sensor_fusion

March 18, 2021

[7]:
```
import math
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

GRAVITY = 9.80665
RAD_TO_DEG = 180 / math.pi
```

[8]:
```
# read test params from CSVP
csvp = open("euler_angles.csvp")

# create params array
params = []
for line in csvp: params.append(eval(line))
params = np.array(params)

print(params)
```

```
[   1 3813    0  300    0    1    0  960   96    8 2000   92   92   63
   63   14   14   86   86    0    0    0    0    0    0    0    0    0
    0    0    0]
```

[9]:
```
# read data from CSV
data = pd.read_csv("euler_angles.csv", names=["AccelX", "AccelY", "AccelZ",␣
 ↪"GyroX", "GyroY", "GyroZ", "MagX", "MagY", "MagZ"], index_col=False)

sample_rate = params[7]

# add time axis to data set
time = np.arange(0, len(data)/sample_rate, 1/sample_rate)
data.insert(0, "Time", time)

# sign data
data = data.applymap(lambda x: x-65535 if x > 32767 else x)

# apply accel sensitivity
acc_cols = ["AccelX", "AccelY", "AccelZ"]
acc_sens = params[9]
```

```
data[acc_cols] = data[acc_cols].applymap(lambda x: x * acc_sens * GRAVITY /␣
↪32768)

# apply gyro sensitivity
gyro_cols = ["GyroX", "GyroY", "GyroZ"]
gyro_sens = params[10]
data[gyro_cols] = data[gyro_cols].applymap(lambda x: x * gyro_sens / 32768)

# apply mag sensitivity
mag_cols = ["MagX", "MagY", "MagZ"]
mag_sens = 4800
data[mag_cols] = data[mag_cols].applymap(lambda x: x * mag_sens / 8192)

# use this instead of data
mag_data = data[mag_cols + ["Time"]].dropna()

print(data.head(20))
```

```
        Time    AccelX    AccelY     AccelZ     GyroX      GyroY      GyroZ  \
0   0.000000 -0.299275 -0.936133 -10.685322 -2.136230  -1.464844   2.685547
1   0.001042  1.431733 -0.639252 -10.120290  5.615234  -3.173828  24.597168
2   0.002083  0.555455  0.957681 -10.168174  2.807617 -12.451172  14.160156
3   0.003125  0.052672 -1.094150  -9.512163 -0.122070 -17.639160  -2.014160
4   0.004167  0.167594 -0.179565  -9.890447 -5.004883   0.000000  13.854980
5   0.005208  0.435745 -0.869095 -10.915165 -0.366211  -5.920410  20.141602
6   0.006250  0.699107 -0.940921  -9.454702 -1.831055  -5.798340  32.104492
7   0.007292  1.115698  0.359130 -11.657368  1.892090 -11.657715  15.930176
8   0.008333  1.288080 -1.127669  -9.742007 -0.549316  -3.906250   6.286621
9   0.009375  0.541090  1.278504  -9.512163 -2.807617 -12.573242   1.586914
10  0.010417  0.708684 -0.711078  -9.028534  0.244141 -13.549805  27.893066
11  0.011458  0.924162 -0.849942  -9.818621 -4.394531  -5.126953   6.103516
12  0.012500  0.967257  0.569820 -10.053253 -1.770020  -8.972168  18.188477
13  0.013542  0.885855  0.316035  -9.220071 -9.399414   0.854492   3.662109
14  0.014583  0.612916  0.201113  -9.449914 -1.403809 -21.545410  20.935059
15  0.015625  1.010353 -0.284910  -9.224859 -2.136230  -2.563477  28.564453
16  0.016667  0.397437 -1.319205  -9.052476 -3.295898  -5.371094  12.023926
17  0.017708  0.086191 -0.505177 -10.316615  9.643555   1.708984  11.047363
18  0.018750  0.119710  0.459687  -9.325415 -3.112793  -7.141113  11.657715
19  0.019792  0.814029  0.612916  -9.943119  0.732422  -2.014160  10.925293

          MagX        MagY        MagZ
0  -281.835938   71.484375 -118.945312
1          NaN         NaN         NaN
2          NaN         NaN         NaN
3          NaN         NaN         NaN
4          NaN         NaN         NaN
5          NaN         NaN         NaN
6          NaN         NaN         NaN
```

