Python e Arduino: Analisando dados em tempo real.

Dr. Eduardo S. Pereira.
@duducosmos/ pereira.somoza@gmail.com
https://github.com/duducosmos

01/04/2017

Introdução

A linguagem Python

A linguagem Python

- Python é uma linguagem interpretada desenvolvida por Guido Van Rossum em 1991;
- É multiparadigma, porém tudo em python é objeto;
- É livre, aberta e com baterias incluídas;
- Versões 2.7.X e 3.X; (Existem projetos que não foram migrados - E que provavelmente não serão em python 2.X);

A linguagem Python

A linguagem Python

- Python é uma linguagem interpretada desenvolvida por Guido Van Rossum em 1991;
- É multiparadigma, porém tudo em python é objeto;
- É livre, aberta e com baterias incluídas;
- Versões 2.7.X e 3.X; (Existem projetos que não foram migrados - E que provavelmente não serão em python 2.X);

A linguagem Python

A linguagem Python

- Python é uma linguagem interpretada desenvolvida por Guido Van Rossum em 1991;
- É multiparadigma, porém tudo em python é objeto;
- É livre, aberta e com baterias incluídas;
- Versões 2.7.X e 3.X; (Existem projetos que não foram migrados - E que provavelmente não serão em python 2.X);

A linguagem Python

A linguagem Python

- Python é uma linguagem interpretada desenvolvida por Guido Van Rossum em 1991;
- É multiparadigma, porém tudo em python é objeto;
- É livre, aberta e com baterias incluídas;
- Versões 2.7.X e 3.X; (Existem projetos que não foram migrados - E que provavelmente não serão em python 2.X);

Medida de Temperatura com LM35

O experimento

Evolução da temperatura com o tempo

- LM35 - Sensor que apresenta saída de tensão linear proporcional à temperatura em que o mesmo se encontra, tendo em sua saída um sinal de 10mV para cada Grau Célsius de temperatura;
- Opera na faixa de $[-50^o, 150^0]$C.

O experimento

Evolução da temperatura com o tempo

- A saída do LM35 será conectada a porta analógica A0; O valor do sinal de A0 irá variar entre 0 a 1023, com 0 correspondendo a 0 Volts e 1023 correspondendo a 5 Volts.
- Como para cada variação de 1^oC o sensor registra uma mudança de 10mV, então:

$$V_{A0} = (V_{lido} * (5/1023)); \quad (1)$$

$$T = V_{A0}/0.01; \quad (2)$$

com V_{A0} a voltagem da saída do sensor, V_{lido} é o valor lido na porta analógica A0 e T é a temperatura em graus Célsius;

O experimento

Evolução da temperatura com o tempo

- A saída do LM35 será conectada a porta analógica A0; O valor do sinal de A0 irá variar entre 0 a 1023, com 0 correspondendo a 0 Volts e 1023 correspondendo a 5 Volts.
- Como para cada variação de 1^oC o sensor registra uma mudança de 10mV, então:

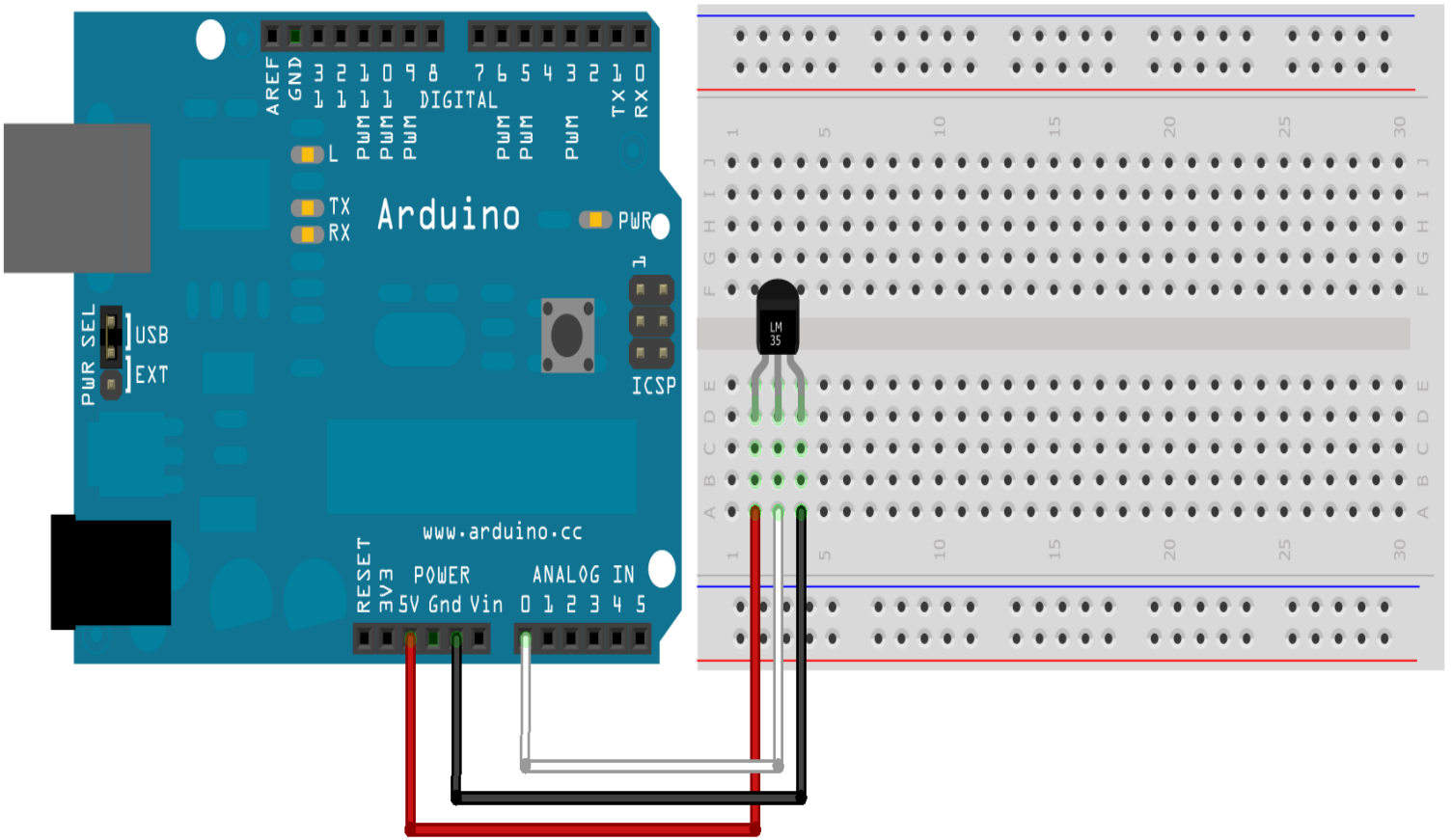
$$V_{A0} = (V_{lido} * (5/1023)); \quad (1)$$

$$T = V_{A0}/0.01; \quad (2)$$

com V_{A0} a voltagem da saída do sensor, V_{lido} é o valor lido na porta analógica A0 e T é a temperatura em graus Célsius;

O experimento

Conexão com Arduino



O código

LM35 v0.1

```
#include "Arduino.h"
const int LM35 = A0;
float temperatura;
unsigned long t0 = millis();
unsigned long t1 = t0;

void setup(){
    Serial.begin(9600);
}
void loop(){
    temperatura = (float(analogRead(LM35)) * 5 / 1023) / 0.01;
    if(millis() - t1 >= 1000) {
        t1 = millis();
        unsigned long t2 =  (t1 - t0) /  1000;
        Serial.print(t2, DEC);
        Serial.print(',');
        Serial.println(temperatura,DEC);
    }
}
```

Comunicação Serial - python-serial

Comunicação Serial

Comunicação Serial

- Para a comunicação serial é preciso instalar a biblioteca pyserial
- Para usuários de sistemas tipo debian: sudo apt-get install python-serial

Comunicação Serial

Comunicação Serial

- Para a comunicação serial é preciso instalar a biblioteca pyserial
- Para usuários de sistemas tipo debian: sudo apt-get install python-serial

