

Nama : I Dewa Ayu Indira Wulandari Chrisna

NIM : 1301204152

Kelas : IF-44-04 / IF-43-PIL-CPS01

Bagian 1: Teori MQTT

1.

- a. Dengan mengasumsikan bahwa tidak ada device lain yang menggunakan frekuensi jaringan yang sama, saya akan menggunakan Bluetooth, karena data yang dihasilkan dan harus dikirimkan ke cloud adalah data numerik yang ukurannya relatif kecil sehingga Bluetooth saja cukup. Selain itu, Bluetooth dirancang untuk menggunakan lebih sedikit daya (power), sehingga untuk pengiriman data dalam jangka panjang dan akurat, Bluetooth lebih efektif karena bisa langsung terhubung ke ponsel dan memerlukan daya yang lebih sedikit.
- b. MQTT-SN, karena sesuai namanya, SN merupakan singkatan dari Sensor Networks, yang berarti versi MQTT ini memang dirancang khusus untuk melakukan pengiriman data sensor ke cloud dengan menggunakan lebih banyak pilihan jaringan, seperti ZigBee, Bluetooth, dan RF. Sesuai dengan pilihan protokol komunikasi yang disebutkan pada poin a, Bluetooth bisa digunakan pada MQTT-SN.
- c. Bisa. Pada bahasa pemrograman python, penulisan kode MQTT-SN tidak jauh berbeda dengan penulisan kode MQTT. Seperti pembuatan kode publisher MQTT pada umumnya, yang pertama harus dilakukan adalah mendefinisikan host dan port yang akan digunakan dalam pengiriman data ini. Kemudian, definisikan data yang didapatkan dari kedua sensor tersebut, lalu definisikan topik yang berbeda untuk kedua sensor tersebut. Untuk mengirimkan 2 data dari 2 sensor yang berbeda, dapat dilakukan dengan menggunakan fungsi `publish()` pada `library paho.mqtt.publish`. Caranya sama seperti MQTT biasa, hanya saja nama topik yang digunakan sebagai parameter harus diperhatikan sesuai data, sensor, dan topik tujuan pengiriman data tersebut.

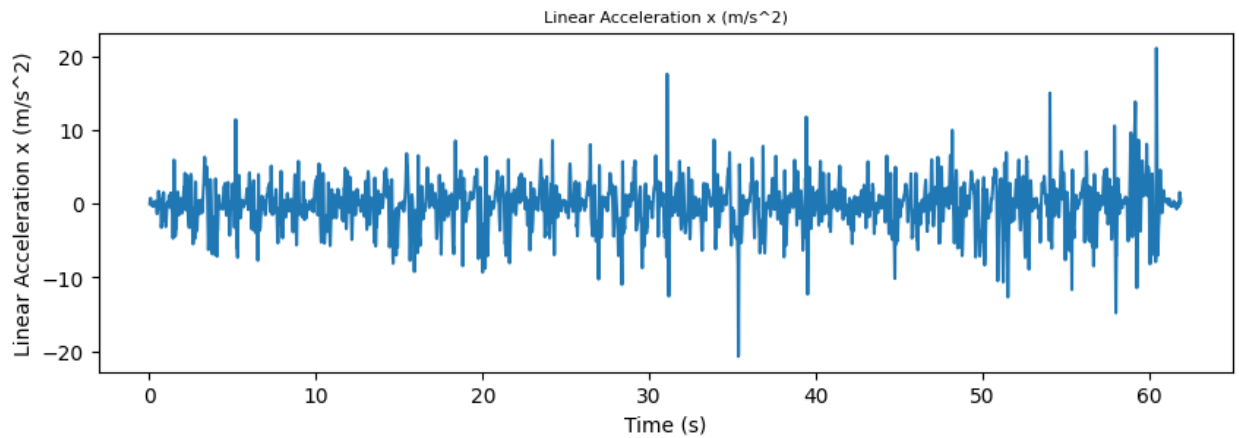
2.

