9

การได้มาซึ่งข้อมูลด้วย Python และ Tkinter

9.1 พื้นฐาน

วิธีการเปิด ():ฟังก์ชันนี้เป็นวิธีการที่ใช้บ่อยที่สุด
ที่มีอยู่ใน Python และใช้เพื่อจัดการไฟล์

ในการเปิดคำสั่ง le:

> > >f = เปิด ('test.txt', 'w')

คำสั่งนี้สร้างไฟล์ test.txt ในโฟลเดอร์เดียวกับที่ไฟล์
ล่าม Python ถูกบันทึกหรือตำแหน่งจากตำแหน่งที่รันโค้ด

โหมดที่สามารถใช้กับฟังก์ชัน open() ได้มีอธิบายไว้ใน
ตาราง 9.1.

วิธีเขียน ():เมื่อไฟล์เปิดอยู่ในโหมดเขียน
วิธีเขียน () ใช้เพื่อเริ่มเขียนไปยังวัตถุไฟล์ วิธีเขียน () รับอาร์กิวเมนต์อินพุตในรูป
แบบของสตริงเท่านั้น

> > >F.write("สวัสดีชาวโลก!\n")

ในที่นี้ “Hello World” เป็นสตริง และ \n หมายถึงอักขระขึ้นบรรทัดใหม่ ในการ
เขียนลำดับของสตริง ใช้เมธอด writelines():

> > >sq = ["การเขียนโปรแกรม Python สำหรับ Arduino\n", "ลาก่อน\n"]

> > >W.writelines(ตร.)

วิธีปิด ():ใช้เมธอด close() เพื่อปิดไฟล์ และ
วัตถุไฟล์ไม่สามารถใช้ได้อีก คำสั่งคือ:

> > >W.close()

วิธีการอ่าน ():วิธี read() นี้จะอ่านข้อมูลของไฟล์ ใช้
วิธีนี้ เปิดไฟล์ด้วยโหมดที่เข้ากันได้กับการอ่าน เช่น w+, r, r+ หรือ a+:

> > >D = เปิด ('test.txt', 'r')

> > >กลัว()

ตารางที่ 9.1

คำอธิบายของ Modes

โหมด	คำอธิบาย
W	โหมดนี้เปิดไฟล์เพื่อเขียนเท่านั้น มันเขียนทับไฟล์ที่มีอยู่
w+	โหมดนี้เปิดไฟล์เพื่อเขียนและอ่านทั้งสองอย่าง มันเขียนทับไฟล์ที่มีอยู่ โหมดนี้เปิดไฟล์เพื่อ
R	อ่านเท่านั้น
r+	โหมดนี้เปิดไฟล์เพื่อเขียนและอ่านทั้งสองอย่าง
อา	โหมดนี้เปิดไฟล์สำหรับการต่อท้าย โดยเริ่มจากส่วนท้ายของเอกสาร
a+	โหมดนี้จะเปิดไฟล์สำหรับการต่อท้ายและการอ่าน โดยเริ่มจากจุดสิ้นสุดของ เอกสาร.

'สวัสดีชาวโลก!\nการเขียนโปรแกรม Python สำหรับ Arduino\nบาย\n'

> > >ง. ปิด()

ด้วยวิธีนี้ เนื้อหาทั้งหมดของไฟล์จะถูกเก็บไว้ในหน่วยความจำ
หากต้องการอ่านเนื้อหาทีละบรรทัด ให้ใช้เมธอด readlines():

> > >D= เปิด ('test.txt', 'r')

> > >X =D.readlines()

> > >พิมพ์ X

['สวัสดีชาวโลก!\n', 'การเขียนโปรแกรม Python สำหรับ Arduino\n', 'ลาก่อน\n']

> > >ง. ปิด()

9.2 ไฟล์ CSV

ไฟล์ CSV ใช้เพื่อเก็บข้อมูลใน Python ตัวเขียน CSV ใช้เพื่อเขียนข้อมูลบนไฟล์ CSV และผู้อ่านอ่านด้วยคำสั่งง่ายๆ:

