Proceedings of the 2023 Winter Simulation Conference
C. G. Corlu, S. R. Hunter, H. Lam, B. S. Onggo, J. Shortle, and B. Biller, eds.

SENSOR FUSION DEVS FOR ANGLE ESTIMATION ON INERTIAL MEASUREMENT UNIT

Gabriel Wainer
Joseph Boi-Ukeme
Vedant Paranjape

Department of Systems and Computer Engineering
Carleton University
1125 Colonel By Drive
Ottawa, ON K1S 5B6, CANADA

2023 Winter Simulation Conference (WSC) | 979-8-3503-6966-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/WSC60868.2023.10407940

ABSTRACT

We explore the application of a Sensor Fusion Framework, called SAFE (Simple, Applicable, Extensible, and Flexible) to improve the reliability of measurements obtained from Inertial Measurement Unit (IMU) sensors. SAFE is built using a DEVS specification and the Cadmium tool. Measuring angular position is a difficult task due to the unreliability of gyroscopes and accelerometers, two sensors widely used to measure angles. Although angular position can be measured using imaging systems, these are costly, and not ideal for handheld and portable devices. An alternative solution is to use sensor fusion to fuse the readings of both accelerometer and gyroscope, obtaining reliable readings. We show the application of the SAFE methodology and the results of our case study showing the potential of this method.

1 INTRODUCTION

Over the past few decades, growing public awareness of healthcare, physical activity, safety, and environmental detection has created a new need for smart sensor technologies and surveillance devices that can detect, classify, and provide quick feedback to a user. They aim to assess environmental and safety conditions in a broad, accurate and reliable manner. For example, monitoring and accurately quantifying a user's physical activity using an inertial measurement unit (IMU)-based device has proven important in the health care of patients with chronic illness (Ometov et al. 2021). IMUs fall under the classification of Microelectromechanical Systems (MEMS). MEMS devices are becoming popular in safety-critical applications and their reliability has become important (Aggarwal 2010). There has also been work done about failure and fault-tolerant design that could be useful in looking at the reliable design of CPS, however, the tight interconnectivity of CPS presents new challenges; for example, distinguishing between a fault and uncertainty or measurement noise, handling multiple faults in various subsystems, isolating faults in a large system, preventing faults from causing failure, and accommodating faults in real-time.

There has been some effort to improve the reliability of MEMS by employing sensor fusion, which allows for the combination of the variables from different types of MEMS to complement each other (Megnusson and Odenman 2012). One commonly used method employed to improve the accuracy of MEMS readings is complementary sensor fusion. Other methods include Kalman-based methods, wavelet-based methods, machine learning, and deep learning (Han et al 2020). The Kalman and wavelet-based methods work by suppressing random noise while the machine learning and deep learning methods use the data from the IMU sensors to build data driven models that can minimize errors. Complementary sensor fusion provides some degree of accuracy in sensor readings obtained from MEMS but cannot guarantee the reliability of the output when one or more of the input sensors fail (Tseng et al. 2011).

There has been some work done in adopting formal modeling and simulation techniques for these kinds of applications. In particular, this research focuses on investigating the use of a well-established formal modeling technique, DEVS (Discrete Event System Specification), in these kinds of models. The main reason is because DEVS has the potential of ensuring model evolvability, and the original model used for simulation can be used in specialized hardware to be part of the end product in the target hardware, and formal models can be used to study the interaction with the physical environment. This is done by gradually replacing models with hardware surrogates and new software components without changing the original models. We show how to do this. The idea is to model the system of interest using DEVS theory by properly describing the systems requirements and the environment in which it operates after which these models can be formally validated. Simulations and tests can be run by using sensor data and when the behavior of the model(s) is satisfactory, the models can be reused in a specific hardware target.