- a. X merupakan broker yang berperan sebagai penerima dan yang merutekan data antar perangkat yang digunakan dalam sistem IoT tersebut. X akan menerima data yang dikirimkan oleh mikrokontroler. X (broker) dapat melakukan pemrosesan data, seperti filtering atau transformasi pesan. X juga dapat menyimpan cache data. Kemudian, data akan dikirimkan dari X (broker) ke MQTT client melalui jaringan internet.
- b. X berperan sebagai publisher, dan terjadi ketika data diterima dari sensor dan telah selesai dilakukan pemrosesan tambahan (jika diperlukan). Kemudian, X akan melakukan publishing data sesuai topik yang telah ditentukan.

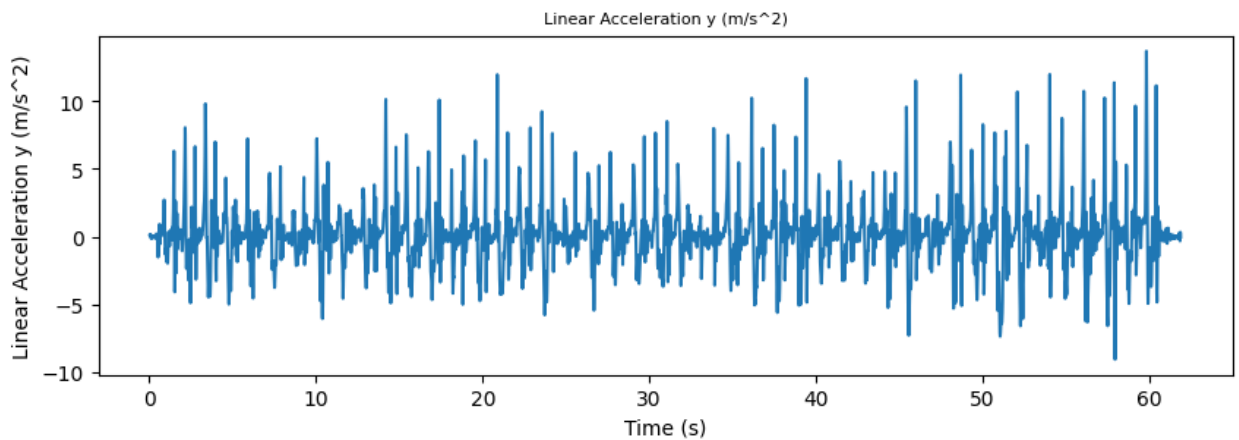
Bagian 2: Eksperimen Data Fusion

3.

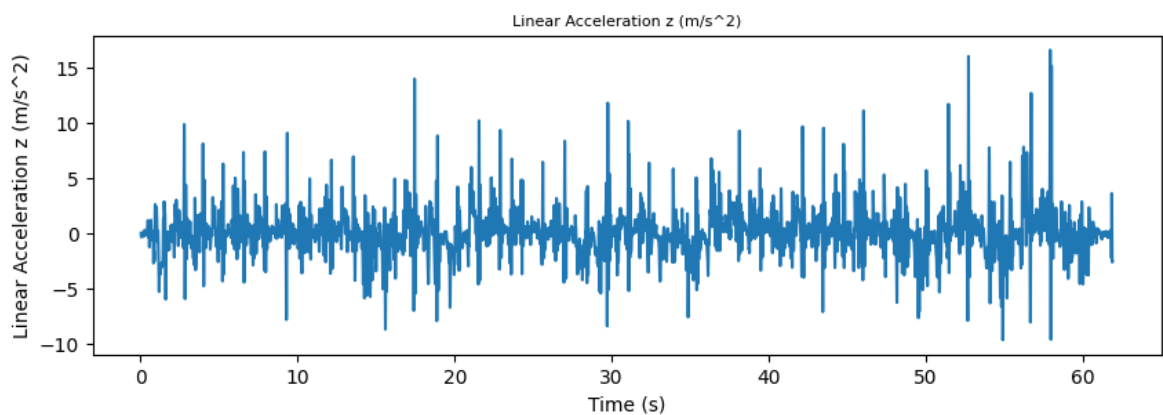
- Plots (Accelerometer)



Seperti yang terlihat pada gambar, data sumbu x pada accelerometer memiliki range antara ± 21 - ± 21 . Dengan nilai maksimum dari data tersebut adalah $21,10 \text{ m/s}^2$.