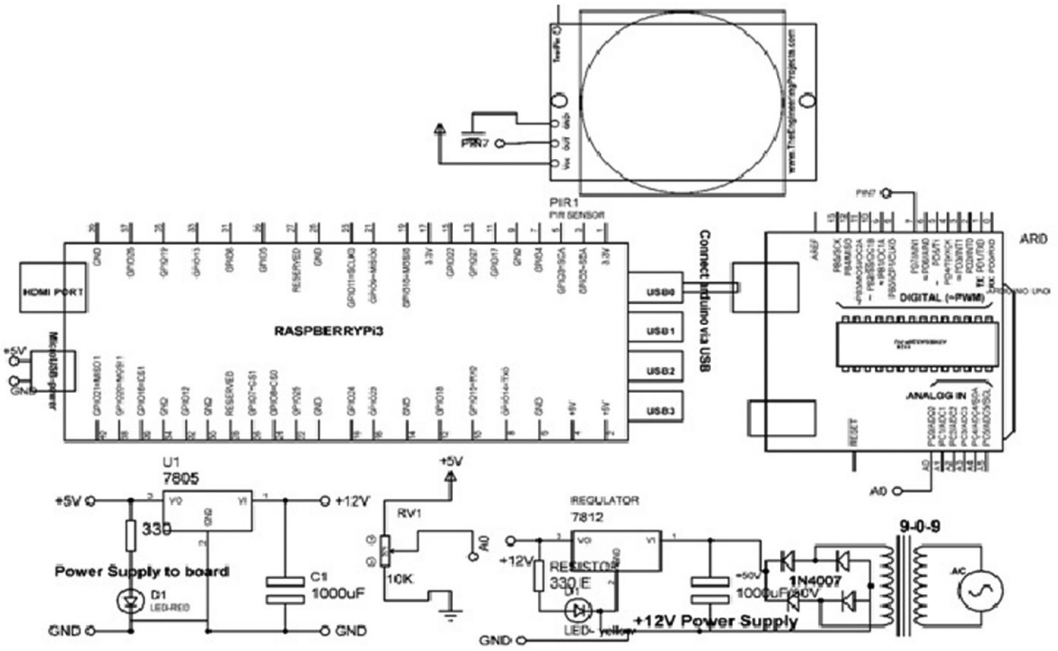
```
นักเขียน CSV
    นำเข้า csv
    data = [[1, 2, 3], ['a', 'b', 'c'], ['Python', 'Arduino', 'Programming']]
    พร้อม open('example.csv', 'w ') เป็น f:
        w = csv.writer(ฉ)
        สำหรับแถวในข้อมูล:
            w.writeow(แถว)
```

โปรแกรมอ่าน CSV

```
นำเข้า csv
ด้วย open('example.csv', 'r') เป็นไฟล์:
    r = csv.reader(le)
    สำหรับแถวใน r:
        พิมพ์แถว
```

9.3 การจัดเก็บข้อมูล Arduino ด้วยไฟล์ CSV

ไฟล์ CSV สามารถใช้เก็บข้อมูลทางประสาทสัมผัสจาก Arduino ได้ เพื่อทำความเข้าใจแนวคิด จะมีการหารือเกี่ยวกับระบบ ระบบประกอบด้วย Raspberry Pi, Arduino, เซ็นเซอร์อินฟราเรดแบบไพโรอิเล็กทริก (PIR) (เซ็นเซอร์ดิจิตอล), โพเทนชิออมิเตอร์ (POT) (เป็นเซ็นเซอร์อะนาล็อก) และแหล่งจ่ายไฟ มีวัตถุประสงค์เพื่อเก็บข้อมูลทางประสาทสัมผัสด้วยไฟล์ CSV และ Arduino ที่เชื่อมต่อกับ Pythonรูปที่ 9.1แสดงแผนภาพวงจรของระบบ



รูปที่ 9.1
แผนภาพวงจรเชื่อมต่อเซ็นเซอร์ PIR และ POT กับ Arduino UNO และ Pi

การเชื่อมต่อ:

- เชื่อมต่อขั้วหนึ่งของ POT กับ +5 V อีกขั้วหนึ่งเข้ากับกราวด์ และปัดน้ำฝนเพื่อยึด (A0) ของ Arduino Uno
- เชื่อมต่อพิน (Vcc และกราวด์) ของเซ็นเซอร์ PIR กับ +5 VDC และกราวด์ตามลำดับ
- ต่อพิน (OUT) ของเซ็นเซอร์ PIR เข้ากับพิน (7) ของ Arduino Uno
- เชื่อมต่อ Arduino Uni กับ Raspberry Pi ผ่าน USB

