Hardware support for research of the sensor fusion of inertial sensors

1st Ján Andel, 2nd Vojtech Šimák
Dept. of Control and Information Systems
University of Žilina
Žilina, Slovakia
jan.andel@feit.uniza.sk, vojtech.simak@uniza.sk

2022 International Conference on Applied Electronics (AE) | 978-1-6654-9482-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/AE54730.2022.9920062

***Abstract*— The aim of this paper was to propose a design of a module that has several inertial sensors of the same type in order to test various approaches of homogeneous sensor fusion. According to the statistics the mean of readings from the same-type sensors should have higher precision than a single sensor.However, this statement is not always correct for real sensors, as identical sensors may not have the same error characteristics. Sensor manufacturers state the typical sensor RMS (root mean square) error, the actual sensor RMS error can differ significantly from piece to piece. When averaging a sensor output from the same manufacturer, we can under certain conditions, obtain a worse value than the output error of the best sensor. This error can be eliminated by fusing the sensors using weighing. To verify this statement, we decided to assemble with as many identical sensors as possible. The IMU (inertial measurement unit) sensor, which measures acceleration, angular acceleration, and magnetic field in three axes, was chosen as the sensor for the variety of measurements. Thanks to this, we can compare up to 9 different outputs at the same time. In the end, we designed a module that has 16 IMUs. As the number of sensors increases, the resulting error decreases on average. However, weighting based on calibration errors did not prove to be the optimal solution because the sensors contain not only stochastic but also systematic errors. The module designed by us will be used mainly for further scientific research in the field of IMU sensor fusion in order to reduce the error.**

Keywords— MSE, IMU, data fusion, sensors, error

I. Introduction

The theory is that the larger the number of identical sensors, the more accurate the resulting value measure. In practice, however, we can often find that after adding another sensor, we get an output measured value worse than at the beginning, many times we do not even know about it. This is the case when the added sensor has a worse systematic error but still does not exceed the maximum error specified by the manufacturer. When connecting several sensors, we can weigh their input values to obtain the most accurate output value. The weights of individual sensors can be given, for example, based on the MSE (Mean squared error) of the sensor obtained during calibration. The main goal of this paper is to propose a design of hardware and software tool allowing the research and optimization of the homogeneous sensor fusion algorithms. To find the optimal algorithm for weighing real sensors, we designed a module that has multiple sensors from the same manufacturer. For our purposes, we chose the IMU unit as a sensor because it contains up to 9 sensors (3 axes x 3 sensors).

As a result, with the right scales, we can get a more accurate result from more inexpensive sensors than from one more expensive sensor. At present, in the absence of semiconductors, such a solution is also an advantage.

When choosing a sensor, we thought of several options. We thought about sensors for humidity, pressure, light intensity, and much more. However, the best alternative was to use an IMU unit that would contain 3-axes: an accelerometer, a gyroscope, and a magnetometer. With such an IMU, we would be able to verify the weighting algorithm for up to 9 different output values. For better verification, each IMU should also have different accuracy. Cheap IMU sensors have different accuracies. For this reason, we decided to use the MPU9250 sensor that meets our requirements. MPU9250 is one of the most used IMU units, it is very small, its size is 3x3x1mm. Includes accelerometer, gyroscope, magnetometer and thermometer. It can measure acceleration in all three axes up to 16g, angular acceleration up to 2000 °/s and magnetic field up to 4800uT. These cheap IMUs suffer from several sources of systematic and stochastic errors, such as bias, bias instability, random angular walk (gyroscopes) and velocity random walk (accelerometer), quantization, scale factor (SF), etc. (Figure 1) [1, 11]. We can reduce the sensor error in several ways. Many articles have been published on this topic, but the best-known approaches include the use of simple filters [3,4], Kalman filter [5,6], sensor weighting [7,8], data fusion [9], or the use of neural networks [10].