The research is based on early work by (Boi-Ukeme and Wainer 2020), which introduced SAFE (Simple, Applicable, Extensible, and Flexible), a framework built as DEVS to improve sensor fusion reliability by providing a formal way to handle faults within a modeling framework. We want to test new methods for fault-tolerance in the sensing system provided by the IME through a sensor fusion framework, and to explore the use of sensor fusion techniques in DEVS. To do so, we use a framework with essential properties relevant to sensor fusion and integrate it with a real-time DEVS simulator (E-Cadmium). The various aspects of our research consider the definition of a the IMU model, a method for real-time fault detection at the design stage using DEVS and at the operation stage through data driven techniques, and how to facilitate the use of sensor fusion techniques in DEVS by developing a framework using the essential properties relevant to sensor fusion. The research shows how to use the SAFE in the field of MEMS to improve the reliability in the measurement obtained from an IMU by employing redundancy. We evaluate the functionality of combining sensor fusion algorithms proposed in the SAFE framework by combining two different algorithms: the complementary filter and the majority vote algorithm. We show that these algorithms can first be simulated and later easily deployed into a hardware prototype for a real-world application using real sensors and microcontrollers. We provide a simulation and prototype hardware environment that can serve as a tool for testing different aspects of the SAFE framework for improvement and to evaluate the applicability of the proposed framework.

The body of the paper is organized into five parts, in section 2 we discuss the background and related work, in section 3 we present the case study, the hardware design is discussed in section 4, the results are presented in section 5 while the discussion and future work is presented in section 6.

2 BACKGROUND

In (Boi-Ukeme and Wainer 2020), a framework called SAFE (Simple, Applicable, Extensible, and Flexible) was proposed for combining sensor fusion algorithms and configurations. SAFE improves the reliability of sensor systems by sensor fusion. SAFE facilitates different sensor fusion configurations and the combination of sensor fusion algorithms to meet fusion objectives. It also allows us to have a library of algorithms that simplifies the selection and implementation of Sensor fusion within embedded systems. The framework is shown in Figure 1.

The SAFE framework is organized in two layers: the sensor layer, and the fusion layer. In the sensor layer, data is collected by sensor type while in the fusion layer, sensor fusion algorithms are applied based on the configuration selected. SAFE is defined using DEVS for formally specifying the components of the framework, and Cadmium (Belolli et al. 2019) to implement the environment, including sensor fusion algorithms and failure checking algorithms. They are built as DEVS models that can easily be coupled with other DEVS models, ensuring that the sensor fusion models are isolated from the simulation and real-time execution engines. In a sensor system measuring a given property, SAFE will receive inputs from different sensors and give out an output with the property measured after fusion algorithms are applied. As these are built as DEVS models, the sensors and actuators can be simulated if not available, and later ported to a real-world hardware platform. Cadmium is implemented as a library written in C++, compliant with the C++17 and the Boost library coding standard. It supports multiple data types for the Time, and Messages, and compiles in multiple platforms, including Linux, Windows, and Mac OS.

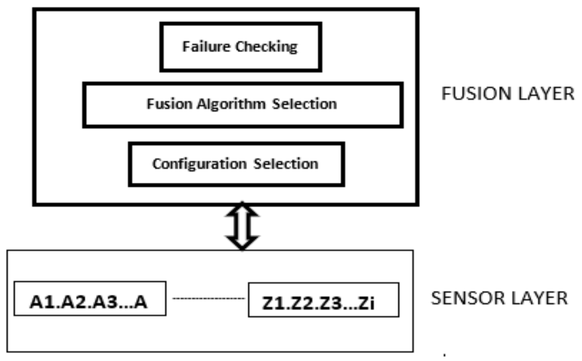

Figure 1: SAFE framework

We will use SAFE to provide safety-critical measuring of angular positions. This is a challenging task due to the unreliability of the two commonly used sensors to measure angles: gyroscopes and accelerometers. An accelerometer is a sensor that measures acceleration of a body in its instantaneous rest frame. A gyroscope is a device that measures angular velocity. For instance, in robotic applications, tilt detection is essential for balance control, and it can be measured using these devices. However, it is challenging to use each of these devices by themselves. On one hand, accelerometer data is generally noisy and susceptible to external acceleration interference, making it difficult to obtain accurate results in vibrating environments such as cars and airplanes. However, since the noise is random, over a long period, the errors average out and the readings become reliable, making the accelerometer data stable. The short-term behavior of accelerometers makes them unsuitable for real time observation. The gyroscope, on the other hand, provides an angular velocity around three axes, and the signal is not affected by external forces making it less susceptible to noise when compared to the accelerometer. Although the gyroscope readings are fairly reliable, they still have some amount of error. This error is irrelevant for instantaneous readings but becomes significant when integrating the values. It simply adds the change in angular position at every instant of time and over a long period, this error can grow and be large enough to render the readings incorrect. In summary, gyroscope readings are unreliable over a long period, but fairly accurate over the short term. Angular position could also be measured using imaging systems, but these are expensive and not suitable portable devices.

