Lab3 - FYS4240

In this document you can find answers to the following tasks for lab3:

- **Task 1:** 1.7, 1.8, and 1.9
- **Task 2:** 2.6, 2.7 and 2.9
- **Task 3:** 3.4

Task 1

1.7

See Figure 1

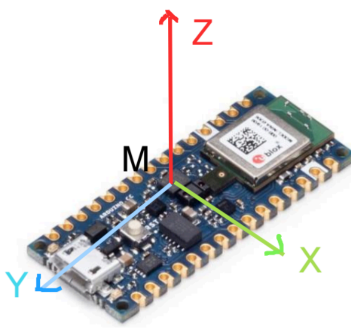
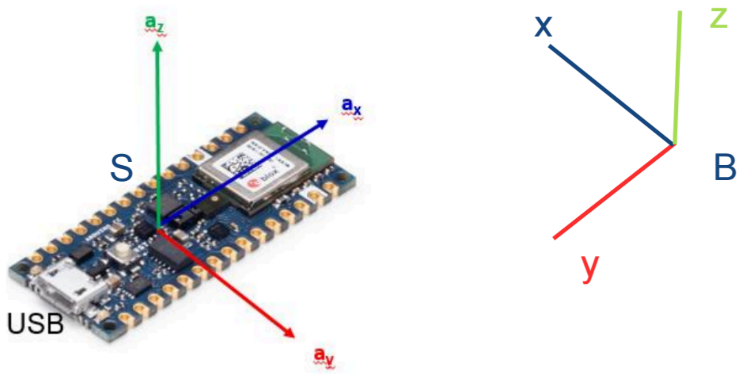


Figure 1: Magnetometer frame, M

1.8

Yes, all of the sensitive sensor axis are adhere to the datasheet(Figure).For the Arduino Nano 33 BLE rev2, the BMM150 and BMI270 were used.

1.9



Given that the transformation matrix $R_S^B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$. Determine how the basis of M is represented in B:

1. **M's x-axis => $B's$ negative x-axis:**

$X_B = -X_M$: first column of R_M^B is $\begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}$

1. **M's y-axis => $B's$ y-axis:**

$Y_B = Y_M$: first column of R_M^B is $\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$

1. **M's z-axis => $B's$ z-axis:**

$-Z_B = -Z_M$: first column of R_M^B is $\begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix}$

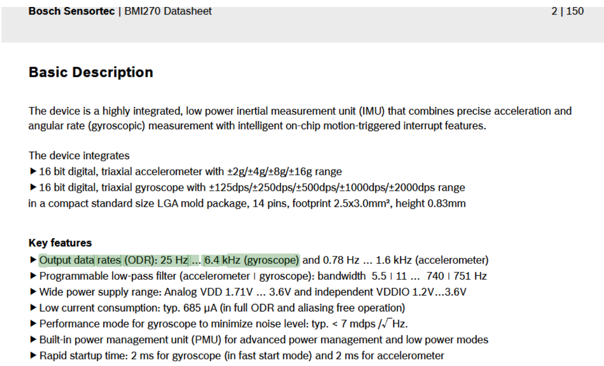
Thus the resulting transformation matrix, $R_M^B = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$.

Assuming that the z-axis pointing upwards gives it a negative component.

Task 2

2.6

Output data rate(loop frequency) is not fixed at 119 Hz, but varied somewhere between 83-125Hz observed from the `SerialDataREad_3CH.vi`. This interval falls within the indicated in the datasheet.


Bosch Sensortec | BMI270 Datasheet 2 | 150

Basic Description

The device is a highly integrated, low power inertial measurement unit (IMU) that combines precise acceleration and angular rate (gyroscopic) measurement with intelligent on-chip motion-triggered interrupt features.

The device integrates
- ▸ 16 bit digital, triaxial accelerometer with ±2g/±4g/±8g/±16g range
- ▸ 16 bit digital, triaxial gyroscope with ±125dps/±250dps/±500dps/±1000dps/±2000dps range

in a compact standard size LGA mold package, 14 pins, footprint 2.5x3.0mm², height 0.83mm

Key features
- ▸ Output data rates (ODR): 25 Hz ... 6.4 kHz (gyroscope) and 0.78 Hz ... 1.6 kHz (accelerometer)
- ▸ Programmable low-pass filter (accelerometer | gyroscope): bandwidth 5.5 | 11 ... 740 | 751 Hz
- ▸ Wide power supply range: Analog VDD 1.71V ... 3.6V and independent VDDIO 1.2V...3.6V
- ▸ Low current consumption: typ. 685 µA (in full ODR and aliasing free operation)
- ▸ Performance mode for gyroscope to minimize noise level: typ. < 7 mdps /√Hz.
- ▸ Built-in power management unit (PMU) for advanced power management and low power modes
- ▸ Rapid startup time: 2 ms for gyroscope (in fast start mode) and 2 ms for accelerometer

Figure 3: Facsimile - Bosch Sensortec | BMI270 Datasheet

Note: The program in `.vi-file` does not run while the `Serial Monitor` or `Serial Display` in the Arduino IDE are open. Only a single program can access the `COM`-port at a time.