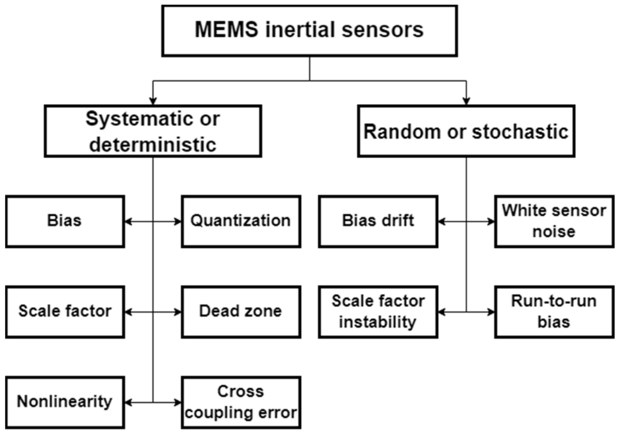

Figure 1 Different types of sensor errors [11]

II. Module Design

IMU sensors must be placed as close to each other as possible to eliminate the effects of distance, especially with acceleration sensors, also when measuring the magnetic field that can be curved in space (building). For this reason, we decided to make the module two-stage. At the bottom will be a microcontroller with a connector for power and communication. At the top will be IMU sensors. The top plate with IMU should contain as many IMU units as densely as possible. After careful consideration, we decided to use 16 IMUs in a 4x4 array mounted on a four-layer printed circuit board. The distance between the sensors was 5 mm, if we used more sensors, the distance between the sensors increased. The lower module contains an STM32 microcontroller which should guarantee communication with the sensors via SPI communication and then send raw data via UART at high transmission speed. At the same time, the microcontroller has large computing power, so after finding the optimal weighting algorithm, the data can be processed directly by the microcontroller. In case of insufficient power, it is possible to change the bottom plate and the matrix of the sensors will remain the same.

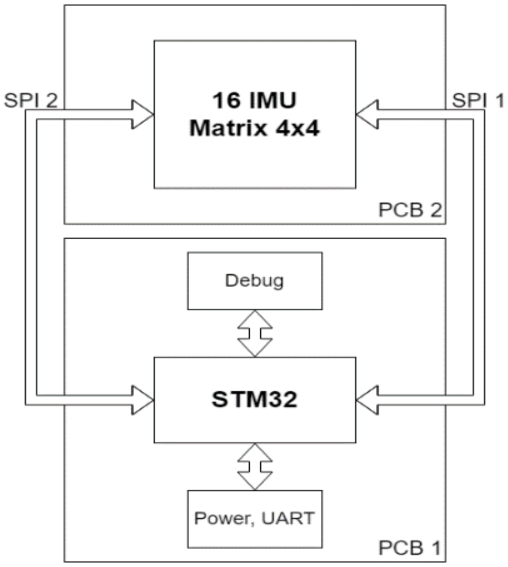

Figure 2 Block diagram

Figure 3 The complete module with 16 IMU sensors

A. Design of hardware solution

Based on the block diagram, we designed a wiring diagram and then designed the printed circuit boards. The top board is four-layered and contains connection pins, IMU sensors, and the necessary components to the sensors to manage functionality. According to the block diagram, the sensors will be connected via two SPI communications. For the possibility of speed-up reading in case of a change of the bottom plate, we designed the connection of IMU sensors for the possibility of connecting up to four independent SPI communications. The size of the board is 30x22 mm. The bottom plate contains two connectors. The first is used for communication and power supply for data provision, the second connector is for service and is used for debugging the device. There is an STM32F446 microcontroller on the module which communicates with the IMU units of two independent SPI communications. The size of the bottom plate of the module is 30x28mm. The module can be seen in Figure 3. Using a linear stabilizer, we can supply the module in a wide range, 3.3 - 12V. At a supply voltage of 5V, the consumption of the module is 97mA (after initialization of all sensors). I / O connectors are protected by diode arrays designed to protect high-speed data interfaces. The microcontroller protects against ESD (electrostatic discharge), CDE (Cable Discharge Events), and EFT (electric fast transitions).

