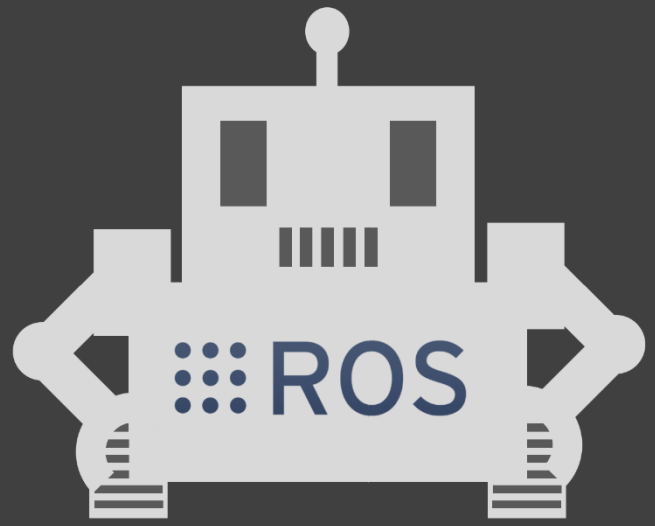
라즈베리 파이 기초

Chapter 3. IMU 센서

구선생 로보틱스



강의 자료 다운로드

https://github.com/DoveSensei/Note

라즈베리 파이에 원격접속 하기

원격 접속 명령어

```
ssh msrose@192.168.10.10
```

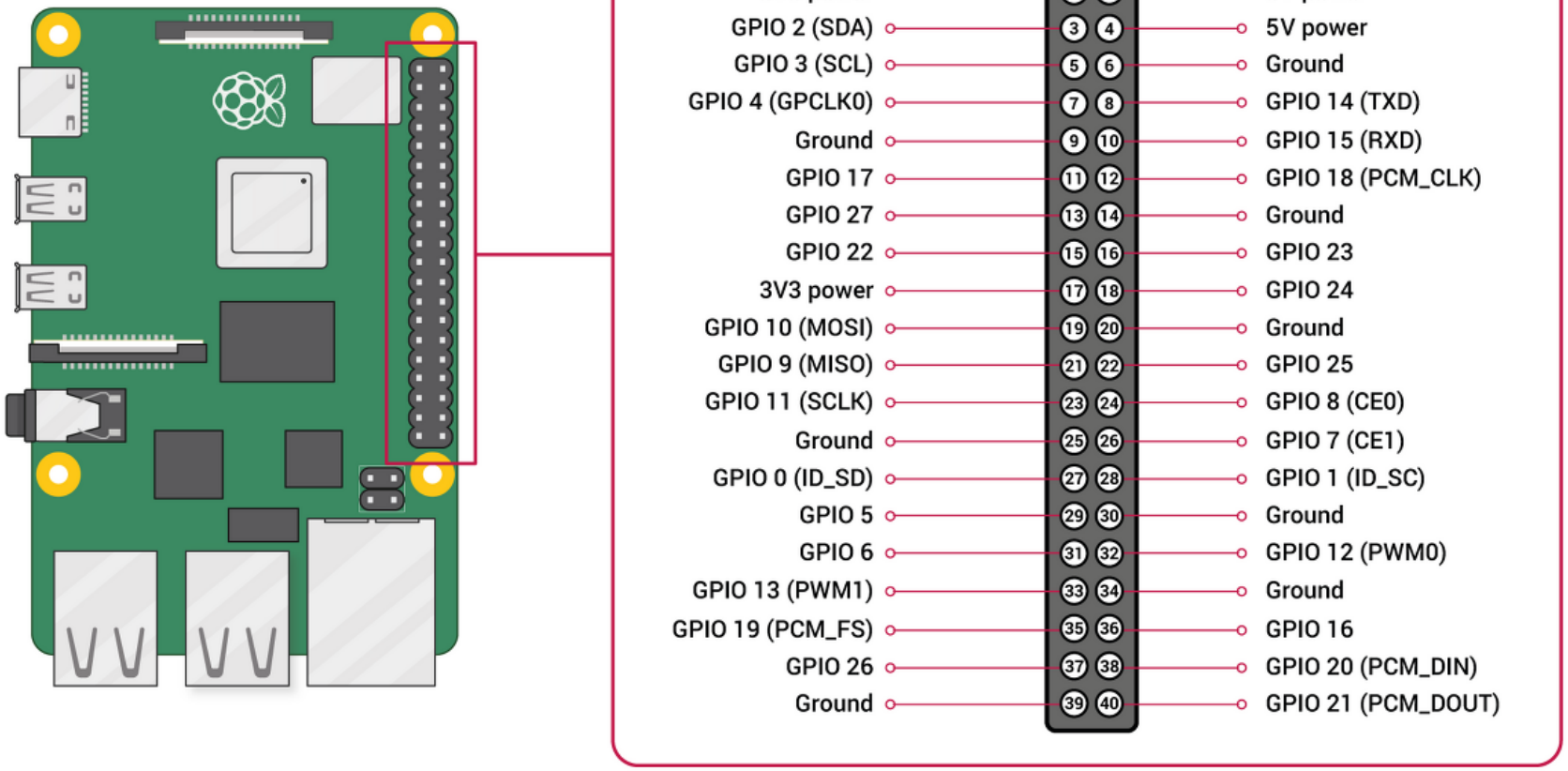
```
Linux raspberrypi 6.1.21-v8+ #1642 SMP PREEMPT Mon Apr  3 17:24:16 BST 2023 aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri May 12 04:03:16 2023 from 192.168.170.89
msrose@raspberrypi:~ $
```

원격 접속 후 터미널의 사용자가 msrose로 바뀌었다

라즈베리 파이 Pin Map



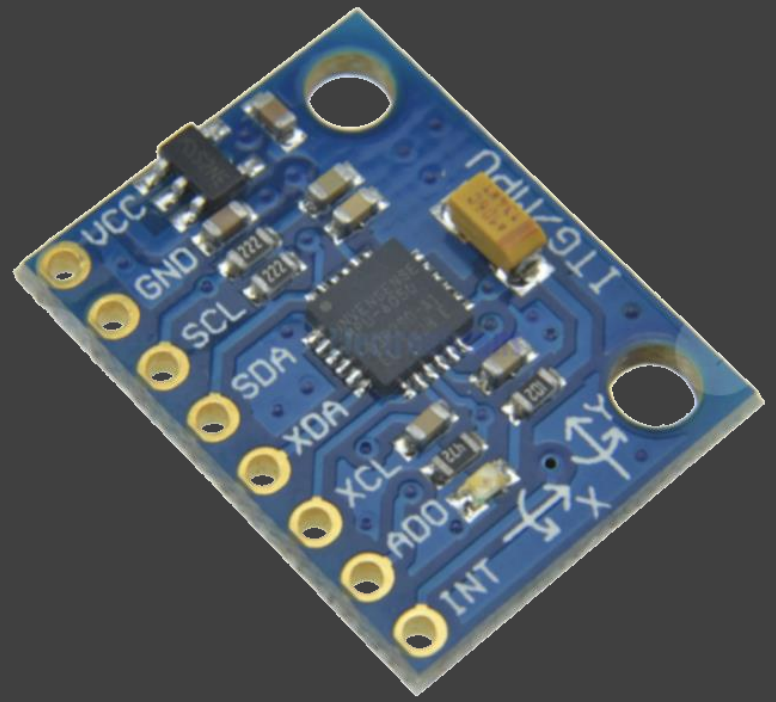
IMU 센서

IMU 센서

IMU 센서란 무엇인가?