O código

pyserial

```
#/usr/bin/env python
#-*- coding: UTF-8 -*-
'''
Sistema de comunicacao via porta serial com arduino.
'''
_author = "E. S. Pereira"
_date = "31/03/2017"
_version= "0.0.1"

import serial

ser = serial.Serial("/dev/ttyUSB0", 9600)
ser.write('5')
print ser.read()
```

Gerador de Gráficos em tempo Real

O código

Gerador de Gráficos Parte -1

```
#/usr/bin/env python
#-*- coding: UTF-8 -*-
'''
Sistema de comunicacao via porta serial com arduino.
'''
_author = "E. S. Pereira"
_date = "31/03/2017"
_version= "0.0.1"

import serial
import matplotlib.pyplot as plt
plt.ion()

ser = serial.Serial("/dev/ttyUSB0", 9600)
ser.read()
```

O código

Gerador de Gráficos Parte -2

```
def serOut(ser):
    tmp = ser.read()
    a = tmp
    while(tmp != '\n'):
        tmp = ser.read()
        a += tmp
    b = a.split('\r')[0]
    b = b.split(',')
    return b[0], b[1]
```

O código

Gerador de Gráficos Parte -3

```
class DynamicUpdate():
    # Suppose we know the x range
    min_x = 0
    max_x = 20

    def on_launch(self):
        # Set up plot
        self.figure, self.ax = plt.subplots(
        # subplot_kw=dict(projection='polar')
        )
        self.lines, = self.ax.plot([], [], 'o')
        # Autoscale on unknown axis and known lims on the other
        self.ax.set_autoscaley_on(True)
        #self.ax.set_xlim(self.min_x, self.max_x)
        #self.ax.set_ylim(self.min_x, self.max_x)
        # Other stuff
        self.ax.grid()
```

O código

Gerador de Gráficos Parte -4

```
def on_running(self, xdata, ydata):
    # Update data (with the new _and_ the old points)
    self.lines.set_xdata(xdata)
    self.lines.set_ydata(ydata)
    # Need both of these in order to rescale
    self.ax.relim()
    self.ax.autoscale_view()
    # We need to draw *and* flush
    self.figure.canvas.draw()
    self.figure.canvas.flush_events()
```

O código

Gerador de Gráficos Parte -5

```
def __call__(self):
    import numpy as np
    import time
    self.on_launch()
    xdata = []
    ydata = []
    i = 0
    for x in np.arange(0, 100000, 1):

        x0, y0 = serOut(ser)

        if(len(x0) <= 6):
            if(y0 != '' and x0 != ''):
                print(x0, y0)
                xdata.append(float(x0))
                ydata.append(float(y0))
                self.on_running(xdata, ydata)
    return xdata, ydata
```

O código

Gerador de Gráficos Parte - 6

```
if(__name__ == "__main__):
    d = DynamicUpdate()
    d()
```

Código Disponível GitHub

Código Disponível GitHub

- https://github.com/duducosmos/arduinoday2017

Suavização de Dados Coletados

Suavização de Dados de sensores

Suavização de Dados de sensores

- Ao invés de coletar o dado direto, iremos passar o valor médio dos últimos N dados coletados;
- Lembrando que o valor médio representa o valor esperado, ou o valor mais provável de ser o correto;

Suavização de Dados de sensores

Suavização de Dados de sensores

- Ao invés de coletar o dado direto, iremos passar o valor médio dos últimos N dados coletados;
- Lembrando que o valor médio representa o valor esperado, ou o valor mais provável de ser o correto;

O código

LM35 v0.1 - Parte 1

```
#include "Arduino.h"
const int LM35 = A0;
float temperatura;
unsigned long t0 = millis();
unsigned long t1 = t0;

// Smooth Temp Data
const int readsSize = 30;
unsigned int readIndex = 0;
unsigned int reads[readsSize];
unsigned int totalReads;
unsigned int averageReads;
```