9.3.1 สูตร

```
นำเข้า csv # นำเข้าไลบรารี CSV
    นำเข้า pyrmata # นำเข้าห้องสมุด pyrmata นำเข้า
    เวลารอ # นำเข้าบอร์ดห้องสมุดเวลา =
    pyrmata.Arduino ('/dev/ttyUSB0') มัน =
    pyrmata.util.Iterator (บอร์ด)
    it.start() # start iterator
    PIR_pin = board.get_pin('d:7:i') # เชื่อมต่อเซ็นเซอร์ PIR กับพิน 7 เป็นอินพุต
    POT_pin = board.get_pin('a:0:i') # เชื่อมต่อ POT กับพิน 0 เป็นอินพุตโดยเปิด
    ('SensorDataStore .csv', 'w') เป็น f:
        w = csv.writer(ฉ)
        ว. writerow(["Number", "Potentiometer", "Motion sensor"]) i =
        0
        PIR_Data = PIR_pin.read () # อ่านเซ็นเซอร์ PIR
        POT_Data = POT_pin.read () # อ่าน POT ในขณะที
        ฉัน < 25:
            นอน(1)
            ถ้า PIR_Data ไม่มี:
                ผม += 1
                แถว = [i, POT_Data, PIR_Data]
                w. นักเขียน(แถว)
    พิมพ์ "เสร็จสิ้นขั้นตอน CSV พร้อมแล้ว!"
    board.exit()
```

9.4 การพล็อตตัวเลขสุ่มโดยใช้ Matplotlib

การติดตั้ง matplotlib เป็นกระบวนการที่ง่ายบน Ubuntu ด้วยคำสั่งง่ายๆ:

```
$ sudo apt-get ติดตั้ง python-matplotlib
```

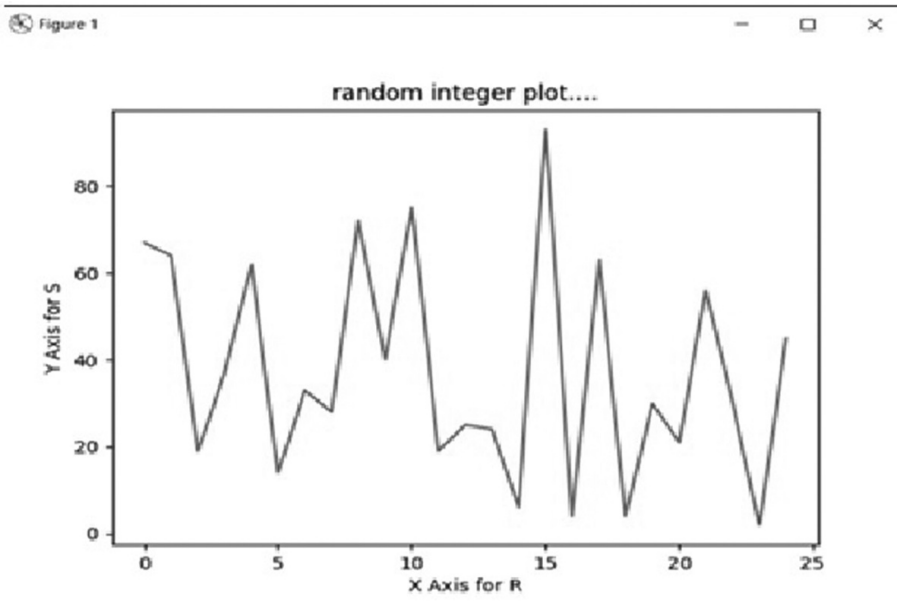
คลิกที่ "ใช่" เมื่อได้รับแจ้งให้ติดตั้งการพึ่งพา ไลบรารี matplotlib จัดเตรียมเมธอด plot() เพื่อสร้างแผนภูมิเส้น เมธอด plot() ใช้รายการหรือโครงสร้างข้อมูลอาร์เรย์ที่ประกอบด้วยตัวเลขจำนวนเต็มหรือจุดอ้างอิงเป็นอินพุต Plot() ใช้ค่าสำหรับx-แกนและย-แกน ถ้าให้สองอาร์เรย์เป็นอินพุต หากมีการระบุรายการหรืออาร์เรย์เพียงรายการเดียวเป็นอินพุต plot() จะถือว่าค่าสำหรับย-แกนและสร้างค่าที่เพิ่มขึ้นโดยอัตโนมัติสำหรับx-แกน:

```
pyplot.plot(x, y)
```