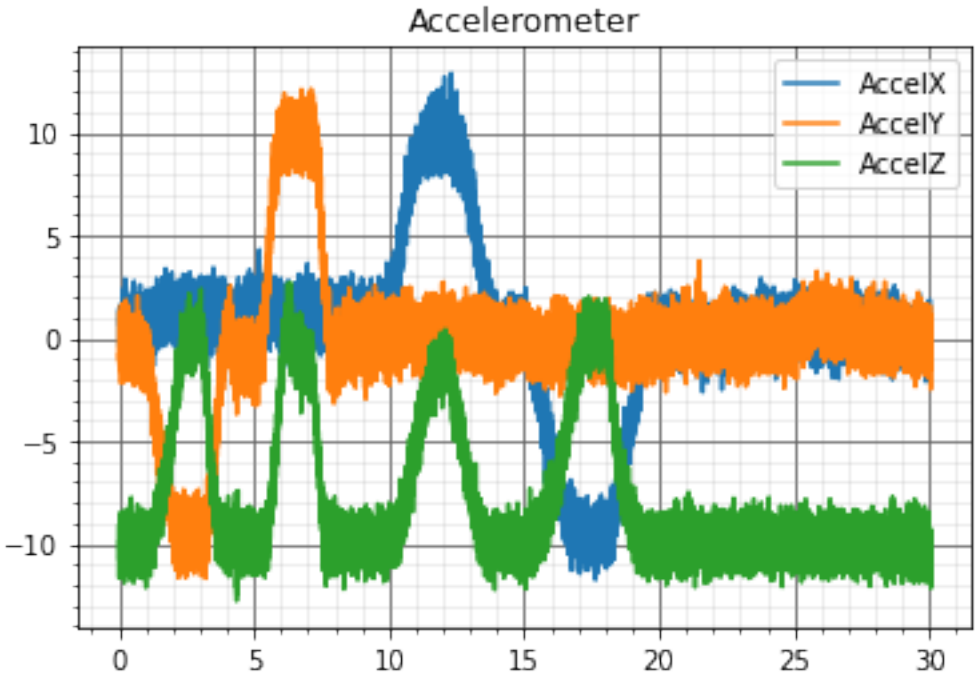
```
7           NaN         NaN         NaN
8           NaN         NaN         NaN
9           NaN         NaN         NaN
10 -343.359375  152.929688 -192.187500
11          NaN         NaN         NaN
12          NaN         NaN         NaN
13          NaN         NaN         NaN
14          NaN         NaN         NaN
15          NaN         NaN         NaN
16          NaN         NaN         NaN
17          NaN         NaN         NaN
18          NaN         NaN         NaN
19          NaN         NaN         NaN
```

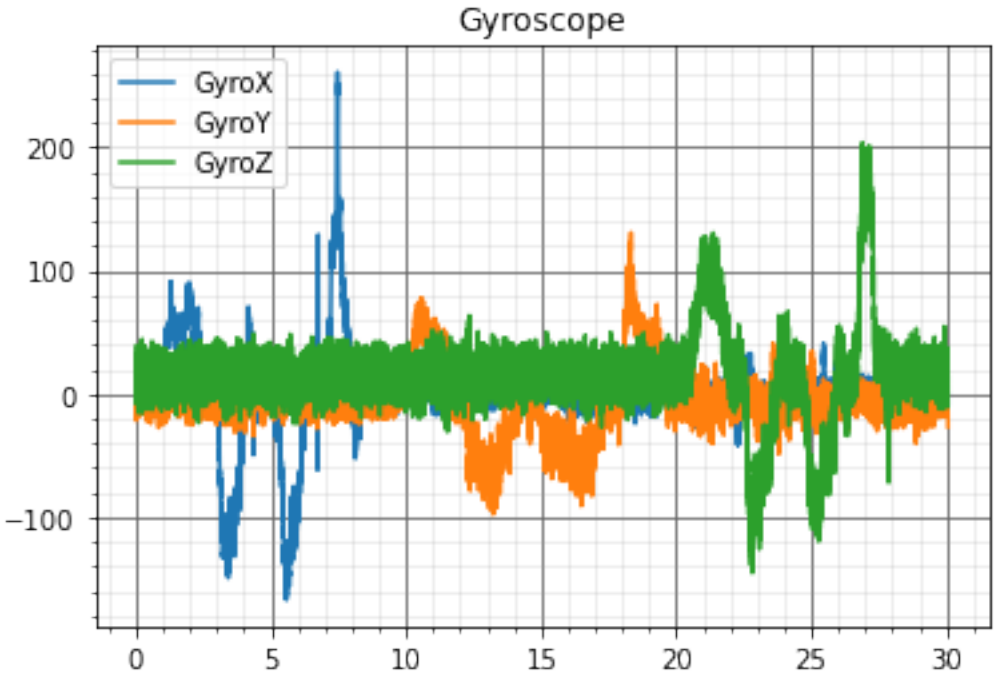
[10]:
```
plt.plot(data["Time"], data["AccelX"], label="AccelX")
plt.plot(data["Time"], data["AccelY"], label="AccelY")
plt.plot(data["Time"], data["AccelZ"], label="AccelZ")

# display the plot
plt.title("Accelerometer")
plt.grid(b=True, which='major', color='#666666', linestyle='-')
plt.minorticks_on()
plt.grid(b=True, which='minor', color='#999999', linestyle='-', alpha=0.2)
plt.legend()
plt.show()

plt.plot(data["Time"], data["GyroX"], label="GyroX")
plt.plot(data["Time"], data["GyroY"], label="GyroY")
plt.plot(data["Time"], data["GyroZ"], label="GyroZ")

# display the plot
plt.title("Gyroscope")
plt.grid(b=True, which='major', color='#666666', linestyle='-')
plt.minorticks_on()
plt.grid(b=True, which='minor', color='#999999', linestyle='-', alpha=0.2)
plt.legend()
plt.show()
```