Considering these issues, our goal is to use SAFE and build DEVS models including sensor fusion capabilities to fuse readings from both accelerometers and gyroscopes to get reliable readings, even using inexpensive inertial measurement unit (IMU) sensors. The platform allows simulating the sensor data and when ready, execute the models in a target hardware using embedded Cadmium. Given the strengths and weaknesses of gyroscopes and accelerometers, IMU data fusion is a good solution for reliable tilt detection. Several authors have combined data from IMUs like accelerometers and gyroscopes to measure tilt using data fusion algorithms (Gui et al. 2015).

The most widely used data fusion algorithms for IMUs include the Complementary Filter and the Kalman Filter (Han et al. 2020). The Kalman filter is an iterative filter, which is efficient but very complex to calculate, while the Complementary filters are adequate for embedded platforms with small memory footprint and simple processors (Islam et al. 2019). These data fusion algorithms go a long way in improving the accuracy of data measurements from IMUs (Mumuni and Mumuni 2021). However, these methods have no way to guarantee the reliability of fault tolerance when the input sensors fail.

There have been studies that suggest sensor replication techniques to improve reliability in these types of systems (Han et al. 2020). However, incorporating these techniques would necessitate the combination of competitive sensor fusion algorithms with the existing complementary filter technique.

2.1. IMU Sensors

The IMU (Inertia Measurement Unit) sensors measure angular rate, force, and sometimes magnetic field. They use a 3-axis accelerometer (which measures the acceleration of a body in its instantaneous rest frame), a 3-axis gyroscope (a device that measures angular velocity), and in some cases a 3-axis magnetic field to measure acceleration, rotational speed, and even the earth's magnetic field to determine the orientation of an object. Due to its unique characteristics, the IMU has long been the subject of extensive research in the fields of aerospace (Sahawneh and Jarrah 2008) and navigation (Sukkarieh et al. 1999). In recent years, with the advent of MEMS-based IMUs, the size of IMUs has been dramatically reduced, along with their cost and power consumption. They are now applied in several domains including robotics, human motion analysis, and consumer handheld devices (Gui et al. 2015).

As discussed earlier, an accelerometer measures the linear acceleration of the device to which it is attached. The angular position (ϕ_A) is calculated as follows using an accelerometer reading:

$$\phi_A = arctan(\frac{A_y}{\sqrt{{A_x}^2 + {A_z}^2}}) \cdot \frac{180}{\pi}$$

This gives the angular position on the Y-axis. Here, A_x, A_y and A_z are the linear acceleration along the respective axes, and can be interchanged to find the angular position in the X and Z axis respectively.

A gyroscope measures angular velocity (ω). Since angular velocity is not enough to give us the angular position (ϕ_G), we need to integrate angular velocity to get the angular position. Considering that θ is the position angle, the angular velocity is calculated as:

$$\omega = \frac{d\theta}{dt}$$

So that

$$\phi_G = \theta(t) = \int_0^t \omega \cdot dt \approx \sum_0^t \omega \cdot \Delta t$$

2.2. Complementary Filter

A complementary filter is a sensor fusion algorithm that uses a low pass and high pass filter to fuse two separate readings to get a reliable result. As we discussed earlier, the readings of an accelerometer are reliable over the long term whereas they are too noisy over the short term, whereas for the gyroscope it is the opposite. A complementary filter looks to exploit the good features of the accelerometer and gyroscope. In the short term, data from gyroscopes is used as it is very reliable and not susceptible to external forces. In the long term, data from the accelerometer is used as it does not drift. So, we apply a low pass filter on accelerometer data and a high pass filter on the gyroscope data and then combine the results, this is the basic idea behind a complementary filter (Narkhede et. al 2021).

$$\phi = \alpha \cdot \phi_G + (1 - \alpha) \cdot \phi_A \qquad \text{where } 0 < \alpha < 1$$

The above equation describes the formula of the complementary filter. If we use higher values of α, more weight is given to the accelerometer value and less to the gyroscope value. A low pass filter is applied to the accelerometer values and a high pass filter on the gyroscope values, and then these signals are added to get the final signal ϕ, which depends on the value of α. The best value of α can be determined by trial and error, but in practice, most designers set the value of α to be greater than 0.6 (Kok, Hol and Schön 2017)

2.3. Majority Vote Algorithm

A Majority vote algorithm is used for finding the majority element in a sequence of elements using linear time and constant space. In its simplest form, the algorithm finds if an element is repeated for more than half of the elements of the sequence, then a second pass through the data verifies if the element is indeed the majority element. If it is found that the element is not the majority, then the action taken can either return the older majority element or return an arbitrary value.