Seperti yang terlihat pada gambar, data sumbu y pada accelerometer memiliki range antara ± 13 - ± 13 . Dengan nilai maksimum dari data tersebut adalah $13,71 \text{ m/s}^2$.



Seperti yang terlihat pada gambar, data sumbu z pada accelerometer memiliki range antara $\pm 16 - \pm 16$. Dengan nilai maksimum dari data tersebut adalah $16,57\text{m/s}^2$.

- Code

```
[3] # Importing accelerometer data
accelData = pd.read_excel(r'/content/Acceleration without g 2023-04-15 13-07-26.xls')
print(accelData)

print("")
```

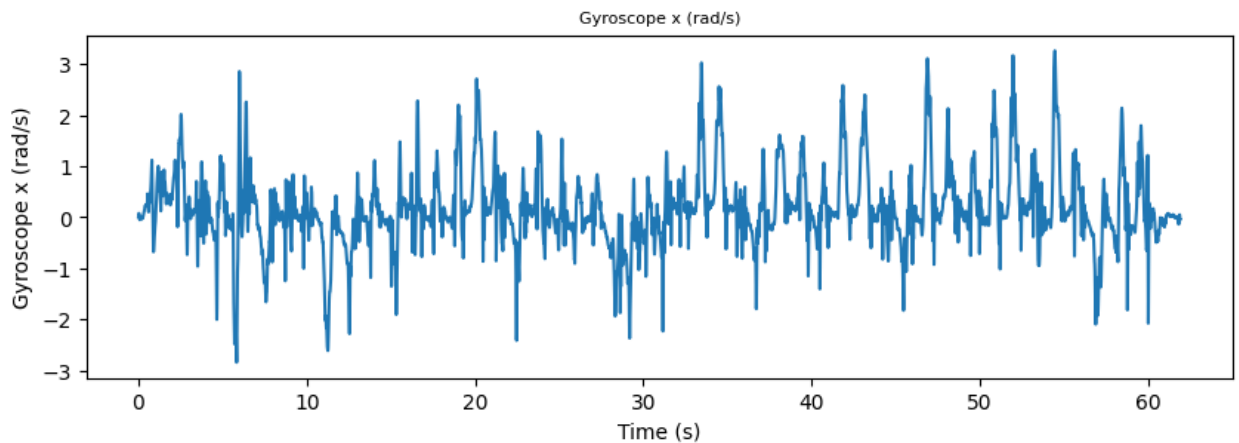
Data acceleration diimport menjadi dataframe bernama accelData dengan menggunakan library pandas.

```
# Plotting axis data in accelerometer
for col in accelData:
    if col != "Time (s)":
        plt.figure(figsize=(10,3))
        sns.lineplot(data = accelData, x = "Time (s)", y = col)
        plt.title(col, fontsize= 8)

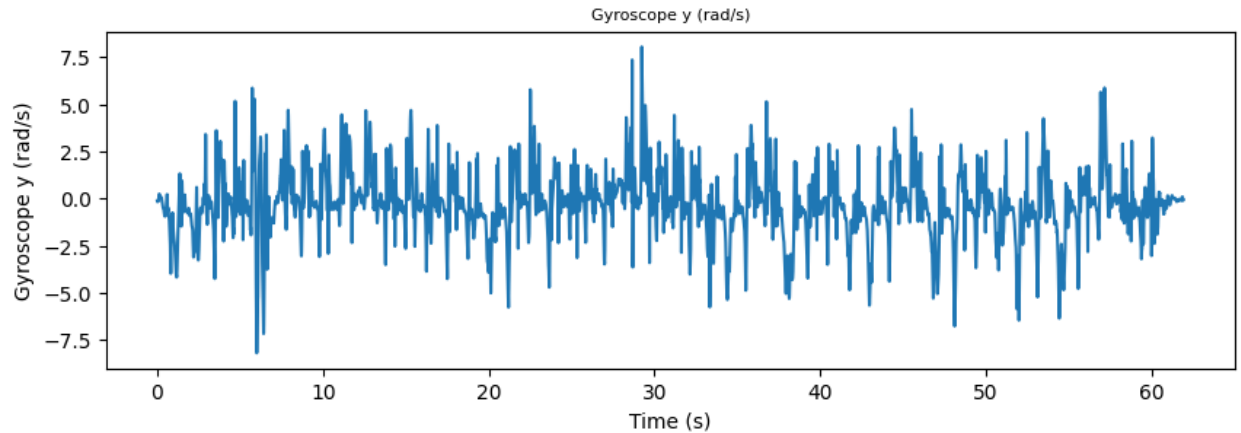
plt.show()
```