ในการเปลี่ยนรูปแบบของเส้นและตัวสร้างด้วยสีที่ต่างกัน สามารถใช้เมธอด plot() เช่น สำหรับคำสั่งรูปแบบเส้นทึบ:

```
pyplot.plot(x, y, '-')
```

รูปที่ 9.2แสดงการพล็อตของตัวเลขสุ่ม



รูปที่ 9.2
การพล็อตตัวเลขสุ่ม

9.4.1 สูตร

สำหรับการสร้างและพล็อตตัวเลขสุ่ม:

```
นำเข้าสุ่ม
R = ช่วง (0,25)
S = [random.randint(0,100) สำหรับ r ในช่วง
(0,25)] รูปที่ 1 = pyplot.gure()
pyplot.plot(R, S, '-')
pyplot.title('พล็อตจำนวนเต็มแบบสุ่ม....')
pyplot.xlabel('แกน X สำหรับ R')
pyplot.ylabel('แกน Y สำหรับ S')
pyplot.show()
```

9.5 พล็อตเรียลไทม์จาก Arduino

การวางแผนข้อมูลแบบเรียลไทม์จาก Arduino เป็นงานสำคัญที่ข้อมูลทางประสาทสัมผัสมีความสำคัญ เพื่อทำความเข้าใจแนวคิด ค่า POT แบบเรียลไทม์จะกล่าวถึงในส่วนนี้ ระบบประกอบด้วย Raspberry Pi, Arduino, POT และแหล่งจ่ายไฟ Arduino เชื่อมต่อกับ Raspberry Pi ผ่าน USB ข้างหนึ่งของ POT เชื่อมต่อกับ +5 V อีกข้างหนึ่งกับกราวด์ และที่ปัดน้ำฝนเชื่อมต่อกับพิน (A0) ของ Arduino (รูปที่ 9.3). หากต้องการพล็อตข้อมูลแบบเรียลไทม์ ให้เลื่อนปุ่ม POT และตรวจสอบผลลัพธ์

การพล็อตแบบเรียลไทม์สามารถทำได้โดยใช้การรวมกันของฟังก์ชันไพพล็อต ion(), draw(), set_xdata() และ set_data() วิธีการ ion() ใช้เพื่อเริ่มต้นโหมดโต้ตอบของ pyplot สิ่งนี้ช่วยในการเปลี่ยน .แบบไดนามิกxและyค่าของแปลงใน gure:

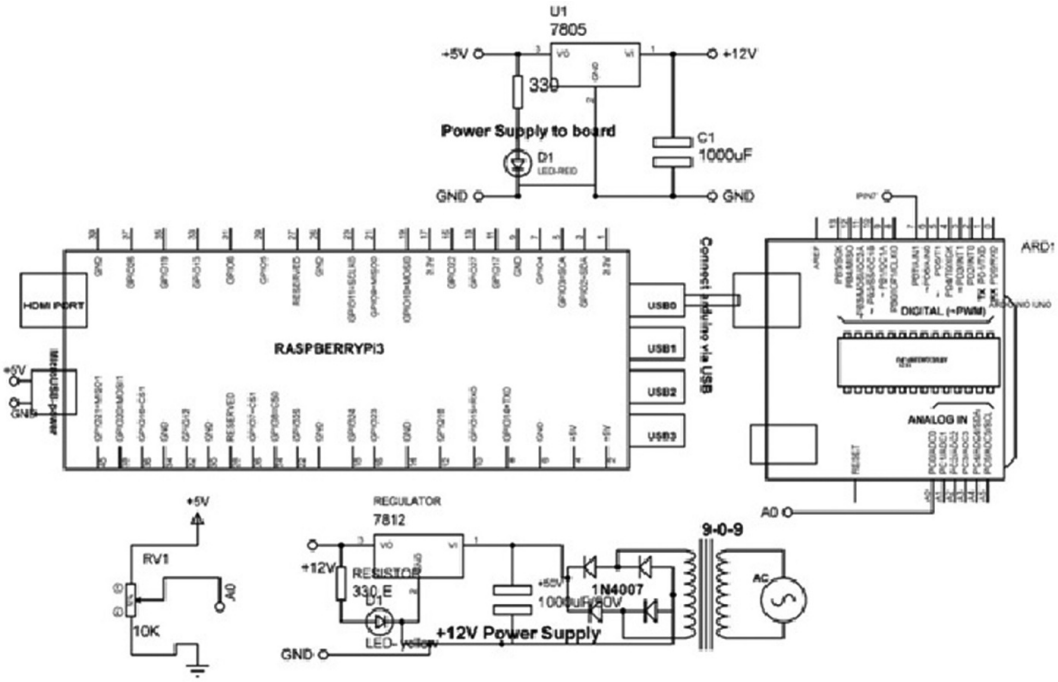
```
pyplot.ion()
```

เมื่อตั้งค่าโหมดโต้ตอบแล้ว พล็อตจะถูกวาดโดยการเรียกเมธอด draw() ตอนนี้ เริ่มต้นพล็อตด้วยชุดข้อมูลว่าง 0 ในกรณีนี้:

```
pData = [0] * 25
```

