## Introduzione

In questo documento spiego come convertire i dati ricevuti dai sensori inerziali in dati di roll pitch e yaw usando le formule spiegate in “Stima orientamento di un corpo.docx”. Quello che spiego qui è alla base del progetto [Sensors To RPY Client](https://github.com/riccardo-florio/Sensors_To_RPY_Client).

## Strumenti utilizzati

App: ServerSensor

Link: <https://github.com/umer0586/SensorServer>

Altri riferimenti:

* Script python per comunicare con l’app: <https://stackoverflow.com/questions/68127440/how-do-i-transmit-real-time-android-sensor-data-to-a-pc-wirelessly>
* Per creare l’ambiente virtuale in python: <https://www.youtube.com/watch?v=S6LMNLKhaEM&t=470s>
* Libreria web: <https://github.com/websocket-client/websocket-client>

Versione Python: 3.11.3

Editor: PyCharm 2022.2.1

## Procedimento

Innanzitutto, dopo aver installato l’applicazione ServerSensor sul cellulare, ho creato la cartella "sersor\_client", l’ho aperta con l’editor e ho creato ed attivato un ambiente virtuale. Quindi, ho selezionato l’interpreter python desiderato (quello della versione 3.11.1).

I dati dei sensori estratti dall’applicazione vengono trasmessi mediante il protocollo WebSocket sotto forma di file json.

In prima istanza creiamo un semplice script python che riceve e mostra a video i dati estratti: uso il codice specificato nella documentazione all’interno nel file "main.py"

Si noti che, affinché tutto funzioni correttamente, deve essere installata la libreria "websocket-client".

Quando tutto funziona, procedo col ricavare le rotazioni di Roll – Pitch – Yaw a partire dai dati estratti dai sensori del cellulare. Un primo codice per estrarre dati di roll e pitch sfruttando l’accelerometro è il seguente:

import websocket  
import json  
import math  
  
# IP del dispositivo connesso  
device\_ip = "ws://192.168.1.75:8080"  
# Inizializzazione di RPY  
roll, pitch, yaw = 0, 0, 0  
# Campo magnetico di riferimento  
  
def on\_message(ws, message):  
 # Estrazione dei dati inviati dallo smartphone  
 values = json.loads(message)['values']  
 type = json.loads(message)['type']  
  
 # Estraggo roll e pitch dall'accelerometro  
 global roll, pitch, yaw  
 if type == 'android.sensor.accelerometer':  
 a\_sx = values[0]  
 a\_sy = values[1]  
 a\_sz = values[2]  
 roll = math.atan2(a\_sy, a\_sz)  
 pitch = math.atan2(a\_sx, math.sqrt(pow(a\_sy, 2) + pow(a\_sz, 2)))  
  
 if type == 'android.sensor.magnetic\_field':  
 pass  
  
 print("roll = ", roll, "pitch = ", pitch, "yaw = ", yaw)  
  
  
def on\_error(ws, error):  
 print("error occurred")  
 print(error)  
  
  
def on\_close(ws, close\_code, reason):  
 print("connection close")  
 print("close code : ", close\_code)  
 print("reason : ", reason)  
  
  
def on\_open(ws):  
 print("connected")  
  
  
def connect(url):  
 ws = websocket.WebSocketApp(url,  
 on\_open=on\_open,  
 on\_message=on\_message,  
 on\_error=on\_error,  
 on\_close=on\_close)  
  
 ws.run\_forever()  
  
  
connect(device\_ip + '/sensors/connect?types=[' +  
 '"android.sensor.accelerometer",' +  
 '"android.sensor.gyroscope",' +  
 '"android.sensor.magnetic\_field"]')