Membuat plot dari data masing-masing sumbu accelerometer

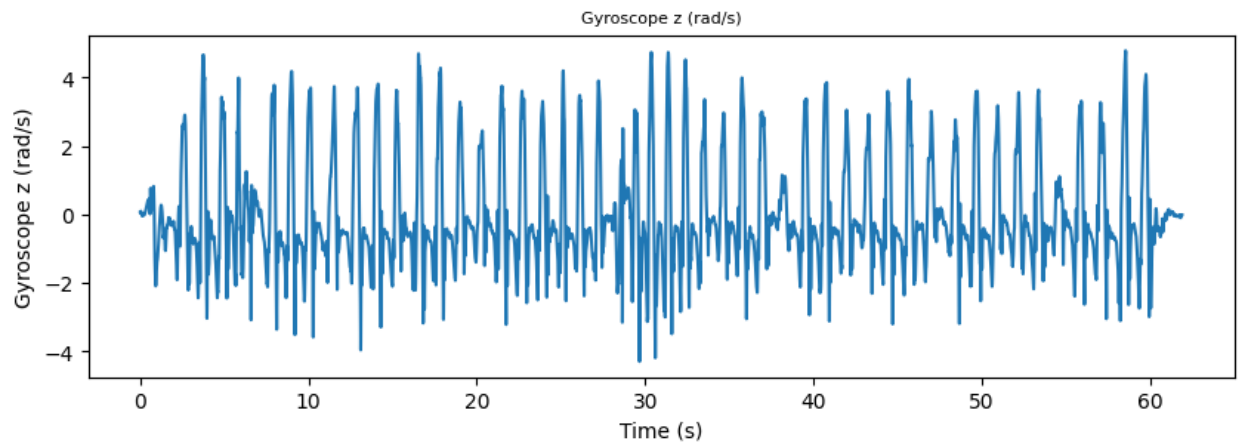
- Plots (Gyroscope)



Seperti yang terlihat pada gambar, data sumbu x pada gyroscope memiliki range antara $\pm 3 - \pm 3$. Dengan nilai maksimum dari data tersebut adalah $3,25\text{ rad/s}$.



Seperti yang terlihat pada gambar, data sumbu y pada gyroscope memiliki range antara ± 8 - ± 8 . Dengan nilai maksimum dari data tersebut adalah 8,03 rad/s.



Seperti yang terlihat pada gambar, data sumbu x pada gyroscope memiliki range antara ± 5 - ± 5 . Dengan nilai maksimum dari data tersebut adalah 4,79 rad/s.