Since majority vote relies on counting if the same element exists multiple times in a sequence, this sequence is composed of integers. But the readings from the IMU are all decimal values, thus making it unlikely that values will be equal, even though being close enough to be considered similar. A modified version of the Majority Vote algorithm would need to be applied. In this version, instead of comparing elements for equality, it compares the difference with a threshold using the equation below:

$$s[0] - s[1] \leq \delta$$

The value of δ needs to be set using trial and error, it will majorly depend on the configuration of the readings (Karimi et al. 2014). In practice, to estimate the value of δ, it is a good idea to collect raw IMU data from the sensors using the given configuration, and then find the standard deviation of the readings. The appropriate value is approximately equal to the standard deviation of the empirical data.

3. APPLYING SAFE FOR IMU SENSOR FUSION READINGS

In this section we discuss how to combine sensor fusion algorithms to get reliable angle readings by fusing accelerometer and gyroscope readings from the IMU. A complementary filter algorithm is applied on five IMU sensors and they are fused using a Majority Vote algorithm. A model of the system was specified using DEVS and simulation, and executed using the Cadmium simulator (Ruiz-Martin and Wainer 2022), after which we ported the models to hardware for testing.

A model of the system was specified using DEVS. This model directly maps to the SAFE framework as shown in Figure 2. From the figure, there are two layers, the sensor layer, and the fusion layer. The sensor layer has five IMU sensors, each of which contains a gyroscope and an accelerometer. The fusion layer has two sensor fusion algorithms which are based on two different configurations, the complementary filter based on the complementary sensor fusion configuration and the majority vote algorithm based on the competitive configuration.

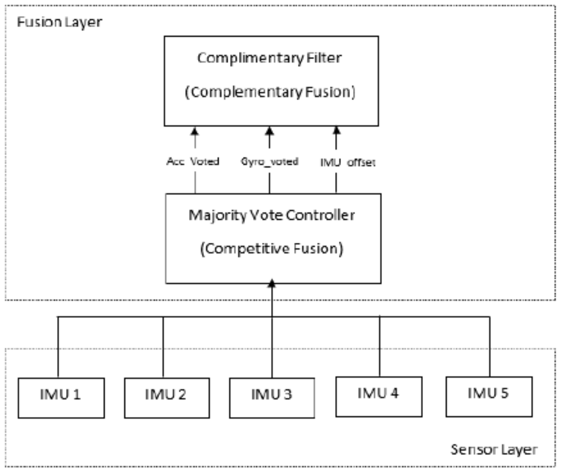

Figure 2: DEVS model structure of the complete system.

The sensor readings are collected in the sensor layer and the outputs are fed to the fusion layer where the fusion algorithms are applied. In this case study, the order of application of the algorithm was of no significance as they both yielded similar results. In our system, a majority vote algorithm is applied to the input of the five IMU sensors to select a reading which is fed to the complementary filter.

3.1. IMU Sensors

As discussed earlier, IMU sensors measure angular velocity, force, and magnetic fields. The DEVS models for the IMU for this case study is specified below (the accelerometer, gyroscope and remaining models in this section have been formally defined using a similar notation).

```
       IMU = <S, X, Y, δint, δext, δcon, λ, ta>
S={active, passive}
X ={acc_raw_values_x, acc_raw_values_y, acc_raw_values_z, gyro_raw_values_x,
          gyro_raw_values_y, gyro_raw_values_z }
Y={IMUData_out_acc, IMUData_out_gyro }

δ int (active) = passive

δ ext (acc_raw_values_x, acc_raw_values_y, acc_raw_values_z, passive) {
   accelerometer[3] = { acc_raw_values_x, acc_raw_values_y, acc_raw_values_z };
   state = active;
}

δ ext (gyro_raw_values_x, gyro_raw_values_y, gyro_raw_values_z, passive) {
   gyroscope [3] = { gyro_raw_values_x, gyro_raw_values_y, gyro_raw_values_z };
   state = active;
}

δ ext() = active

λ (active)accelerometer [3], active {
   if values are valid:
       IMUData_out_acc = accelerometer [3]
   else:
       IMUData_out_acc = None;
       Send data to output port
}

λ (active)gyroscope [3], active {
   if values are valid:
       IMUData_out_gyro = gyroscope [3];
   else:
       IMUData_out_gyro = None;

   Send data to output port
}

ta (passive) = ∞
ta (active) = refresh rate of sensors
```

