

myIMU

本專案為使用PC USB透過serial port與IMU主機做溝通，將IMU數據封包接收完後以ros sensor_msgs/Imu 格式 publish出去。

linux 版本: ubuntu 18.04

ros 版本: melodic

python 版本: 2.7

Author: Adam Shiau

Email: bwshiau@ncu.edu.tw

- [myIMU](#)
 - [IMU 規格](#)
 - [ROS /Imu msg 輸入參數](#)
 - [Running the program](#)
 - [step1](#)
 - [step2](#)
 - [step3](#)
 - [API usage](#)
 - [讀取IMU資料](#)
 - [1. 建立類別物件](#)
 - [2. 宣告Callback函數](#)
 - [3. 連接IMU](#)
 - [4. 啟動 thread 讀取IMU數據](#)
 - [中斷讀取IMU數據](#)

IMU 規格

	dynamic range	random walk	Noise Density	type
gyro.x	250 °/s	$0.59\text{ }^{\circ}/\sqrt{hr}$	$35.4\text{ }^{\circ}/hr/\sqrt{Hz}$	MEMS
gyro.y	250 °/s	$0.59\text{ }^{\circ}/\sqrt{hr}$	$35.4\text{ }^{\circ}/hr/\sqrt{Hz}$	MEMS
gyro.z	350 °/s	$0.01\text{ }^{\circ}/\sqrt{hr}$	$0.6\text{ }^{\circ}/hr/\sqrt{Hz}$	FOG
accelerometer.x	$\pm 8g$	$5.3\text{ mm/s}/\sqrt{hr}$	$25\text{ }\mu g/\sqrt{Hz}$	MEMS
accelerometer.y	$\pm 8g$	$5.3\text{ mm/s}/\sqrt{hr}$	$25\text{ }\mu g/\sqrt{Hz}$	MESM
accelerometer.z	$\pm 8g$	$7.7\text{ mm/s}/\sqrt{hr}$	$25\text{ }\mu g/\sqrt{Hz}$	MEMS

ROS /Imu msg 輸入參數

以下參數值可在 `./rosParameters.py` 設定

python var	unit	description
ORI_X、ORI_Y、ORI_Z、ORI_W		Quaternion orientation
COV_ORI_XX、COV_ORI_XY、COV_ORI_XZ COV_ORI_YX、COV_ORI_YY、COV_ORI_YZ COV_ORI_ZX、COV_ORI_ZY、COV_ORI_ZZ		Orientation covariance
COV_W_XX、COV_W_XY、COV_W_XZ COV_W_YX、COV_W_YY、COV_W_YZ COV_W_ZX、COV_W_ZY、COV_W_ZZ	$[rad/s]^2$	Angular velocity covariance
COV_A_XX、COV_A_XY、COV_A_XZ COV_A_YX、COV_A_YY、COV_A_YZ COV_A_ZX、COV_A_ZY、COV_A_ZZ	$[m/s^2]^2$	Linear acceleration covariance

Running the program

在 terminal 執行下列 command:

step1

設定USB port name使用權限(假設port name = /dev/ttyACM0):

```
$ sudo chmod +777 /dev/ttyACM0
```

step2

```
$ roscore
```

step3

```
$ python myImuRos.py ttyACM0 1 1
```

參數名稱	型態	說明
arg[0]	string	程式名稱
arg[1]	string	USB對應的 port name

參數名稱	型態	說明
arg[2]	int	[1/0] : [扣除/不扣除] gyro offset
arg[3]	int	[1/0] : [扣除/不扣除] accelerometer offset

API usage

讀取IMU資料

1. 建立類別物件

建立類別ImuReader的物件, 分別將Port name、gyro offset 校正、accelerometer offset 校正 參數帶入, 參數型態請參考 "Running the program step 3":

```
from imuLib.ImuReader import ImuReader

myImu = ImuReader("Port Name": str, cali_gyro: int, cali_accelerometer: int)
```

2. 宣告Callback函數

宣告一個帶有兩個參數的函數作為 Callback funcion, 之後將此函數註冊於Imureader.setCallback()來接收IMU數據。其中第一個參數為IMU原始數據, 第二個參數為取樣時間1秒的IMU offset值, 此兩參數會接收來自類別ImuReader回傳的dict變數。當類別ImuReader之物件在建立時參數 cali_gyro 或 cali_accelerometer 為 1 時對應的 IMU offset才会有值, 否則為零:

```
#宣告Callback函數
def myCallBack(imudata, offset):
    t = imudata["TIME"]
    fog_wz = imudata["FOG_W"] - offset["FOG_OS"]
    wx, wy, wz = [imudata["MEMS_W"][i] - offset["MEMS_W_OS"][i] for i in range(3)]
    ax, ay, az = [imudata["MEMS_A"][i] - offset["MEMS_A_OS"][i] for i in range(3)]

    print("%.5f, %d, %.5f, %.5f, %.5f, %.5f, %.5f" % (t, fog_wz_dph, wx, wy, ax,

#註冊Callback至Imureader類別
Imureader.setCallback(myCallBack)
```



Key	型態	說明
TIME	float	來自IMU的計時器, 解析度 0.1ms, [s]
FOG_W	float	Z軸FOG角速度值, [°/s]
MEMS_W	float*3	三軸MEMS角速度, [°/s]

Key	型態	說明
MEMS_A	float*3	三軸MEMS加速度 · [g]
FOG_OS	float	Z軸FOG角速度offset · [°/s]
MEMS_W_OS	float*3	三軸MEMS角速度offset · [°/s]
MEMS_A_OS	float*3	三軸MEMS加速度offset · [g]

3. 連接IMU

```
Imureader.connectIMU()
```

4. 啟動 thread 讀取IMU數據

```
Imureader.start()
```

中斷讀取IMU數據

```
#停止thread讀取數據  
myImu.isRun = False  
  
#斷開與IMU的連接  
myImu.disconnectIMU()  
  
#中止thread  
myImu.join()
```