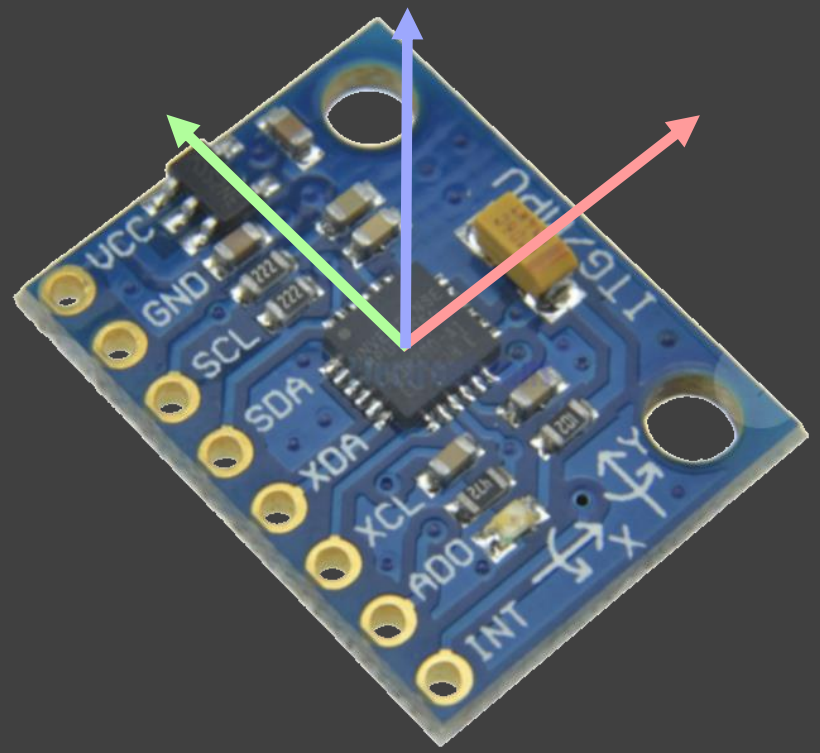
가속도 센서, 자이로스코프, 그리고 때때로는 자기장 센서를 포함하는 센서.

물체의 방향, 속도, 그리고 위치를 추정하기 위해 사용한다.