2.7

The frequency varies between 46-80Hz, which is lower than the one in previous task. I think this may happen because within the loop frequency the application in labView both reads from the serial port and writes the read data to `test1.txt`-file. I reckon the operations run at different rates, hence the write-operation slows the whole program as the read-operation has to "wait" for it to finish writing to file. That is, the write-operation is slower than the read-operation. While in task 2.6 the frequency is higher as it only contains a read-operation.

2.9

The loop frequency shows the same measurements obtained in **2.6**, displaying values between 83-125Hz. These values are not absolute as the update rate of the frequency is too fast to read. By introducing a producer-consumer-structure we enhance the data sharing between the producer- and consumer-loops, which run at different rates. These processes run now in parallel and the quicker read-operation no longer needs to wait for the write-operation to finalize. As a result, the loop frequency equals the quickest operation, i.e. the read-operation and we obtain similar results as in **2.6**.

Task 3

3.4

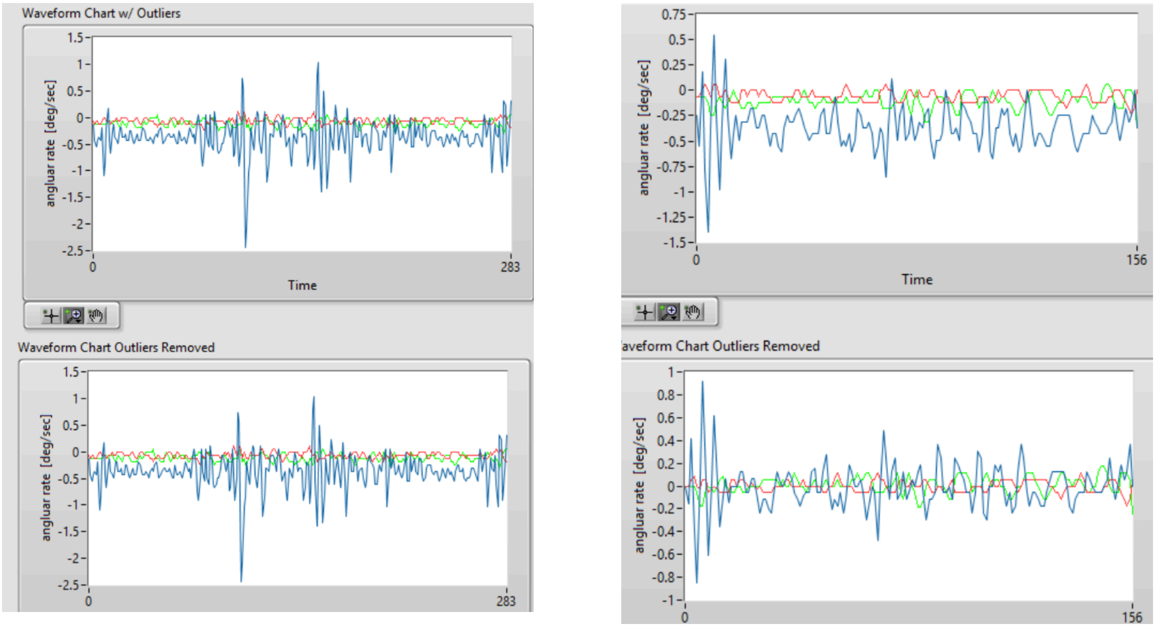


Figure 4: Left - Displays the angular rates without the *"remove initial offset"* ticked off, producing identical charts. Right - Shows the angular rates with the offset. The bottom chart displays a vertical shift in blue graph as a result of removing of outliers using `RemoveMean.vi`.

There can be several reasons for the drift in the measured angle, even though we have taken into account the outliers. For instance, we have been working with a gyroscope bias, i.e. **zero-offset error** - expecting the measured quantity to be zero, where the reading deviates from this zero point. This introduces a small bias in the gyroscope reading, which over time leads to a slow accumulation of error, resulting in drift in the angular rates.

Another potential source of error, is the **numerical integration errors**. The gyroscope measures angular velocity, where the angle is acquired by integrating these values over time. Any bias or or noise in the measurements are also integrated readings which are integrated. Once more, these errors accumulate and result in drift.

I imagine these sources of error have the highest likelihood to occur in the readings. Below is a list of other potential contributors to the observed drift:

- *Temperature Variations within the Hardware*: Temperature changes or longer use can affect the long-term stability of the sensor. The IMU has not undergone any particular long time operation, so sensor heating is not a contributor here.
- *External disturbances*: Again the sensor being exposed to high external temperatures can be omitted discarded; Vibration and external disturbances tend to induce mechanical vibrations in the gyroscope resulting in noise and outliers.