ในอาร์เรย์นี้สำหรับyค่าพีดาต้า,ใช้เพื่อผนวกค่าจากเซ็นเซอร์ในวง while เพื่อต่อท้ายค่าใหม่ล่าสุดไปยังอาร์เรย์ข้อมูลนี้ และวาดพล็อตใหม่ด้วยอาร์เรย์ที่อัปเดตเหล่านี้สำหรับxและyค่า

```
pData.append(oat(a0.read()))
เดล pData[0]
```



รูปที่ 9.3
แผนภาพวงจรสำหรับเชื่อมต่อ POT กับ Arduino

เมธอด set_xdata() และ set_ydata() ใช้เพื่ออัปเดตxและyข้อมูลแกน

```
l1.set_xdata([i for i in xrange(25)])
l1.set_ydata(pData) # อัปเดตข้อมูล
pyplot.draw() # อัปเดตพล็อต
```

ข้อมูลโค้ด [i for i ใน xrange(25)] คือการสร้างรายการตัวเลขจำนวนเต็ม 25 ตัวที่จะเริ่มต้นทีละน้อยที่ 0 และสิ้นสุดที่ 24

9.5.1 สูตร

```
// จาก matplotlib นำเข้า pyplot
นำเข้า pyrmata # นำเข้าเวลานำเข้าไลบรารี
pyrmata เป็นรอ
กระดาน = pyrmata.Arduino('/dev/ttyUSB0')
wait.sleep(5) # wait for 5 Sec
มัน = pyrmata.util.Iterator(บอร์ด)
it.start() # start iterator
POT_pin = board.get_pin('a:0:i') # เชื่อมต่อ POT กับพิน A0 เป็นอินพุต
```

```
pyplot.ion()
pData = [0.0] * 25 g
= pyplot.gure()
pyplot.title('พล็อตข้อมูลเรียลไทม์จาก POT')
ax1 = pyplot.axes()
l1, = pyplot.plot(pData)
pyplot.ylim([0, 1])

ในขณะที่จริง:
  ลอง:
รอ.นอน(1)
      pData.append(oat(POT_pin.read()))
      pyplot.ylim([0, 1])
      เดล พีดาต้า[0]
      l1.set_xdata([i for i in xrange(25)])
      l1.set_ydata(pData) # อัปเดตข้อมูล
      pyplot.draw() # อัปเดตพล็อตยกเว้น
      KeyboardInterrupt:
      board.exit()
      หยุดพัก
```

9.6 การรวมพล็อตใน Tkinter Window

มาตรา 9.5อธิบายวิธีการวาดพล็อตสำหรับข้อมูลทางประสาทสัมผัสอย่างต่อเนื่องจาก Arduino ด้วยความช่วยเหลือของ POT Python มีความสามารถในการรวมเข้ากับไลบรารี matplotlib และอินเทอร์เฟซกราฟิก Tkinter สำหรับวงจรเดียวกันกับรูปที่ 9.3จะมีการกล่าวถึงการรวมนี้ โปรแกรมใช้การเชื่อมต่อของ Tkinter กับ matplotlib