IMU 센서

MPU 6050



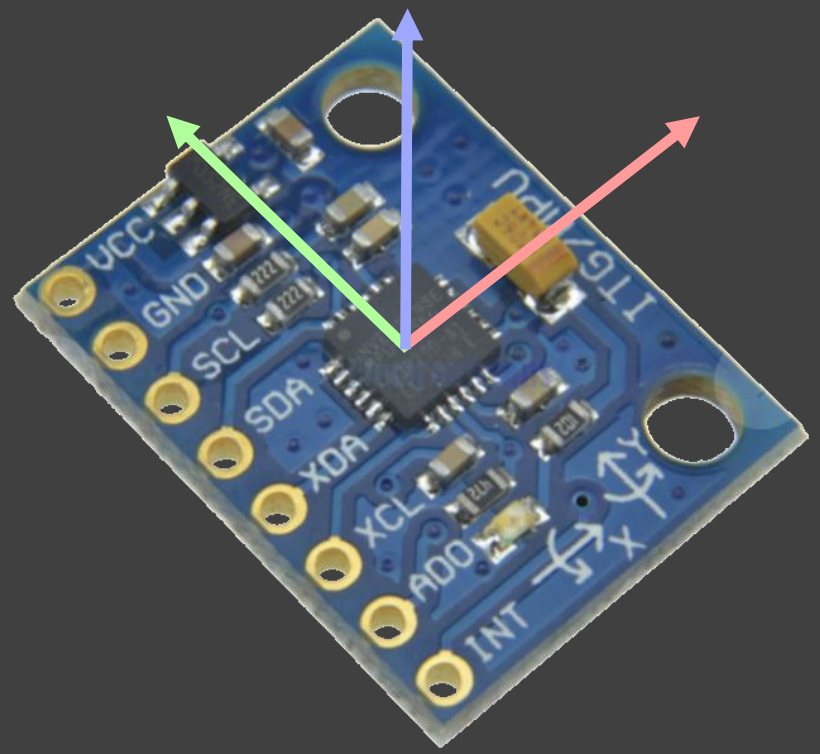
가속도 센서: 가속도를 측정하는 센서

자이로스코프 : 각속도를 측정하는 센서

I2C 통신 사용

IMU 센서

센서의 물리량



가속도 : 속도의 변화량 $[m/s^2]$

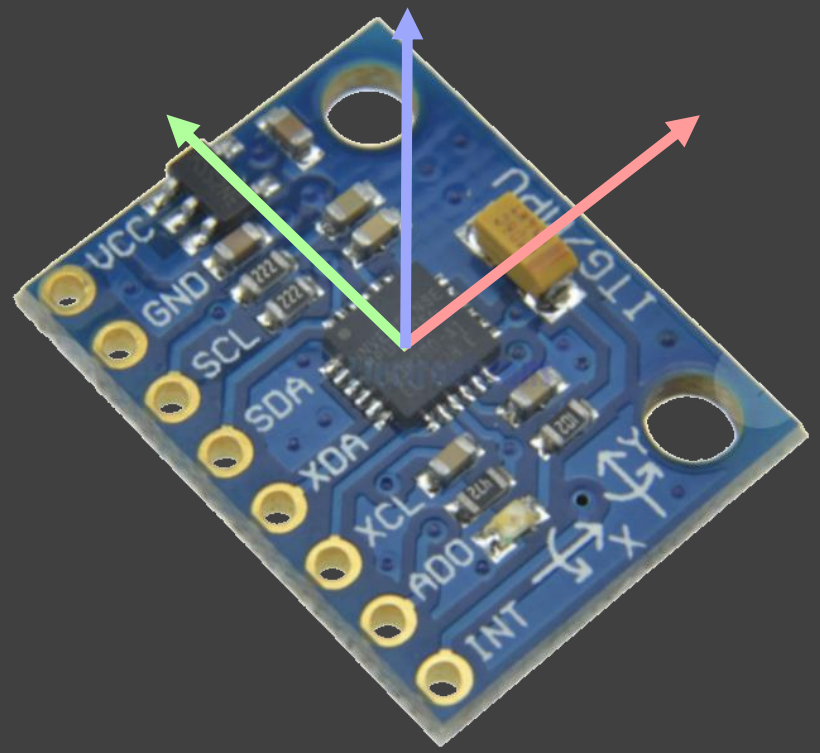
MPU6050는 g 로 출력

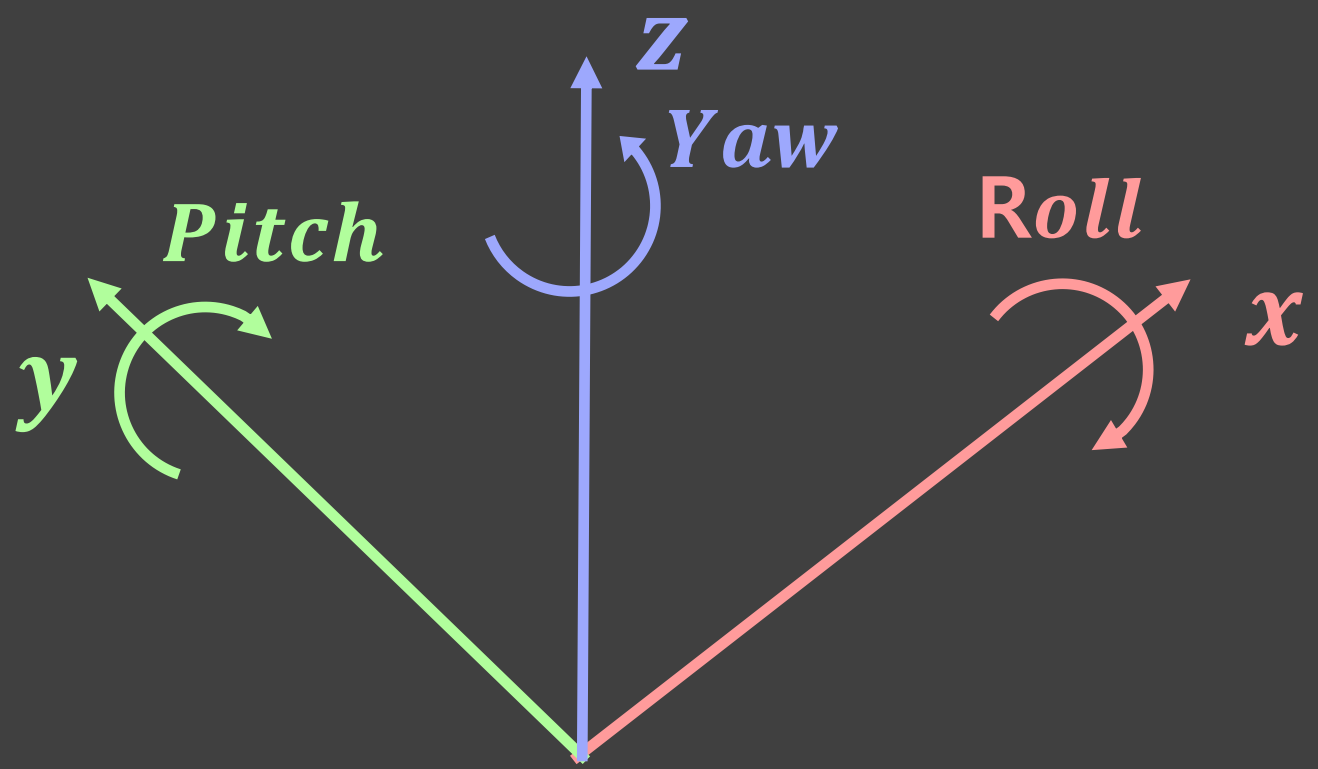
각속도 : 각의 변화량 $[\mathrm{rad}/s]$

MPU6050는 $^\circ/s$로 출력

IMU 센서

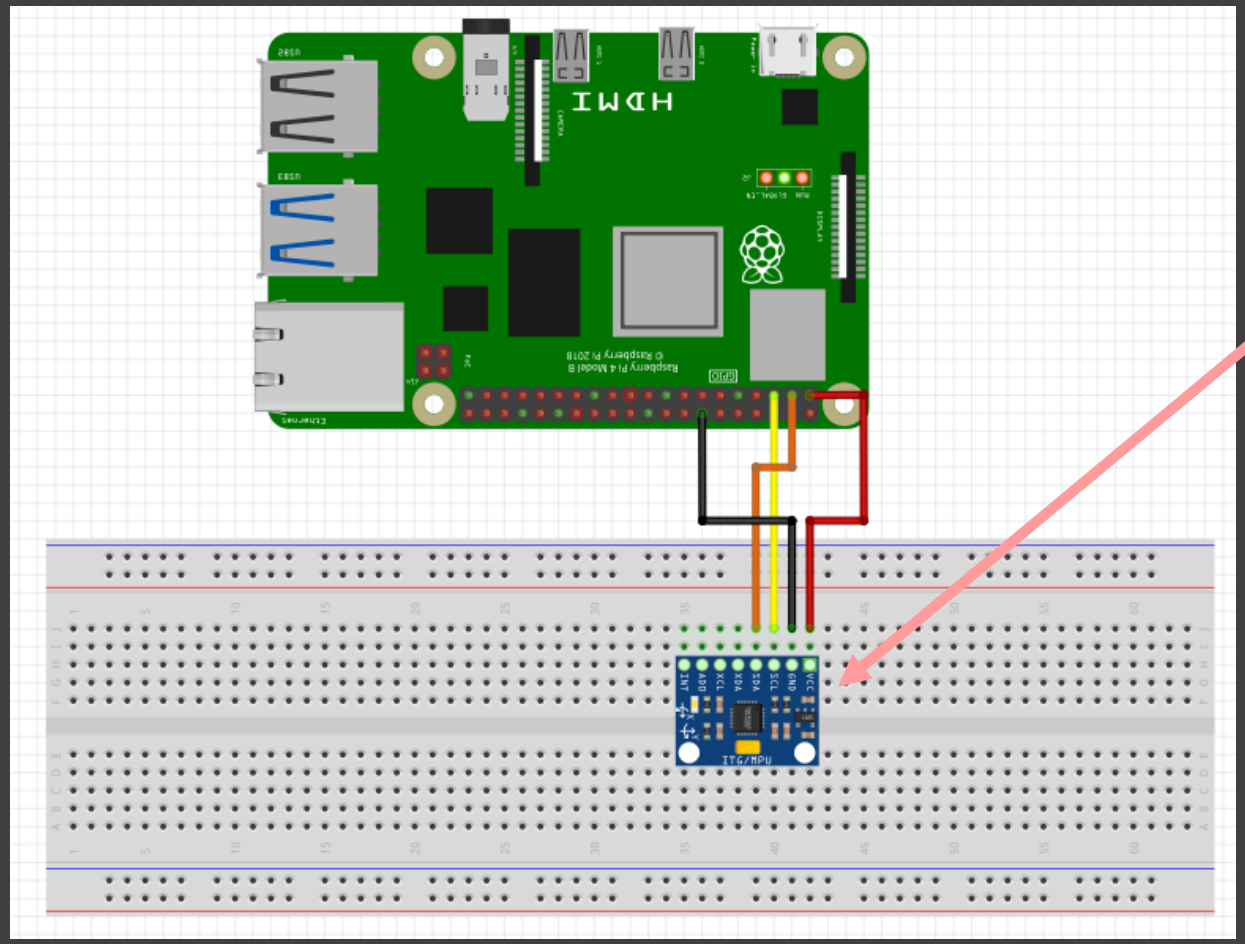
회전의 표현

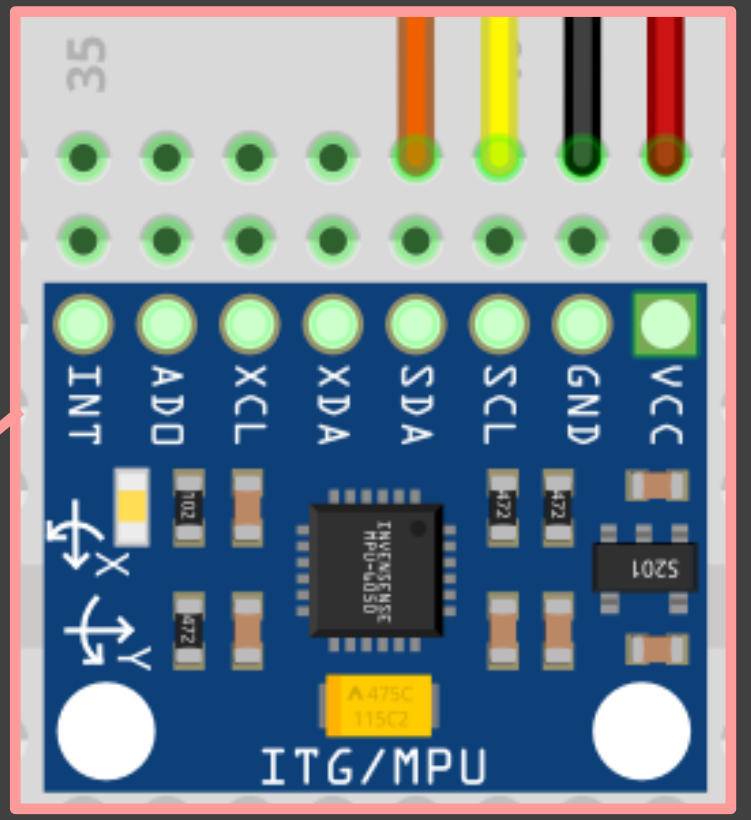




IMU 센서

MPU6050 센서 연결





핀 이름 확인 후 연결

IMU 센서

MPU6050 기본 소스코드 작성

```
import smbus
import time

PWR_MGMT_1     = 0x6B
SMPLRT_DIV     = 0x19
CONFIG         = 0x1A
GYRO_CONFIG    = 0x1B
INT_ENABLE     = 0x38
ACCEL_XOUT_H   = 0x3B
ACCEL_YOUT_H   = 0x3D
ACCEL_ZOUT_H   = 0x3F
GYRO_XOUT_H    = 0x43
GYRO_YOUT_H    = 0x45
GYRO_ZOUT_H    = 0x47

bus = smbus.SMBus(1)
Device_Address = 0x68

def MPU_Init():
        bus.write_byte_data(Device_Address, SMPLRT_DIV, 7)
        bus.write_byte_data(Device_Address, PWR_MGMT_1, 1)
        bus.write_byte_data(Device_Address, CONFIG, 0)
        bus.write_byte_data(Device_Address, GYRO_CONFIG, 24)
        bus.write_byte_data(Device_Address, INT_ENABLE, 1)

def read_raw_data(addr):
        high = bus.read_byte_data(Device_Address, addr)
        low = bus.read_byte_data(Device_Address, addr+1)
        value = ((high << 8) | low)

        if(value > 32768):
                value = value - 65536

        return value

```

IMU 센서