3.2. Majority Vote Controller

The Majority vote algorithm is used to find the majority element which exists in the given data. In this case, the IMU readings from five sensors are fed to the majority vote algorithm. The flow chart in Figure 3 gives a high-level description of the flow of the code. The Majority vote is applied to X, Y, Z readings from five accelerometers and gyroscopes. The majority vote algorithm used here is a slight variation of the standard algorithm. This change is needed as the standard algorithm uses the comparison of integers, but readings of the IMU sensor are floating-point values, and vary by a small amount due to static noise in the readings.

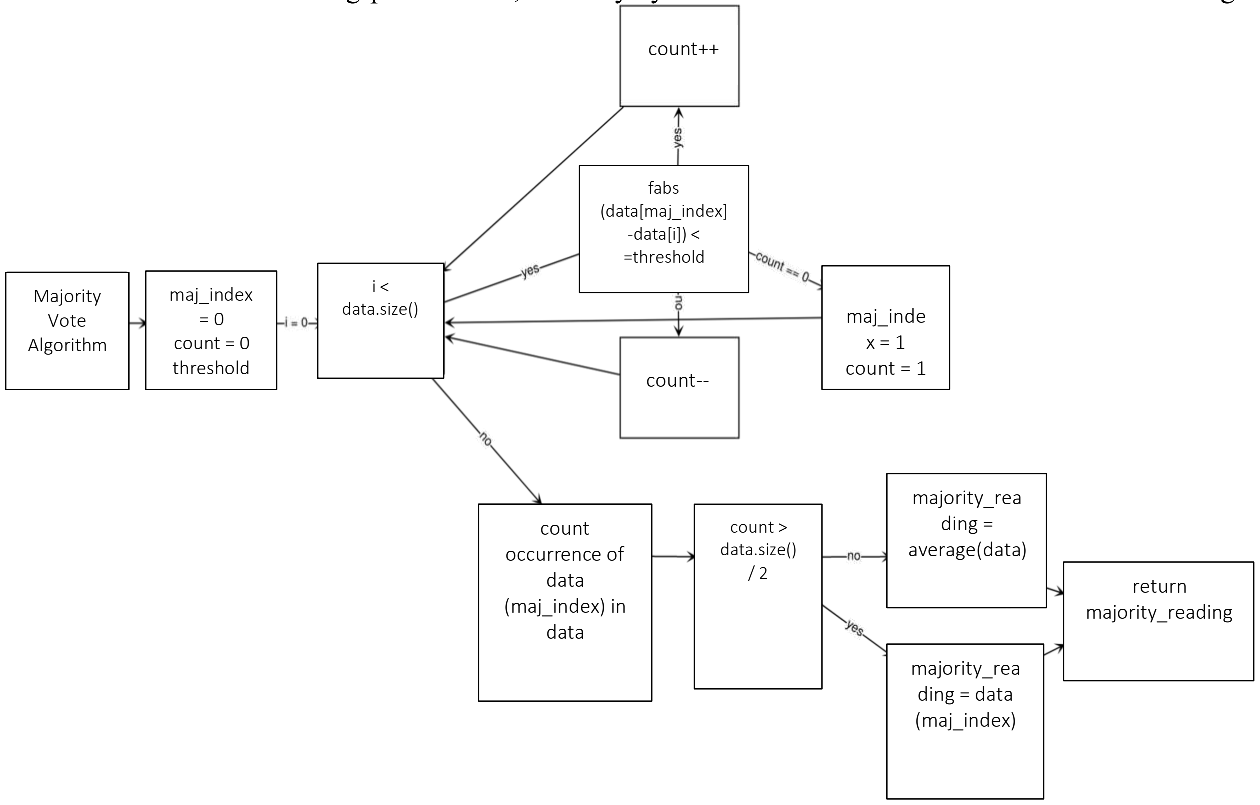
count++
yes
fabs (data[maj_index] -data[i]) < =threshold
yes
count == 0
no
Majority Vote Algorithm
maj_index = 0 count = 0 threshold
i = 0
i < data.size()
maj_inde x = 1 count = 1
count--
no
count occurrence of data (maj_index) in data
count > data.size() / 2
no
yes
majority_rea ding = average(data)
return majority_reading
majority_rea ding = data (maj_index)

Figure 3: Flow chart of the modified majority vote algorithm.