Accelerometer
AccelX
AccelY
AccelZ
10
5
0
−5
−10
0
5
10
15
20
25
30

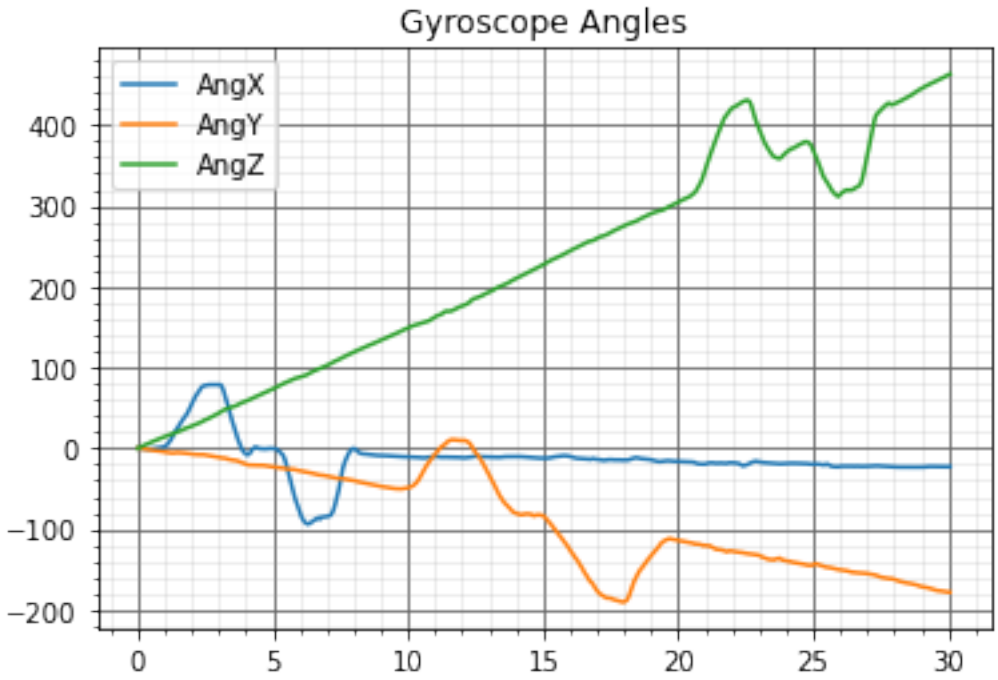

Gyroscope
GyroX
GyroY
GyroZ
200
100
0
−100
0
5
10
15
20
25
30

[11]:
```
from scipy import integrate

# calculate angles from gyroscope
ang_x = integrate.cumtrapz(y=data["GyroX"], x=data["Time"], initial=0)
ang_y = integrate.cumtrapz(y=data["GyroY"], x=data["Time"], initial=0)
ang_z = integrate.cumtrapz(y=data["GyroZ"], x=data["Time"], initial=0)

plt.plot(data["Time"], ang_x, label="AngX")
plt.plot(data["Time"], ang_y, label="AngY")
plt.plot(data["Time"], ang_z, label="AngZ")

# display the plot
plt.title("Gyroscope Angles")
plt.grid(b=True, which='major', color='#666666', linestyle='-')
plt.minorticks_on()
plt.grid(b=True, which='minor', color='#999999', linestyle='-', alpha=0.2)
plt.legend()
plt.show()
```



[12]:
```
# calculate angles from accelerometer
acc_ang_x = np.arctan2(-data["AccelY"], -data["AccelZ"]) * RAD_TO_DEG
acc_ang_y = np.arctan2(-data["AccelX"], np.sqrt(data["AccelY"]**2 +␣
 ↪data["AccelZ"]**2)) * RAD_TO_DEG
```

```
plt.plot(data["Time"], acc_ang_x, label="AngX")
plt.plot(data["Time"], acc_ang_y, label="AngY")

# display the plot
plt.title("Accelerometer Angles")
plt.grid(b=True, which='major', color='#666666', linestyle='-')
plt.minorticks_on()
plt.grid(b=True, which='minor', color='#999999', linestyle='-', alpha=0.2)
plt.legend()
plt.show()
```