MPU6050 오리지날 센서 값 읽기 소스코드 작성

```
36  try:
37      MPU_Init()  #  자이로센서  초기화
38      while True:
39          acc_x = read_raw_data(ACCEL_XOUT_H)  #  가속도  X  센서  값  읽기
40          acc_y = read_raw_data(ACCEL_YOUT_H)  #  가속도  Y  센서  값  읽기
41          acc_z = read_raw_data(ACCEL_ZOUT_H)  #  가속도  Z  센서  값  읽기
42
43          gyro_x = read_raw_data(GYRO_XOUT_H)  #  각속도  X  센서  값  읽기
44          gyro_y = read_raw_data(GYRO_YOUT_H)  #  각속도  Y  센서  값  읽기
45          gyro_z = read_raw_data(GYRO_ZOUT_H)  #  각속도  Z  센서  값  읽기
46
47          print('acc_x : ', acc_x)
48          print('acc_y : ', acc_y)
49          print('acc_z : ', acc_z)
50          print('gyro_x : ', gyro_x)
51          print('gyro_y : ', gyro_y)
52          print('gyro_z : ', gyro_z)
53          print('')
54          time.sleep(0.1)
55
56  except KeyboardInterrupt:
57      pass
58
```

IMU 센서

MPU6050 센서 값 물리적 값 변환

```
acc_x = read_raw_data(ACCEL_XOUT_H)
acc_y = read_raw_data(ACCEL_YOUT_H)
acc_z = read_raw_data(ACCEL_ZOUT_H)
```

AFS_SEL	Full Scale Range	LSB Sensitivity
0	±2*g*	16384 LSB/*g*
1	±4*g*	8192 LSB/*g*
2	±8*g*	4096 LSB/*g*
3	±16*g*	2048 LSB/*g*

센서값에

16384를 나누면 물리적 값으로 변환

```
gyro_x = read_raw_data(GYRO_XOUT_H)
gyro_y = read_raw_data(GYRO_YOUT_H)
gyro_z = read_raw_data(GYRO_ZOUT_H)
```

FS_SEL	Full Scale Range	LSB Sensitivity
0	± 250 °/s	131 LSB/°/s
1	± 500 °/s	65.5 LSB/°/s
2	± 1000 °/s	32.8 LSB/°/s
3	± 2000 °/s	16.4 LSB/°/s