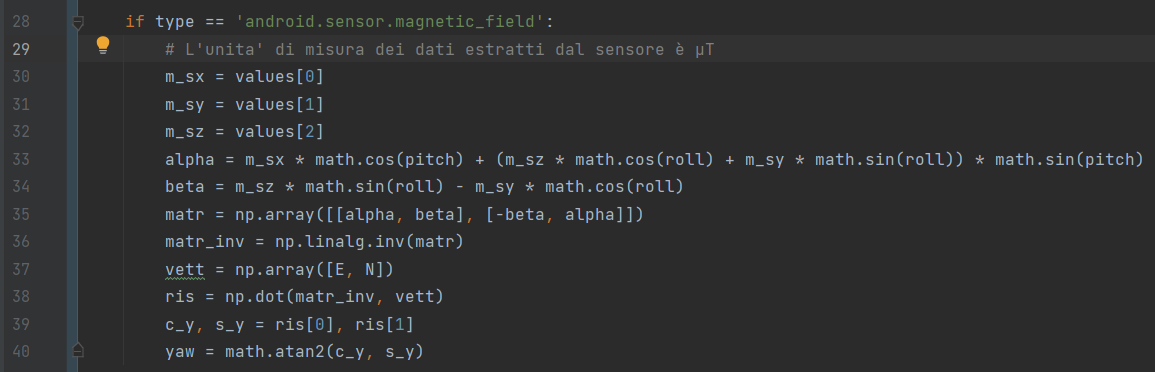
Per ricavare pitch dobbiamo sfruttare il magnetometro. Oltre ai dati da esso estratti, sono necessari i dati del sistema di riferimento globale. Li ricaviamo dal sito indicato di seguito inserendo latitudine e longitudine della zona in cui si trova il sensore, oppure inserendo il cap.

<https://www.ngdc.noaa.gov/geomag/calculators/magcalc.shtml#igrfwmm>

Ottenuti i dati dell’ENU, li converto in microtesla e li memorizzo:

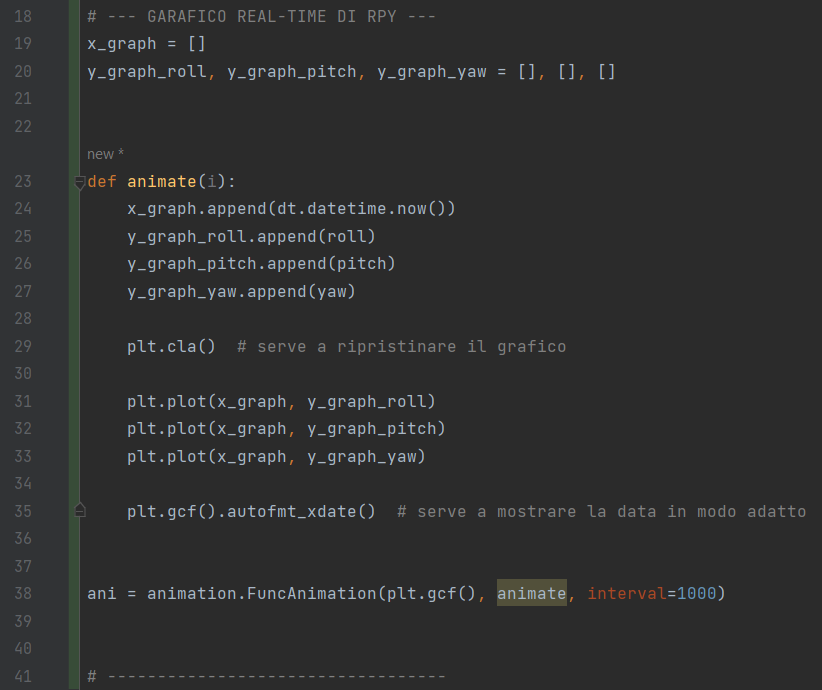


A questo punto applico le formule e calcolo YAW:

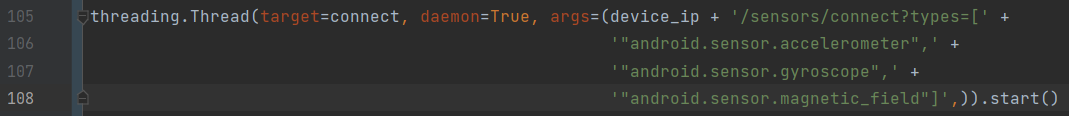


## Grafico real – time di RPY

Per questo scopo usiamo la libreria "matplotlib". Implementiamo un meccanismo per cui ogni secondo viene aggiornato il grafico con i nuovi valori rilevati dai sensori. Il problema incontrato in questa fase è che, per poter rilevare i dati dai sensori e contemporaneamente graficarli, bisogna eseguire due funzioni, "connect(…)" e "plt.show", in maniera concorrente. Eseguiamo la prima in un thread separato da quello principale.



Di seguito, il codice per eseguire il rilevamento dei dati in un thread secondario:



## Applicazione del filtro

A questo punto possiamo notare come i dati dei sensori siano abbastanza rumorosi. Possiamo filtrarli con il seguente filtro passa – basso:

Dove:

* è il valore filtrato al tempo k
* è il valore filtrato al tempo precedente.
* è il valore attuale del segnale (ad esempio, roll, pitch o yaw).
* è un coefficiente che determina il grado di smussamento.