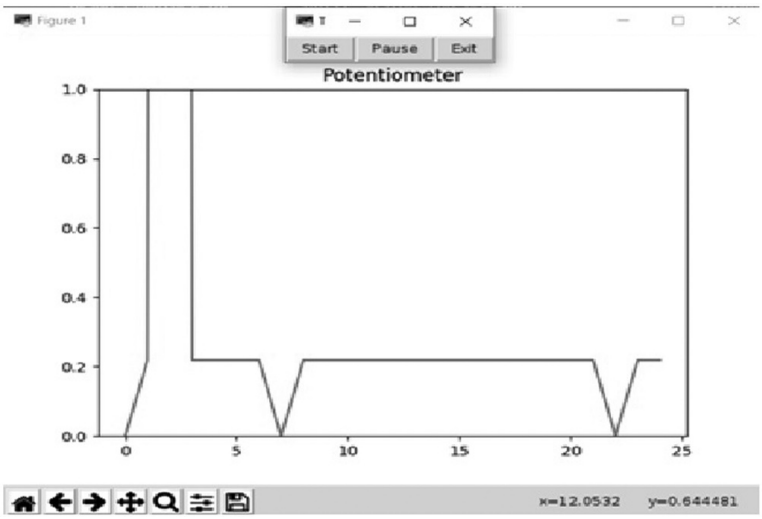
9.6.1 สูตร

```
นำเข้า sys # นำเข้า sys ไลบรารี
จาก matplotlib นำเข้า pyplot # นำเข้าไลบรารี นำเข้า
pyrmat # นำเข้าไลบรารี
นำเข้าเวลาที่รอ # ห้องสมุดเวลานำเข้า นำเข้า
Tkinter # นำเข้าห้องสมุด
```

```
    def start_button_press():
      ในขณะที่จริง:
          ถ้า FLAG.get():
  wait.sleep(1) # รอ 1 วินาที
            pData.append(oat(POT_pin.read()))
            pyplot.ylim([0, 1])
            เดล พีดาต้า[0]
            l1.set_xdata([i for i in xrange(25)])
            l1.set_ydata(pData) # อัปเดตข้อมูล
            pyplot.draw() # อัปเดตพล็อต
            TOP.update()
          อืน:
            FLAG.set(nรู)
            หยุดพัก
def หยุดชั่วคราว_button_press():
  FLAG.set(เท็จ)
def exit_button_press():
  พิมพ์ "ออกจากการบันทึกข้อมูล..." หยุด
ชั่วคราว_button_press()
  board.exit()
  pyplot.close(g)
  TOP.เลิก()
  TOP.ทำลาย()
  พิมพ์ "เสร็จแล้ว....."
  sys.exit()
บอร์ด = pyrmata.Arduino('/dev/ttyUSB0')
# ใช้เธรด iterator เพื่อหลีกเลี่ยงบัฟเฟอร์ overow =
pyrmata.util.Iterator (บอร์ด)
it.start() # start iterator
# กำหนดบทบาทและตัวแปรให้กับขาอะนาล็อก 0
POT_pin = board.get_pin('a:0:i') # อ่าน POT
#ผ้าใบ Tkinter
ด้านบน = Tkinter.Tk()
TOP.title("Tkinter + matplotlib")
# สร้าง ag เพื่อทำงานกับ indenite while loop
FLAG = Tkinter.BooleanVar(TOP)
```

```
FLAG.set(nฐ)
pyplot.ion()
pData = [0.0] * 25 gure
= pyplot.gure()
pyplot.title ('โพเทนชิออมิเตอร์')
ax1 = pyplot.axes ()
l1, = pyplot.plot(pData)
pyplot.ylim([0, 1])
# สร้างปุ่มเริ่มและเชื่อมโยงกับวิธีการกดปุ่มเริ่มต้น
start_Button = Tkinter.Button (บนสุด text="Start" คำสั่ง = start_
    button_press)
start_Button.grid(คอลัมน์=1,แถว=2)
# สร้างปุ่มหยุดและเชื่อมโยงกับวิธีการกดปุ่มหยุดชั่วคราว
Pause_Button = Tkinter.Button (บนสุด, ข้อความ = "หยุดชั่วคราว", คำสั่ง = หยุดชั่วคราว_
    button_press)
# สร้างปุ่มออกเพื่อออกจากหน้าต่าง
exit_Button=Tkinter.Button(TOP,text="Exit",command=exit_button_press)
exit_Button.grid(column=3, row=2)
TOP.mainloop()
```

รันโปรแกรมและหน้าต่างจะปรากฏขึ้นบนหน้าจอ (รูปที่ 9.4). พล็อตสามารถควบคุมได้โดยใช้ปุ่ม "เริ่ม", "หยุดชั่วคราว" และ "ออก" คลิกที่ปุ่มเริ่มต้นและหมุนปุ่ม POT และดูการเปลี่ยนแปลงในพล็อต กระบวนการสามารถหยุดชั่วคราวหรือปิดโปรแกรมด้วยปุ่ม "ออก"



รูปที่ 9.4
พล็อตสำหรับข้อมูลเรียลไทม์จาก Arduino