센서값에

131을 나누면 물리적 값으로 변환

IMU 센서

MPU6050 센서 값 물리적 값 변환 소스코드 작성

```
36  try:
37      MPU_Init()  #  자이로센서  초기화
38      while True:
39          acc_x = read_raw_data(ACCEL_XOUT_H)  #  가속도  X  센서  값  읽기
40          acc_y = read_raw_data(ACCEL_YOUT_H)  #  가속도  Y  센서  값  읽기
41          acc_z = read_raw_data(ACCEL_ZOUT_H)  #  가속도  Z  센서  값  읽기
42
43          gyro_x = read_raw_data(GYRO_XOUT_H)  #  각속도  X  센서  값  읽기
44          gyro_y = read_raw_data(GYRO_YOUT_H)  #  각속도  Y  센서  값  읽기
45          gyro_z = read_raw_data(GYRO_ZOUT_H)  #  각속도  Z  센서  값  읽기
46
47          Ax = (acc_x/16384.0) * 9.80665 #  가속도  X  센서  값  물리적인  값으로  변환
48          Ay = (acc_y/16384.0) * 9.80665 #  가속도  Y  센서  값  물리적인  값으로  변환
49          Az = (acc_z/16384.0) * 9.80665 #  가속도  Z  센서  값  물리적인  값으로  변환
50
51          Gx = (gyro_x/131.0)  #  각속도  X  센서  값  물리적인  값으로  변환
52          Gy = (gyro_y/131.0)  #  각속도  Y  센서  값  물리적인  값으로  변환
53          Gz = (gyro_z/131.0)  #  각속도  Z  센서  값  물리적인  값으로  변환
54
55          print('Ax  :  ', Ax)
56          print('Ay  :  ', Ay)
57          print('Az  :  ', Az)
58          print('Gx  :  ', Gx)
59          print('Gy  :  ', Gy)
60          print('Gz  :  ', Gz)
61          print('')
62          time.sleep(0.1)
63
64  except KeyboardInterrupt:
65      pass
```

IMU 센서

MPU6050 캘리브레이션

센서의 초기 오차를 보정하는 방법

$$sum = \sum_{i=0}^{n} \text{센서 값}$$

↓

$$avg = \frac{sum}{n}$$

↓

$$\text{보정된 센서 값} = \text{센서값} - avg$$

IMU 센서

MPU6050 캘리브레이션 소스코드 작성

```
36  try:
37      MPU_Init()
38
39      calibration_number = 100
40      avg_acc_x = 0
41      avg_acc_y = 0
42      avg_acc_z = 0
43      avg_gyro_x = 0
44      avg_gyro_y = 0
45      avg_gyro_z = 0
46
47      for i in range(calibration_number):
48          acc_x = read_raw_data(ACCEL_XOUT_H)  # 가속도 X 센서 값 읽기
49          acc_y = read_raw_data(ACCEL_YOUT_H)  # 가속도 Y 센서 값 읽기
50          acc_z = read_raw_data(ACCEL_ZOUT_H)  # 가속도 Z 센서 값 읽기
51          gyro_x = read_raw_data(GYRO_XOUT_H)  # 각속도 X 센서 값 읽기
52          gyro_y = read_raw_data(GYRO_YOUT_H)  # 각속도 Y 센서 값 읽기
53          gyro_z = read_raw_data(GYRO_ZOUT_H)  # 각속도 Z 센서 값 읽기
54
55          avg_acc_x = avg_acc_x + acc_x # 가속도 X 센서 값 누적
56          avg_acc_y = avg_acc_y + acc_y # 가속도 Y 센서 값 누적
57          avg_acc_z = avg_acc_z + acc_z # 가속도 Z 센서 값 누적
58          avg_acc_z = avg_acc_z - 16384.0 # 중력 값 보정
59          avg_gyro_x = avg_gyro_x + gyro_x # 각속도 X 센서 값 누적
60          avg_gyro_y = avg_gyro_y + gyro_y # 각속도 Y 센서 값 누적
61          avg_gyro_z = avg_gyro_z + gyro_z # 각속도 Z 센서 값 누적
62          time.sleep(0.1)
63          print('Calibrating ' , i ,'/', calibration_number)
64
65      avg_acc_x = avg_acc_x / calibration_number # 가속도 X 보정값 계산
66      avg_acc_y = avg_acc_y / calibration_number # 가속도 Y 보정값 계산
67      avg_acc_z = avg_acc_z / calibration_number # 가속도 Z 보정값 계산
68      avg_gyro_x = avg_gyro_x / calibration_number # 각속도 X 보정값 계산
69      avg_gyro_y = avg_gyro_y / calibration_number # 각속도 Y 보정값 계산
70      avg_gyro_z = avg_gyro_z / calibration_number # 각속도 Z 보정값 계산
```

IMU 센서

MPU6050 캘리브레이션 소스코드 작성

```
71
72      while True:
73          acc_x = read_raw_data(ACCEL_XOUT_H)  # 가속도 X 센서 값 읽기
74          acc_y = read_raw_data(ACCEL_YOUT_H)  # 가속도 Y 센서 값 읽기
75          acc_z = read_raw_data(ACCEL_ZOUT_H)  # 가속도 Z 센서 값 읽기
76
77          gyro_x = read_raw_data(GYRO_XOUT_H)  # 각속도 X 센서 값 읽기
78          gyro_y = read_raw_data(GYRO_YOUT_H)  # 각속도 Y 센서 값 읽기
79          gyro_z = read_raw_data(GYRO_ZOUT_H)  # 각속도 Z 센서 값 읽기
80
81          Ax = ((acc_x-avg_acc_x)/16384.0) * 9.80665 # 캘리브레이션 값 보정 후 가속도 X 센서 값 물리적인 값으로 변환
82          Ay = ((acc_y-avg_acc_y)/16384.0) * 9.80665 # 캘리브레이션 값 보정 후 가속도 Y 센서 값 물리적인 값으로 변환
83          Az = ((acc_z-avg_acc_z)/16384.0) * 9.80665 # 캘리브레이션 값 보정 후 가속도 Z 센서 값 물리적인 값으로 변환
84
85          Gx = ((gyro_x-avg_gyro_x)/131.0)  # 캘리브레이션 값 보정 후 각속도 X 센서 값 물리적인 값으로 변환
86          Gy = ((gyro_y-avg_gyro_y)/131.0)  # 캘리브레이션 값 보정 후 각속도 Y 센서 값 물리적인 값으로 변환
87          Gz = ((gyro_z-avg_gyro_z)/131.0)  # 캘리브레이션 값 보정 후각속도 Z 센서 값 물리적인 값으로 변환
88
89          print('Ax : ', Ax)
90          print('Ay : ', Ay)
91          print('Az : ', Az)
92          print('Gx : ', Gx)
93          print('Gy : ', Gy)
94          print('Gz : ', Gz)
95          print('')
96          time.sleep(0.1)
97
98  except KeyboardInterrupt:
99      pass
100
```

IMU 센서

파일 입출력 연습

```python
import time
import datetime
import os

# 파일 이름 설정
file_name = 'time_log.txt'

# 파일이 존재할 경우 삭제
if os.path.exists(file_name):
    os.remove(file_name)

try:
    while True:
        now_date_time = datetime.datetime.now() # 현재 시간 얻기
        now_linux_time = time.time() # 리눅스 현재 시간 얻기
        print('now_date_time : ', now_date_time)
        print('now_linux_time: ', now_linux_time)
        print('')
        f = open(file_name,'a') # 파일 오픈
        f.write(str(now_date_time) + ' ' + str(now_linux_time) + '\n') #파일 쓰기
        f.close() #파일 닫기
        time.sleep(1.0)

except KeyboardInterrupt:
    pass
```

IMU 센서

파일 입출력 연습 — 시간 설명

```
2023-06-02 23:54:14.494927 1685717654.4949615
2023-06-02 23:54:15.496887 1685717655.4969163
2023-06-02 23:54:16.498591 1685717656.4986212
```