B. Firmware design

We wrote the program for the microcontroller in the C programming language, we used the hardware components of the SPI, UART microprocessor, and DMA peripherals with HAL drivers from STM. The program initializes all peripherals and then initializes the sensors which set the configuration registers. We set the range of measuring quantities and internal filters in the registers. We set the measuring range for the accelerometer to the level of 4 g, we set the gyroscope to the value of 250 °/s. The built-in filters are low-pass and their setting can be done separately on the accelerometer and separately on the gyroscope. In the case of the gyroscope, we set the cut-off frequency of the filter to 41Hz and we set the accelerometer to 44.8Hz. The magnetometer does not include the possibility of setting the range or the filter. The magnetic sensor that is part of the MPU9250 communicates via I2C communication. When communicating via SPI, it is necessary to set the internal peripheral to read data from I2C and write to the register for subsequent reading via SPI. After successful initialization, an infinite loop is started in which the program reads data from all initialized IMUs (16 IMUs if everything went correctly). It then parses the data into a single packet and sends it via UART. We transfer raw data from the sensors via UART to save data. If the processed data in the string was sent, the string would be many times larger and the sending time would increase. The sample from one sensor contains 3x3x2B = 18B at 16 IMU units per sample 16 * 18B = 288B. We send data via UART via the DMA channel, thanks to which the microprocessor can simultaneously read data from sensors and send data via UART. We set the UART communication baud rate to 576000 bits/s. The transmission of one packet takes 5ms, which limits us to a maximum sampling frequency of 200Hz. In order to be able to process data in the superior system, we set a sampling frequency of 100Hz, which achieves a data rate of 288000 B/s. We are currently limited by the maximum UART communication speed. Increasing the speed could already cause bad data reading. In the event that we would like to increase the sampling frequency of reading data from the sensors, it would be necessary to change the communication with the superior device or to send only certain data. For example, only data from a gyroscope. The module with IMU units is designed to be read via four SPI communications. In case we would like to increase the sampling frequency, for example to 1kHz, it would be necessary to redesign the bottom plate. It would contain a microcontroller that could read data from IMUs through four independent SPI communications with DMA channels and then send the data, for example, via USB. However, the current solution is sufficient for our research.

C. Software design

We can use several tools for data processing. First, it is necessary to process RAW data from the module and save it for further processing. For this purpose, we designed a WinForm application in Visual Studio using the C # programming language (Fig. 5). The application allows us to select the COM port through which we read data from the module. The friendly data must be matched to real values, which will then be written to a "csv" file together with a timestamp for further processing. For data analysis, it is necessary to go through a predefined path/rotation in order to know the reference value and compare it with the data from the IMU. For this purpose, we have designed a device that rotates a stepper motor to a precisely defined angle with a precisely defined speed (Fig. 4). It sends the current rotation position via the UART interface. We then placed the module with IMU units on the motor. We have programmed the motor control and predefined motor rotation paths into the WinForm application. The application, therefore, receives data in parallel with the module with the IMU and at the same time from the engine control unit. It merges the data and writes it to a "csv" file for further processing. The application also allows us to view the last measured data. However, the data is not displayed in the WinForm application, but the application runs a python script that displays it. The data is displayed in three windows, each window contains a different type of sensor, in each window, there are three graphs (three axes) with data from all sensors.

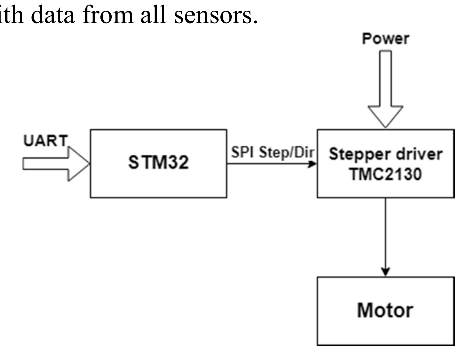

Figure 4 The motor position control diagram

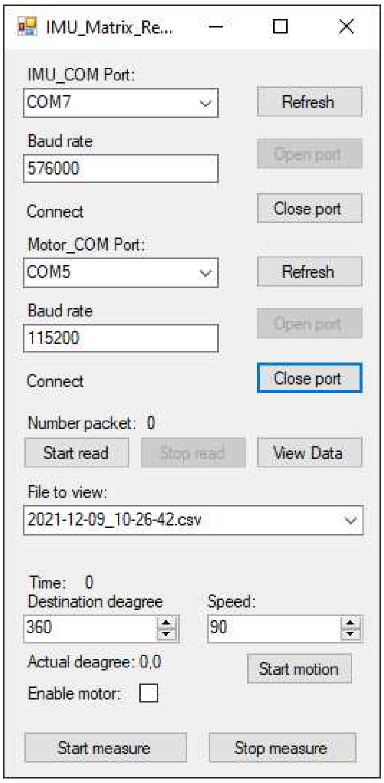

Figure 5 WinForm application for reading data

III. Data analysis

We decided to use the python scripting language for data analysis due to many libraries and programming speed. When analyzing the data, we focused on the z-axis of the gyroscope, which measured the angular speed of rotation on the stepper motor. By deriving the current position of the motor shalf, we were able to calculate the angular velocity of the engine and then compare it with the data from the *z*-axis of the gyroscope. Motor position data and IMU data were recorded at a frequency of 100Hz. We performed the measurements along a predefined path. The measurement lasted 5 minutes, during which the engine stood for the first 20 seconds to calibrate the bias, and then the engine rotated in a predefined path (Fig. 6). By calibrating the bias, we eliminated a systematic error. After subtracting the bias, we determined the MSE from the individual sensors from the calibration time. Based on this error, we weighed the sensors and monitored the output errors after connecting the sensors with the same or different weights. Preliminary results have shown that weighing is not effective and that the use of the same scales or the use of only certain sensors appears more accurate. In some cases, adding a sensor increased the error. Each sensor had a different error.