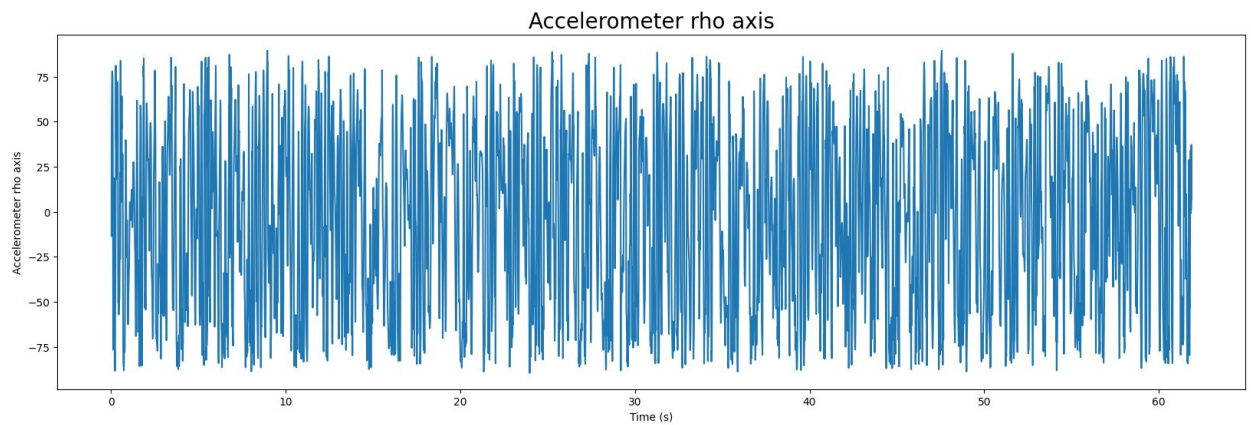
- Code

```
# Importing gyroscope data
gyroData = pd.read_excel(r'/content/Gyroscope rotation rate 2023-04-15 13-19-00.xls')
print(gyroData)
```

```
# Plotting axis data in gyroscope
for col in gyroData:
    if col != "Time (s)":
        plt.figure(figsize=(10,3))
        sns.lineplot(data = gyroData, x = "Time (s)", y = col)
        plt.title(col, fontsize= 8)

plt.show()
```

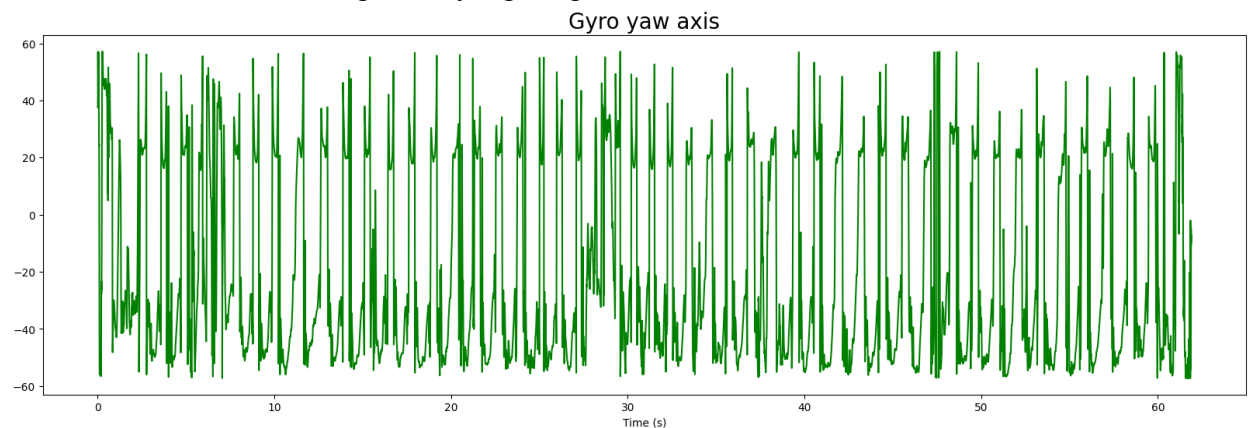
4.



a.