한국 시간 설정 명령어

```
sudo timedatectl set-timezone Asia/Seoul
```

Data time – 년-월-일 시:분:초

Linux time – UNIX가 시작 된 이후 경과 된 시간.
소수점 시간으로 표현

IMU 센서

MPU6050 로그 파일 입출력

```
18    bus = smbus.SMBus(1)
19    Device_Address = 0x68
20
21    # 파일 이름 설정
22    file_name = 'IMU_log.txt'
23
24    # 파일이 존재할 경우 삭제
25    if os.path.exists(file_name):
26        os.remove(file_name)
27
28    def MPU_Init():
29        bus.write_byte_data(Device_Address, SMPLRT_DIV, 0x07)
30        bus.write_byte_data(Device_Address, PWR_MGMT_1, 0x04)
31        bus.write_byte_data(Device_Address, CONFIG, 0x00)
32        bus.write_byte_data(Device_Address, GYRO_CONFIG, 0x00)
33        bus.write_byte_data(Device_Address, INT_ENABLE, 0x01)
34
```

IMU 센서

MPU6050 로그 파일 입출력

```
81      while True:
82          now_date_time = datetime.datetime.now() # 현재 시간 얻기
83          now_linux_time = time.time() # 리눅스 현재 시간 얻기
84
85          acc_x = read_raw_data(ACCEL_XOUT_H) # 가속도 X 센서 값 읽기
86          acc_y = read_raw_data(ACCEL_YOUT_H) # 가속도 Y 센서 값 읽기
87          acc_z = read_raw_data(ACCEL_ZOUT_H) # 가속도 Z 센서 값 읽기
88
89          gyro_x = read_raw_data(GYRO_XOUT_H) # 각속도 X 센서 값 읽기
90          gyro_y = read_raw_data(GYRO_YOUT_H) # 각속도 Y 센서 값 읽기
91          gyro_z = read_raw_data(GYRO_ZOUT_H) # 각속도 Z 센서 값 읽기
92
93          Ax = ((acc_x-avg_acc_x)/16384.0) * 9.80665 # 캘리브레이션 값 보정 후 가속도 X 센서 값 물리적인 값으로 변환
94          Ay = ((acc_y-avg_acc_y)/16384.0) * 9.80665 # 캘리브레이션 값 보정 후 가속도 Y 센서 값 물리적인 값으로 변환
95          Az = ((acc_z-avg_acc_z)/16384.0) * 9.80665 # 캘리브레이션 값 보정 후 가속도 Z 센서 값 물리적인 값으로 변환
96
97          Gx = ((gyro_x-avg_gyro_x)/131.0) # 캘리브레이션 값 보정 후 각속도 X 센서 값 물리적인 값으로 변환
98          Gy = ((gyro_y-avg_gyro_y)/131.0) # 캘리브레이션 값 보정 후 각속도 Y 센서 값 물리적인 값으로 변환
99          Gz = ((gyro_z-avg_gyro_z)/131.0) # 캘리브레이션 값 보정 후각속도 Z 센서 값 물리적인 값으로 변환
100
101         print('Ax : ', Ax)
102         print('Ay : ', Ay)
103         print('Az : ', Az)
104         print('Gx : ', Gx)
105         print('Gy : ', Gy)
106         print('Gz : ', Gz)
107         print('')
108
109         f = open(file_name,'a') # 파일 오픈
110         f.write(str(now_date_time) + ' ' + str(now_linux_time) + ' ' + str(Ax)+ ' ' + str(Ay) + ' ' +  str(Az) + ' ' + str(Gx) + ' ' + str(Gy) + ' '
111         f.close() #파일 닫기
112         time.sleep(0.1)
                                                    str(Gy) + ' ' + str(Gz) + '\n') #파일 쓰기
113
114 except KeyboardInterrupt:
115     pass
```