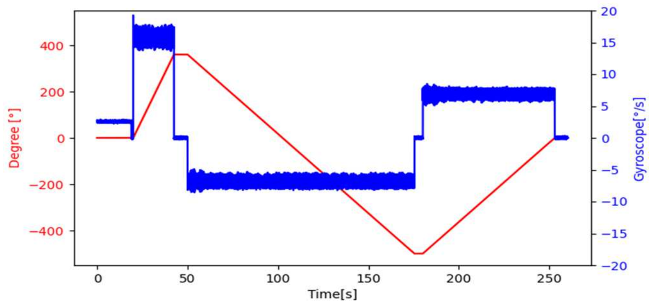

Figure 6 Output data graph, actual angle, gyroscope sensor data

A. Simulation

To verify the correctness of the programmed scripts, we created another script that allows us to generate data from the gyroscope along a predefined path with a certain error. To simulate a systematic error, we added bias to the data, to simulate a stochastic error, we added a random error to the data. We have verified on the data that the scripts are working properly. In the case of generating data with the same RMSE (Root Mean Square Error) noise, the total error according to formula (3) was reduced. In case we gave each sensor a different RMSE error, it turned out to be the optimal solution to give the weight to the individual sensors according to the dependence of the MSE error, specifically the inverse value of the MSE error. In this case, the total MSE value of the error was reduced according to formula (2). In this way, we verified that the programmed scripts work correctly and according to statistical assumptions. It follows from formula (4) that in the case of using sensors that contain only a random error of the same magnitude, the total error decreases. Using 16 sensors would reduce the RMSE error by four times over. This approach to reducing sensor noise is well known and has been described in [9] [12].

$$MSE\ (y) \approx \sum_{k=1}^{n} \left(\frac{\partial f}{\partial x_k}\right)^2 * MSE(x_k) \qquad (1)$$

$$MSE\ (y) \approx \frac{1}{\sum_{k=1}^{n} \frac{1}{MSE(x_k)}} \qquad (2)$$

$$MSE(y) \approx \frac{MSE(x)}{n} \quad (3)$$

$$RMSE(y) \approx \frac{RMSE(x)}{\sqrt{n}} \quad (4)$$

IV. Discussion

After verifying the functionality of the scripts based on the theory, we analyzed the data. The total sensor error decreased on average, however, not according to (2). This was because there is not only a stochastic error in the system but also a systematic one. Respectively, in the case of simulated data, we added a systematic error in the form of a bias, but in real data, a bias is not the only systematic error. As mentioned, IMUs contain several sources of error. We rotated the sensor module with a stepper motor with micro-steps which had 32,000 micro-steps per revolution. However, we did not achieve a completely stable angular velocity with the engine. The gyroscope data showed that the angular velocity was slightly contained spikes. This factor also increased the overall systematic error. We performed a total of 100 measurements. As a result, 96% of measurements had the lowest measurement error if the sensors had the same weight. At 2% of the measurements, the smallest error using weights was based on the 1/MSE calibration error. At 2% of the measurements, the smallest error was when using one particular sensor. Such a result has been expected and is described in several scientific literatures. On data fusion, we have verified that our module works correctly and provides relevant data. The module will be useful for further research.

V. Further direction

In the further direction of the project, we would like to focus on the use of all three axes and all three types of sensors. Respectively, after finding the optimal weighing algorithm, we verify the algorithm on the data from other sensors. We would like to use a unit of 16 IMU sensors in low-cost INS research, which could be an advantage in the current chip crisis. The module we designed has sufficient computational power to be able to integrate a simpler fusion algorithm in real-time. Furthermore, we would like to suppress the systematic error, for example by changing the drive and further look for the optimal weighting algorithm of individual sensors in order to provide the most accurate data. We would also like to take a closer look at the possibilities of using artificial intelligence tools to search for the optimal sensor weighting algorithm.

