Lab 3
IMU Noise Characterization with Allan Variance

Deadline

10:00 PM on Monday, February 28, 2021

Goals

We would like to understand how to characterize and choose IMU sensors for different robotic applications. As part of the exercise we will write a device driver for the IMU and use that to collect data and analyze it.

Readings

We have discussed sensor calibration and sensor stochastic characterization. The Vectornav whitepaper on the subject is included so that you can refresh your memory.

Hardware / Sensors: Vectornav VN-100 IMU. The user manual for the sensor is provided.

Hardware Setup

Imu udev rules Standard Ubuntu is not designed for roboti application and for fast sensors it may not behave as we want it to. There may be latency in terms of its response to the sensor. There are a set of files under /etc/udev that dictate how Ubuntu will react to certain sensors.

To make the VN-100 plug & play and to address Ubuntu USB latency issues we can do the following

Connect your imu and type the following command in the terminal

$ dmesg | grep vn100

If you do not know what those commands mean, check out Linux tutorial.

You will see your sensor attributes including product id and vendor id. If these values differ from the values below, change them in your 50-VN-100.rules file accordingly.

1. As Sudo create a file under /etc/udev/rules.d called 50-VN-100.rules with the following

```
KERNEL=="ttyUSB[0-9]*", ACTION=="add", ATTRS{idVendor}=="1d6b", ATTRS{idProduct}=="0002",
                                        MODE="0666", GROUP="dialout"
```

2. Ubuntu's default latency of 16ms cause issues with high rate sensors, this can be solved by adding the following UDEV rules as 49-USB-LATENCY.rules

```
   ACTION=="add", SUBSYSTEM=="usb-serial", DRIVER=="ftdi_sio", ATTR{latency_timer}="1"
```

3. Once the rules have been added, to get udev to recognize the rule, run the following command:

```
sudo udevadm control --reload-rules && sudo service udev restart && sudo udevadm trigger
```

Finally unplug and replug the VN-100 to have it work with the new rules.
You can verify the applied settings with below command, should report 1 on success.

$ cat /sys/bus/usb-serial/devices/<ttyUSB0 your device path>/latency_timer

Write a device driver for IMU (individual) and do stationary noise analysis

Open serial port with 115200 baudrate. Configure your IMU to output data at 200Hz for this lab.

Parse $vnymr string, to get accel, gyro, orientation (roll, pitch, yaw) and magnetometer data. Refer to sensor user_manual.

Use the standard ROS sensor_msgs/IMU to publish the above data. Convert the above Yaw, Pitch, Roll data into quaternions and publish it as orientation in the same imu msg. Use ROS sensor_msgs/MagneticField to publish magnetometer data.

Begin collecting a time series data (rosbag) for the 3 accelerometers, 3 angular rate gyros, 3-axis magnetometers. Collect at least 10-15 min of data with the *instrument stationary and as far away as possible from your computer.*

Plot each time series in your report and figure out the noise characteristics of each of the values (mean and standard deviation) reported by the IMU.

Allan Variance Data collection and Analysis

For this part we only need to collect one set of data for each team – collect roughly 5 hours worth of stationary IMU data at a location that is not subject to vibrations (passing trains, building sway etc).

The following MathWorks webpage provides code that you can use to analyze your data for Allan variance. We do not need any further analysis for Allan Variance other than that illustrated in this code.

https://www.mathworks.com/help/nav/ug/inertial-sensor-noise-analysis-using-allan-variance.html

You need to analyze this data using the Matlab functions and should be able to answer the following questions.

1. What kind of errors/sources of noise are present?
2. How do we model them? Where do we measure them? Can you relate your measurements to the datasheet for the VN100?

How to Submit Lab_3

1. In your class repo ‘EECE5554_RoboticsSensing’, create a directory called LAB3
2. Copy the ros driver package used for this assignment under LAB3.
3. Inside LAB3. create a sub-directory called ‘analysis’
4. Place your report in pdf format also in the analysis directory. Your report includes analysis for both the stationary data you collected individually, as well as the Matlab analysis on the data on Allan variance collected as a team.
5. Place any matlab / python code you used to generate and plot your noise parameters in this directory as well.

Your repo structure should look similar to

‘<Path_to_repo>/EECE5554_RoboticsSensing/LAB2/src/<your_imu_ros_driver files>’
‘<Path_to_repo>/EECE5554_RoboticsSensing/LAB2/src/analysis/<your analysis files>’
‘<Path_to_repo>/EECE5554_RoboticsSensing/LAB2/src/analysis/report.pdf’

6. Push your local commits to (remote) gitlab server. You can verify this by visiting gitlab.com and making sure you can see the commit there.

Everyone in the team needs to write their own device driver for the IMU. Make sure everyone gets a fair amount of sensor time to test their imu driver, and to analyze sensor noise. You can use the emulator to write your driver even when you do not physically have access to the device. We have provided a sample IMU data file for your use.

Analysis on the collected dataset, should be done individually.