IMU 센서

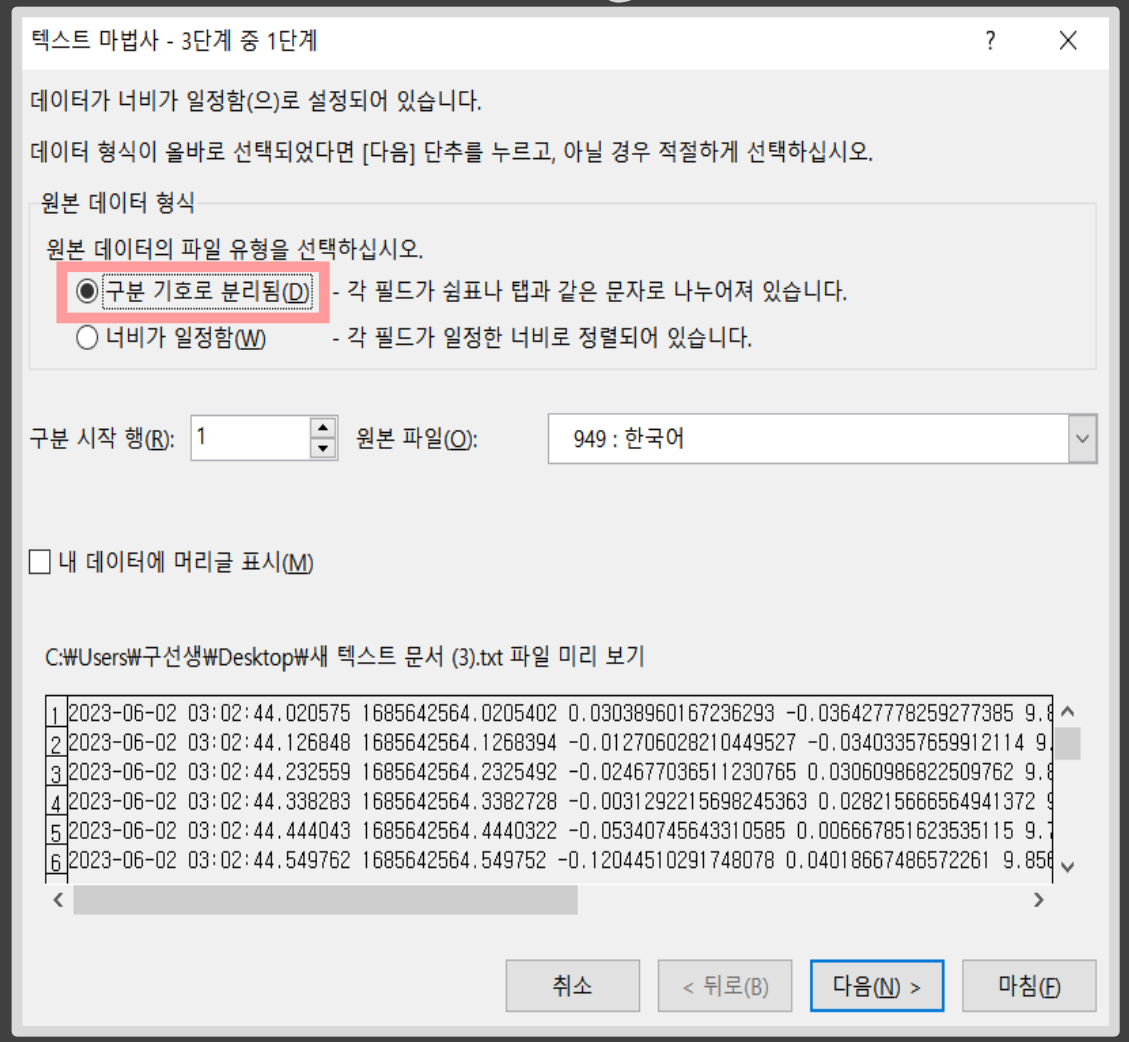
MPU6050 로그 파일 입출력

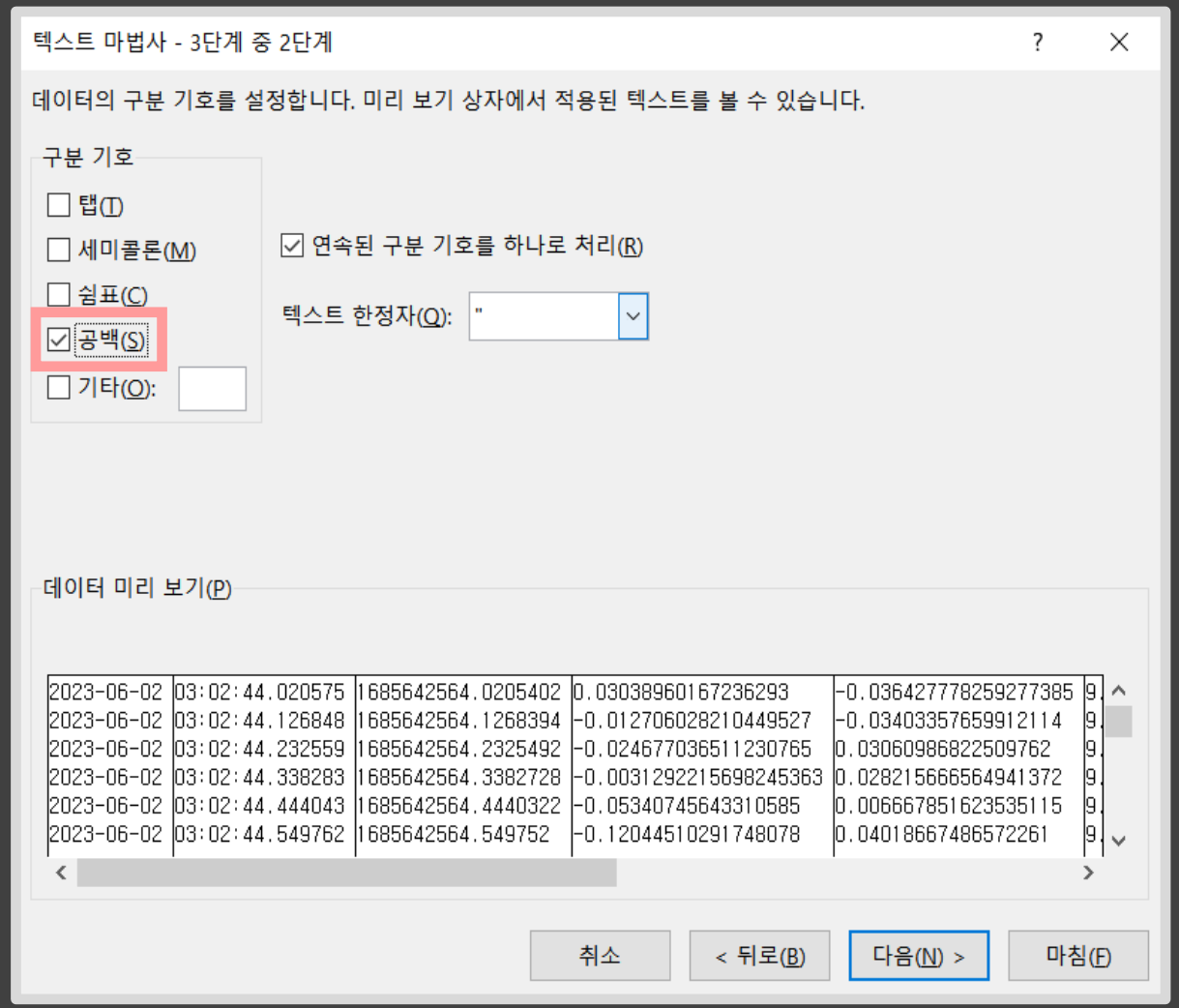
```
2023-06-02 23:59:18.211781 1685717958.2118258 -0.07745242370605467 0.015203180541992174 9.794966295898437 0.0012977099236640267 -0.045572519083969455 0.05404580152671754
2023-06-02 23:59:18.322349 1685717958.3223906 0.15478513732910157 -0.03746925598144532 9.761447472656249 -0.006335877862595515 -0.030305343511450374 -0.0986259541984733
2023-06-02 23:59:18.432771 1685717958.4328125 0.03507505432128907 -0.027892449340820323 9.806937304199218 -0.09030534351145048 0.007862595419847337 0.05404580152671754
2023-06-02 23:59:18.543955 1685717958.5439997 0.030286651000976574 -0.054228667602539075 9.644131591308593 -0.03687022900763368 -0.053206106870228996 0.03114503816793892
2023-06-02 23:59:18.655277 1685717958.6553218 0.030286651000976574 -0.06859387756347657 9.816514110839844 0.05473282442748082 0.07656488549618322 0.015877862595419834
2023-06-02 23:59:18.766570 1685717958.7666106 0.13563152404785156 -0.008738836059570326 9.833273522460937 -0.15900763358778636 0.13763358778625956 0.03877862595419846
2023-06-02 23:59:18.876985 1685717958.877029 0.027892449340820323 0.031962592163085925 9.809331505859374 -0.0750381679389314 0.09946564885496184 0.07694656488549617
2023-06-02 23:59:18.988358 1685717958.9883988 0.015921441040039074 -0.020709844360351577 9.864398144042967 0.03946564885496174 -0.11427480916030533 -0.16732824427480916
```