O código

LM35 v0.1 - Parte 2

```
void smoothTemp(){
    totalReads = totalReads - reads[readIndex];

    reads[readIndex] = analogRead(LM35);

    totalReads = totalReads + reads[readIndex];
    readIndex += 1;
    if(readIndex >= readsSize) {
        readIndex = 0;
    }

    averageReads = totalReads / readsSize;

}
```

O código

LM35 v0.1 - Parte 3

```
void setup(){
    Serial.begin(9600);
    for(int i = 0; i < readsSize; i++ ) {
    reads[i] = 0;
    }
}
```

O código

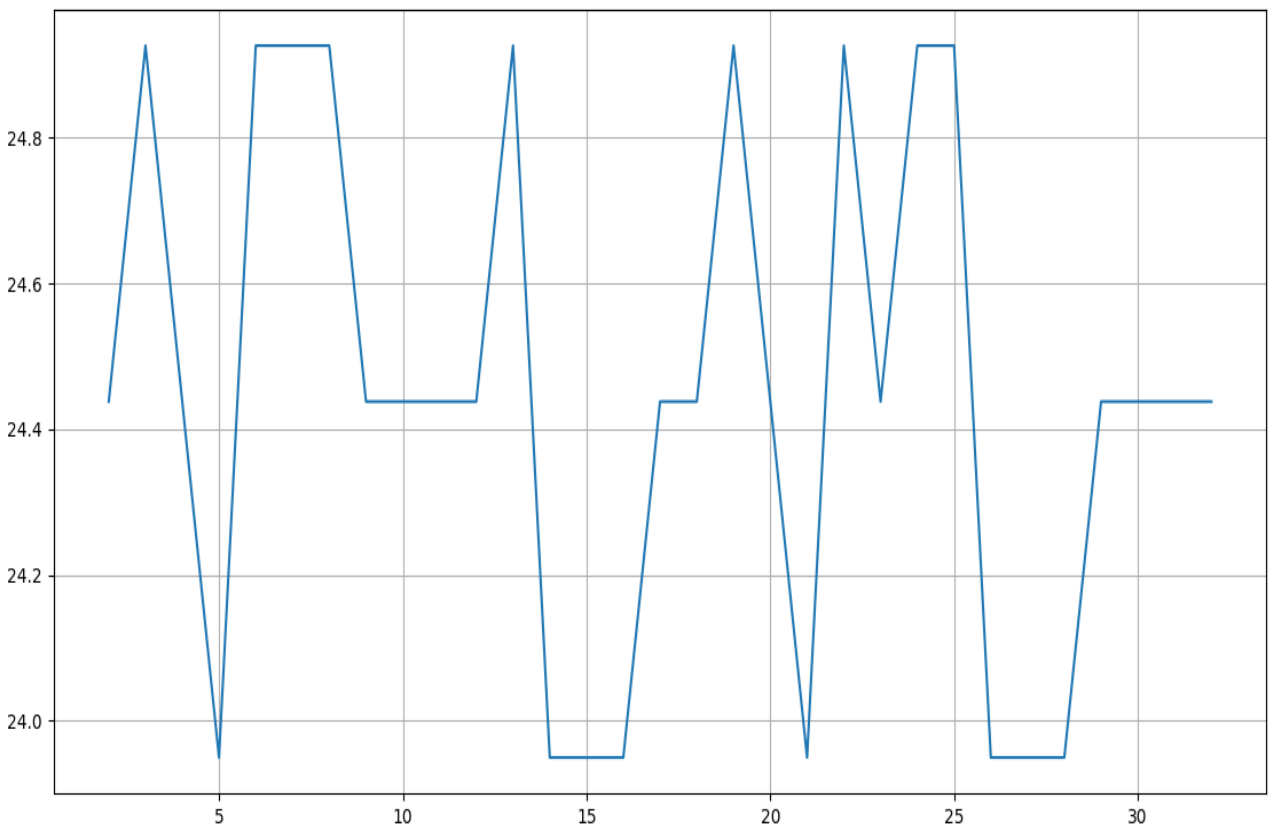
LM35 v0.1 - Parte 3

```
void loop(){
    //temperatura = (float(analogRead(LM35)) * 5 / 1023) / 0.01;

    smoothTemp();

    temperatura = (float(averageReads) * 5 / 1023) / 0.01;

    if(millis() - t1 >= 1000) {
        t1 = millis();
        unsigned long t2 =  (t1 - t0) /  1000;
        Serial.print(t2, DEC);
        Serial.print(',');
        Serial.println(temperatura,DEC);
    }
}
```

