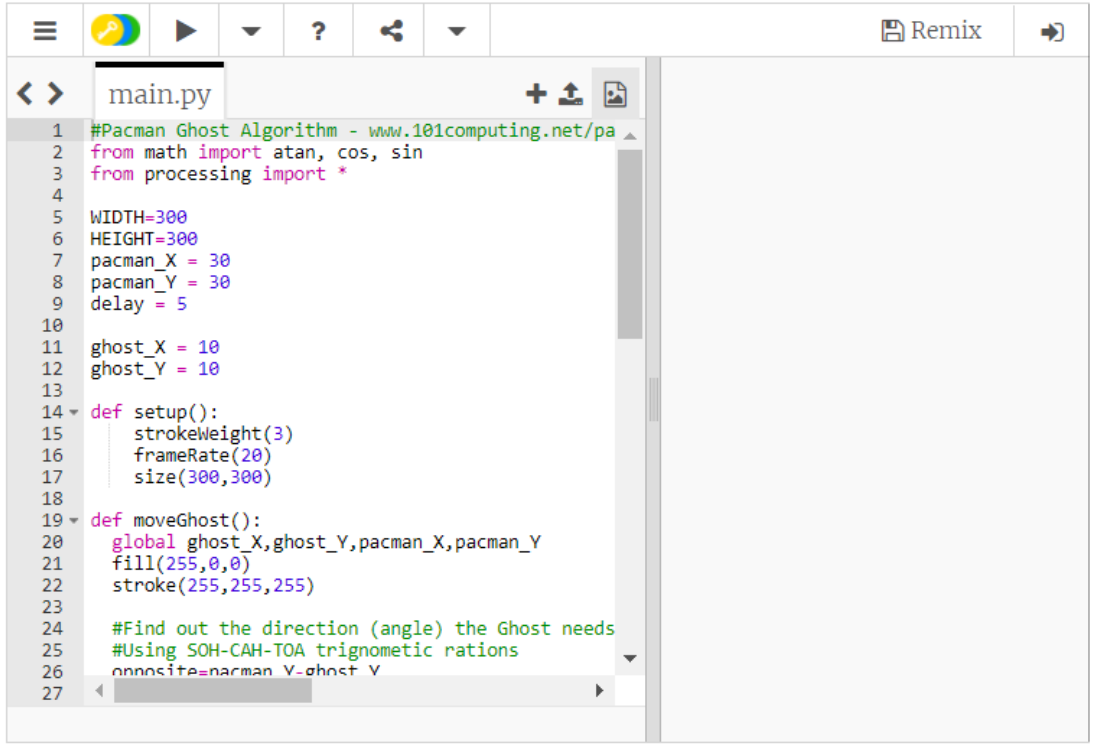
$Vy = \text{Velocity} \times \sin(\text{angle})$

The final step of our algorithm will update the (x,y) coordinates of our ghost to apply the velocity vector translation before drawing the ghost sprite on the screen.



Try it!

Python Code



```
1  #Pacman Ghost Algorithm - www.101computing.net/pacman-ghost-algori
2  from math import atan, cos, sin
3  from processing import *
4
5  WIDTH=300
6  HEIGHT=300
7  pacman_X = 30
8  pacman_Y = 30
9  delay = 5
10
11 ghost_X = 10
12 ghost_Y = 10
13
14 def setup():
15     strokeWeight(3)
16     frameRate(20)
17     size(300,300)
```

基本環境設定

遊戲背景設定

Ghost移動副程式

```
19 def moveGhost():
20   global ghost_X,ghost_Y,pacman_X,pacman_Y
21   fill(255,0,0)
22   stroke(255,255,255)
23
24   #Find out the direction (angle) the Ghost needs to move towards
25   #Using SOH-CAH-TOA trignometic rations
26   opposite=pacman_Y-ghost_Y
27   adjacent=pacman_X-ghost_X
28   angle = atan(opposite/adjacent)
29   if ghost_X>pacman_X:
30     angle=angle+180
31
32   #Use this angle to calculate the velocity vector of the Ghost
33   #Once again using SOH-CAH-TOA trignometic rations
34   velocity=3 #pixels per frame
35
36   vx = velocity * cos(angle)
37   vy = velocity * sin(angle)
38
39   #Apply velocity vector to the Ghost coordinates to move/translat
40   ghost_X = ghost_X + vx
41   ghost_Y = ghost_Y + vy
42
43   #Draw Ghost
44   ellipse( ghost_X,ghost_Y,60,60)
```

Ghost造型

Ghost移動角度

Ghost移動Velocity計算

Ghost移動方向

繪製Ghost

```
46  def movePacman():
47      global pacman_X, pacman_Y
48
49      fill(255,255,0)
50      stroke(0,0,0)
51      fc = environment.frameCount
52
53      #Pacman follows the mouse cursor
54      pacman_X += (mouse.x-pacman_X)/delay;
55      pacman_Y += (mouse.y-pacman_Y)/delay;
56
57      #Draw Pacman
58      ellipse(pacman_X,pacman_Y,30,30)
59
60  def playGame():
61    background(50,50,150)
62    movePacman()
63    moveGhost()
64
65  draw = playGame
66  run()
67
```

Pcaman
移動控制

Playgame
副程式

主程式

END