```
#!/usr/bin/python
# -*- coding: utf-8 -*-

"""
ZetCode Tkinter tutorial

This script draws lines on
the Canvas.

Author: Jan Bodnar
Last modified: November 2015
Website: www.zetcode.com
"""

from Tkinter import Tk, Canvas, Frame, BOTH
import robotState, Postman, time
from math import pi, cos ,sin
import Trig

class Example(Frame):


    center = 300
    scalar = -20
    sleep = 250

    prevx, prevy = center, center
    prevgx, prevgy = prevx, prevy
    canvas = None
    pathHandler = None

    def __init__(self, parent, robotState, pathHandler):
        Frame.__init__(self, parent)
        self.pathHandler = pathHandler
        self.state = robotState
        self.parent = parent
        self.initUI()

    def initUI(self):
        self.canvas = Canvas(self)
        self.parent.title("Draw log")
        self.pack(fill=BOTH, expand=1)

        self.canvas.pack(fill=BOTH, expand=1)

        self.after(self.sleep, self.draw)

        for x in range(0,500,30):
            self.canvas.create_line(x,0,x,500, fill="blue")

        for y in range(0,500,30):
            self.canvas.create_line(0,y,500,y, fill="blue")

        px, py = 300,300
        len = self.pathHandler.length()

        for i in range(0,len):
            x,y = self.pathHandler.position(i)
            x = x*self.scalar+self.center
            y = y*self.scalar+self.center
            self.canvas.create_line(py,px,y,x, fill="red")
            px, py = x, y

            """
            used to mark points on the path of interest
            if i==986 or i==1048:
                self.canvas.create_oval(y-2,x-2,y+2,x+2, fill="black")
```

```
            """

    def draw(self):
        x,y = self.state.getPosition()
        gx, gy = self.pathHandler.position(self.pathHandler.getCurrentIndex())

        self.gx = gx*self.scalar + self.center
        self.gy = gy*self.scalar + self.center

        self.x  = x*self.scalar  + self.center
        self.y  = y*self.scalar  + self.center

        self.canvas.create_line(self.prevy, self.prevx, self.y, self.x, width = 2)

        self.canvas.create_line(self.gy-1, self.gx-1, self.gy+1, self.gx+1, fill="purple",
width = 2)
        self.canvas.create_line(self.gy+1, self.gx-1, self.gy-1, self.gx+1, fill="purple",
width = 2)

        #Used to draw a laser between the current point and the point it is aiming for
        #self.canvas.create_line(self.y,self.x, self.gy, self.gx, fill="pink", width = 3)

        self.prevx = self.x
        self.prevy = self.y
        self.prevgx = self.gx
        self.prevgy = self.gy

        robotDirection = self.state.getDirection()
        laserScan = self.state.getLaserScan()

        for i in range(0,360):
            rad = Trig.degToRad(i)
            index = int(Trig.radToLaser(rad,robotDirection))
            if (index>0 and index<270):
                length = laserScan['Echoes'][index]
                gx= x + cos(rad)*length
                gy= y + sin(rad)*length

                gx= gx*self.scalar + self.center
                gy= gy*self.scalar + self.center

            self.canvas.create_line(gy,gx,gy+2,gx+2)

        # arrange for the next frame to draw in 1/2 seconds
        self.after(self.sleep, self.draw)




def main(robotState,pathHandler):
    root = Tk()
    ex = Example(root,robotState,pathHandler)
    root.geometry("500x500+300+300")
    root.mainloop()
    return ex
```