Resultados

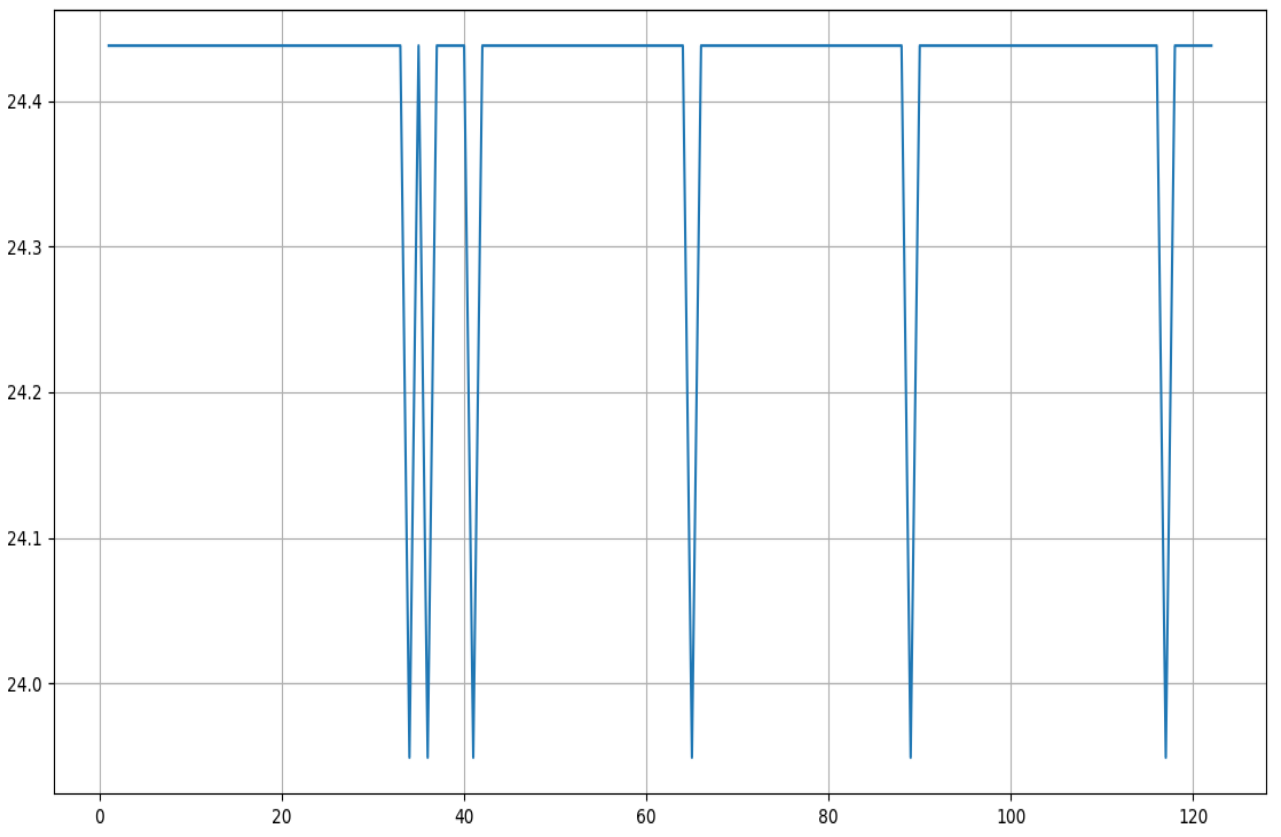
Leitura de Dados Não Suavizados

Comunicação Serial



Leitura de Dados Suavizados

Comunicação Serial



FIM

FIM

- MUITO OBRIGADO.