Perhitungan sudut ρ dengan menggunakan rumus $\rho = \arctan\left(\frac{A_x}{\sqrt{A_y^2 + A_z^2}}\right)$ menghasilkan

data seperti yang ditampilkan pada plot. Rata-rata sudut ρ accelerometer adalah $-4,108^\circ$, sedangkan nilai terbesarnya adalah $89,479^\circ$. Dapat disimpulkan bahwa data sudut ρ pada accelerometer memiliki range data yang sangat besar.




b.

Sudut yaw giro yang memiliki rata-rata nilai -18,59 rad/s dan nilai maksimum 57,26 rad/s

dihitung dengan menggunakan rumus $yaw = 180 \times \arctan\left(\frac{z_{gyro_i}}{\sqrt{x_{gyro_i}^2 + y_{gyro_i}^2}}\right)$. Jika

diperhatikan, sudut p accelerometer memiliki range yang tidak terlalu jauh berbeda dengan sudut yaw gyro, yaitu terpaut 14,482.


5.

 # Adding accelerometer degrees data into gyro dataframe

```
for i in range(len(gyroData)):
    gyroData['accel_Xangle'] = accelData['Accelerometer Xangle']
    gyroData['accel_Yangle'] = accelData['Accelerometer Yangle']
    gyroData['accel_Zangle'] = accelData['Accelerometer Zangle']
```

a.

Karena adanya perbedaan jumlah data yang besar, maka digunakan looping untuk memastikan bahwa panjang 3 kolom baru tersebut sama dengan panjang kolom lainnya pada dataframe gyroData.

 # replacing nan value with mean

```
# adanya perbedaan jumlah data pada acceleration dan gyroscope menyebabkan
# adanya banyak data yang bernilai nan, maka data nan tersebut diganti dengan 0

gyroData['accel_Xangle'] = gyroData['accel_Xangle'].fillna(accelData['Accelerometer Xangle'].mean())
gyroData['accel_Yangle'] = gyroData['accel_Yangle'].fillna(accelData['Accelerometer Yangle'].mean())
gyroData['accel_Zangle'] = gyroData['accel_Zangle'].fillna(accelData['Accelerometer Zangle'].mean())
```

Karena panjang kolom dipaksa untuk sama, menyebabkan banyak kolom bernilai nan, maka semua nilai nan diganti dengan nilai rata-rata tiap kolom sudut (x, y, dan z).

```
[38] # converting gyro rad/s into degrees
gyroData['Xangle'] = gyroData['accel_Xangle'] * 0.98 + gyroData['Gyroscope x (rad/s)'] * 0.02
gyroData['Yangle'] = gyroData['accel_Yangle'] * 0.98 + gyroData['Gyroscope y (rad/s)'] * 0.02
gyroData['Zangle'] = gyroData['accel_Zangle'] * 0.98 + gyroData['Gyroscope z (rad/s)'] * 0.02
```

Konversi rad/s ke degrees dilakukan dengan menggunakan rumus:

$$angle_i = angle_i \times 0,98 + rad_i \times 0.02$$

```

# Complementary Filter Algorithm

compFilterX = np.zeros(len(gyroData))
compFilterY = np.zeros(len(gyroData))
compFilterZ = np.zeros(len(gyroData))

for i in range(len(gyroData)):
    compFilterX[i] = 0.98 * (compFilterX[i-1] + gyroData['Xangle'][i] * 0.02) + 0.2 * gyroData['accel_Xangle'][i]
    compFilterY[i] = 0.98 * (compFilterY[i-1] + gyroData['Yangle'][i] * 0.02) + 0.2 * gyroData['accel_Yangle'][i]
    compFilterZ[i] = 0.98 * (compFilterZ[i-1] + gyroData['Zangle'][i] * 0.02) + 0.2 * gyroData['accel_Zangle'][i]

gyroData['compFilterX'] = compFilterX
gyroData['compFilterY'] = compFilterY
gyroData['compFilterZ'] = compFilterZ

```

b.