VI. Conclusion

The main aim of this work was to design a module with as many sensors as possible to research data fusion from the same real sensors. We designed and built a module that contains 16 IMU sensors that contain three-axis acceleration, angular acceleration, and magnetic intensity sensors. We read data from IMU via two independent SPI buses. Thanks to that, we can synchronize the data. The board from IMU can communicate through up to 4 independent SPIs. We designed an application that reads data from the module and writes to a file. The currently set sampling frequency of the sensors is 10Hz. At the same time, we created a platform that uses a stepper motor to rotate the module on the rotational table. Subsequently, we investigated the weighing options from the data from the gyroscope axis. We provided the scales to individual sensors based on their MSE value during calibration. By increasing the number of identical sensors, we obtain the resulting measurement error. However, this statement applies as long as the sensor does not have a fundamental systematic error. If we suppress all systematic errors in the sensor, the resulting sensor error will be reduced according to formula (2). However, this is impossible with real sensors. In our measurements, we found that if we were unable to eliminate all systematic errors, weighing the sensors based on MSE errors during calibration will not yield better results. It is better to give all sensors the same weight. However, this approach is well known. Our goal was not to propose a new fusion algorithm but to design a module with as many of the same sensors as possible. We encountered several problems during the design and construction that we managed to solve and we met the goal. We will use the proposed module for further scientific research of the optimal weighing algorithm.

VII. Acknowledgement

This article was created with the support of the project VEGA 1/0241/22 Mobile robotic systems for support during crisis situations.

References

[1] M. El-Diasty, A. El-Rabbany and S. Pagiatakis, "Temperature variation effects on stochastic characteristics for low-cost MEMS-based inertial sensor error", Meas. Sci. Technol., vol. 18, no. 11, 2007.

[2] Han S, Meng Z, Omisore O, Akinyemi T, Yan Y. Random Error Reduction Algorithms for MEMS Inertial Sensor Accuracy Improvement—A Review. Micromachines. 2020; 11(11):1021. https://doi.org/10.3390/mi11111021.

[3] Han, K.H.; Guo, W.L.; Gao, X.Q. A noise removal method for MEMS gyroscope. In Proceedings of the IEEE International Conference on Computer Science and Information Technology (ICCSIT 2010), Chengdu, China, 9–11 July 2010; pp. 274–277.

[4] Guo, X.; Sun, C.; Wang, P.; Huang, L. A hybrid method for MEMS gyroscope signal error compensation. Sens. Rev. 2018, 38, 517–525

[5] Hu, Z.X.; Gallacher, B. Extended Kalman filtering based parameter estimation and drift compensation for a MEMS rate integrating gyroscope. Sens. Actuators A Phys. 2016, 250, 96–105

[6] Feng, Y.B.; Li, X.S.; Zhang, X.J. An Adaptive Compensation Algorithm for Temperature Drift of Micro-Electro-Mechanical Systems Gyroscopes Using a Strong Tracking Kalman Filter. Sensors 2015, 15, 11222–11238

[7] Sheng, G.R.; Gao, G.W.; Zhang, B.Y. Application of Improved Wavelet Thresholding Method and an RBF Network in the Error Compensating of an MEMS Gyroscope. Micromachines 2019, 10, 608

[8] Lu, Q.; Pang, L.X.; Huang, H.Q.; Shen, C.; Cao, H.L.; Shi, Y.B.; Liu, J. High-G Calibration Denoising Method for High-G MEMS Accelerometer Based on EMD and Wavelet Threshold. Micromachines 2019, 10, 134

[9] Nemec, D.; Janota, A.; Hrubos, M.; Simak, V. Intelligent Real-Time MEMS Sensor Fusion and Calibration. IEEE Sens. J. 2016, 16, 7150–7160

[10] Zhu, M.; Pang, L.X.; Xiao, Z.J.; Shen, C.; Cao, H.L.; Shi, Y.B.; Liu, J. Temperature Drift Compensation for High-G MEMS Accelerometer Based on RBF NN Improved Method. Appl. Sci. 2019, 9, 695

[11] Elhoushi, Mostafa & Georgy, Jacques & Noureldin, Aboelmagd & Korenberg, Michael. (2016). A Survey on Approaches of Motion Mode Recognition Using Sensors. IEEE Transactions on Intelligent Transportation Systems. PP. 1-25. 10.1109/TITS.2016.2617200.

[12] H. Teodorescu and M. Hagan, "High Accuracy Acceleration Measuring Modules with Improved Signal Processing Capabilities," 2007 4th IEEE Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 2007, pp. 29-34, doi: 10.1109/IDAACS.2007.4488366.