Data_time Linux_time Ax Ay Az Gx Gy Gz
순서로 출력 되었음

IMU 센서

MPU6050 로그파일 그래프 생성





IMU 센서

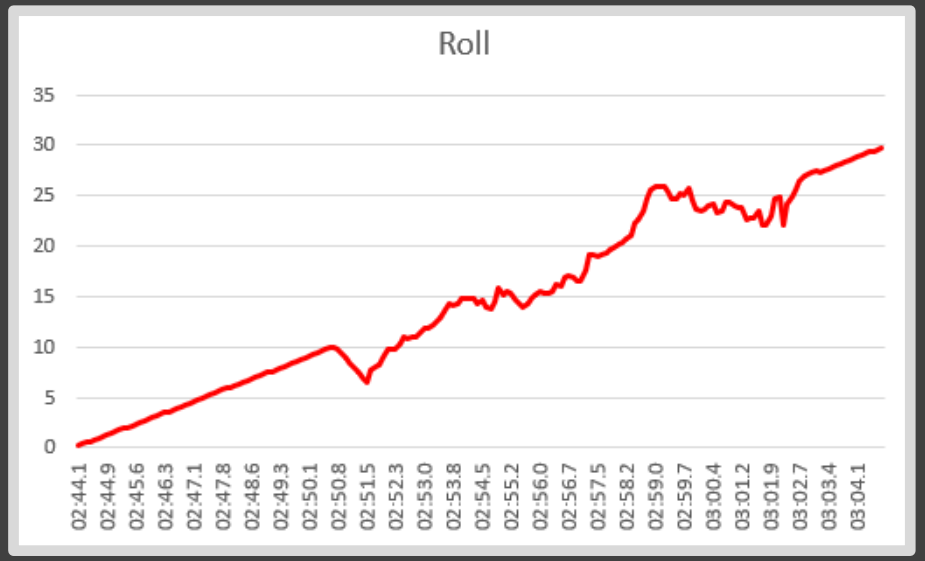
MPU6050 로그파일 그래프 생성

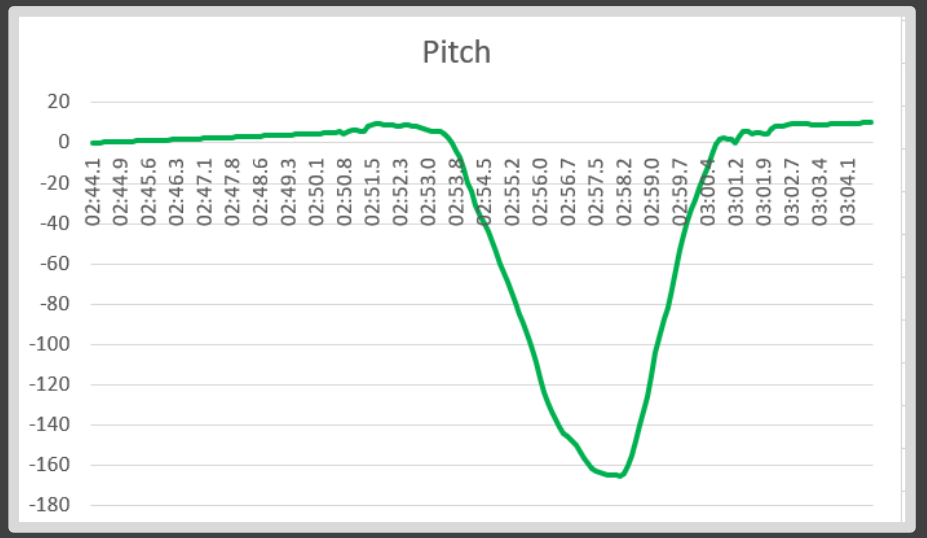
	A	B	C	D	E	F	G	H	I
1	######	02:44.0	1.7E+09	0.03039	-0.0364	9.85656	1.78568	0.66398	0.29
2	######	02:44.1	1.7E+09	-0.0127	-0.034	9.72488	1.51087	0.87772	0.38924
3	######	02:44.2	1.7E+09	-0.0247	0.03061	9.82064	1.51087	0.79375	0.48084
4	######	02:44.3	1.7E+09	-0.0031	0.02822	9.82304	1.46507	0.71741	0.39687
5	######	02:44.4	1.7E+09	-0.0534	0.00667	9.76558	1.48034	0.70978	0.25184
6	######	02:44.5	1.7E+09	-0.1204	0.04019	9.85656	1.25896	0.84718	0.49611
7	######	02:44.7	1.7E+09	0.1501	0.02343	9.84937	1.58721	0.52657	0.36634
8	######	02:44.8	1.7E+09	-0.0319	-0.0316	9.77755	1.4956	0.74031	0.28237
9	######	02:44.9	1.7E+09	0.02081	-0.0029	9.7967	1.63301	0.67925	0.35107
10	######	02:45.0	1.7E+09	0.03518	0.00427	9.62911	1.39637	0.83192	0.29
11	######	02:45.1	1.7E+09	0.01124	-0.0125	9.84219	1.37347	0.56474	0.36634
12	######	02:45.2	1.7E+09	-0.0031	0.033	9.86853	1.37347	0.72505	0.32817
13	######	02:45.3	1.7E+09	-0.0223	-0.0843	9.83022	1.51087	0.68688	0.51901
14	######	02:45.4	1.7E+09	-0.0606	-0.0149	9.79431	1.48797	0.74031	0.38924
15	######	02:45.5	1.7E+09	-0.0223	0.00906	9.88768	1.43453	0.67925	0.46558
16	######	02:45.6	1.7E+09	0.04715	-0.0101	9.81346	1.4269	0.66398	0.38161
17	######	02:45.7	1.7E+09	0.16686	0.01146	9.81825	1.62537	0.71741	0.45794
18	######	02:45.8	1.7E+09	0.00405	0.01864	9.73924	1.53377	0.76321	0.46558

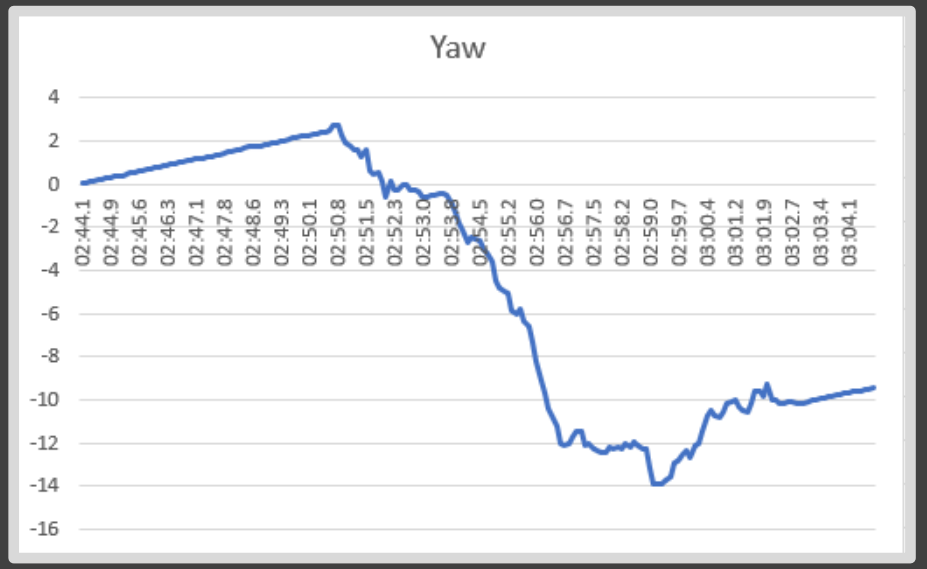
IMU 센서

MPU6050 로그파일 그래프 생성

$$\int \omega dt = \omega \Delta t + C$$







감사합니다

구선생 로보틱스