Dengan mengikuti rumus perhitungan complementary filter pada halaman 26 slide 6, didapatkan nilai complementary filter untuk setiap sudut x, y, dan z.

```

# Calculating Gait Angle
gaitData = pd.DataFrame()
gaitData['Time (s)'] = gyroData['Time (s)']
gaitData['x angle'] = (gyroData['compFilterX'].mean() * 0.8) + (accelData['Accelerometer Xangle'] * np.pi * 0.2)
gaitData['y angle'] = (gyroData['compFilterY'].mean() * 0.8) + (accelData['Accelerometer Yangle'] * np.pi * 0.2)
gaitData['z angle'] = (gyroData['compFilterZ'].mean() * 0.8) + (accelData['Accelerometer Zangle'] * np.pi * 0.2)

gaitData

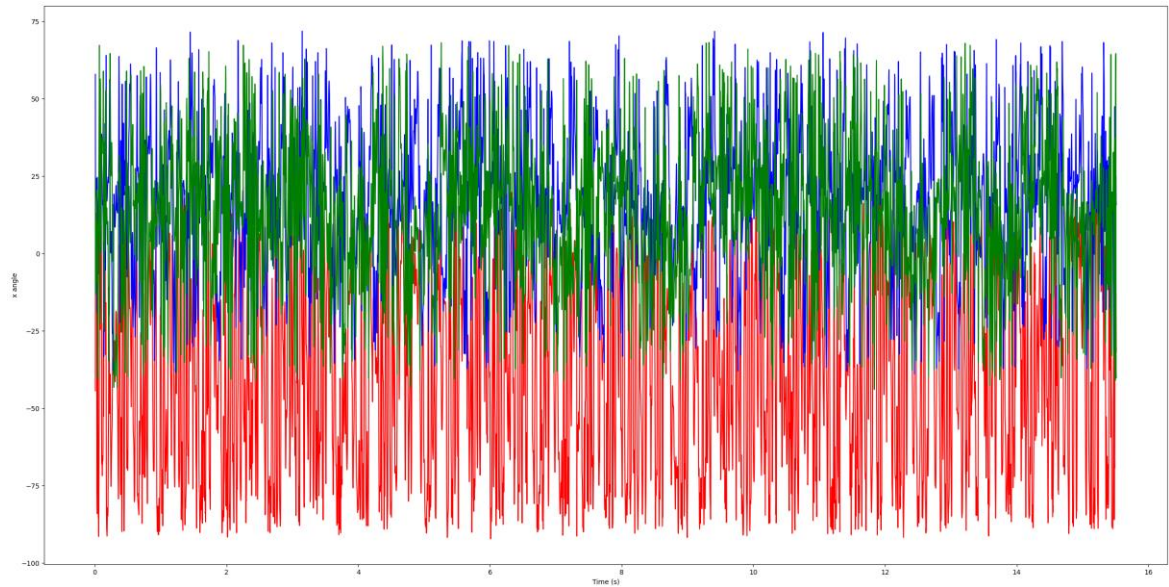
```

c.

	Time (s)	x angle	y angle	z angle
0	0.007693	-44.408868	57.908999	24.221892
1	0.009707	-5.373757	21.322705	-13.166648
2	0.011721	-1.652491	20.712855	-9.575441
3	0.013766	4.800926	18.430146	-3.527333
4	0.015750	10.909456	20.628399	3.469933
...
30700	61.886980	NaN	NaN	NaN
30701	61.888994	NaN	NaN	NaN
30702	61.891009	NaN	NaN	NaN
30703	61.893084	NaN	NaN	NaN
30704	61.895037	NaN	NaN	NaN

30705 rows x 4 columns

Data gait didapatkan dengan menggunakan perhitungan seperti yang tertera pada gambar. Karena adanya perbedaan jumlah data pada gyroscope dan acceleration, maka akan menghasilkan banyak data sudut x, y, z pada dataframe accelData yang bernilai nan.



Link:

Google colab:

https://colab.research.google.com/drive/1Tdpmv7alZIf4hjT2LmTc_6WIDzBu6-Wl?usp=sharing

github: <https://github.com/Indira-Chrsn/UTS-IoT>