The flow chart shown in Figure 3 describes the modified majority vote algorithm, instead of comparing two elements, that is, the majority_index and the current data it takes a difference of the two elements and checks if it is within a threshold.

3.3. Modeling the Complementary Filter

The flow chart shown in Figure 4 describes the logical flow of the complementary filter. It calculates the angle from raw acceleration readings using the formula described earlier. In the flowchart, it reads the IMU and then passes on the data to the complementary filter block. After this, it stores the current time in the *prev_time* variable then it calculates the time difference, and then applies the complementary filter algorithm to the data

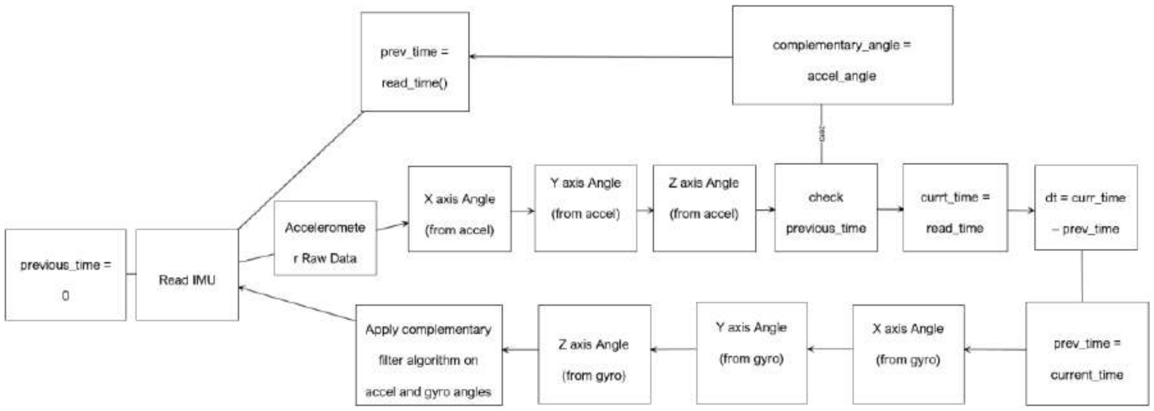

Figure 4: Flow chart of the complementary filter algorithm.

The gyroscope angles depend on the summation of the values over time hence the angles are not included at time $t = 0$. So, we assign the accelerometer angles to the complementary filter output if time $t = 0$. For time $t > 0$, we calculate gyroscope angles as follows:

$$gyro\ angles\ =\ complementary\ angle\ (at\ time\ t-1)\ +gyro\ reading * dt$$

In the next step, the complementary filter formula is used and fused angles are returned upon calculation.

4. HARDWARE IMPLEMENTATION

The model was first extensively simulated in Cadmium, and then was tested on embedded Cadmium running on an STM32 Nucleo-F401RE board. Three MPU6050 sensor modules were used as IMU sensors. The complete hardware is shown in Figure 5.

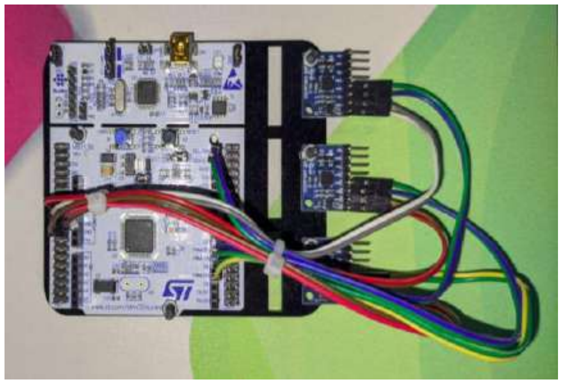

Figure 5: Top View of the actual assembly.

Instead of using five sensors as in the cadmium simulation, this uses three due to the limitation of processing power and a limited number of I2C ports on the STM32 board. So, the code was changed to run only using three sensors. Several simulation scenarios were run to evaluate the combination of the algorithms using SAFE and more tests were carried out on the hardware. The results are presented below.

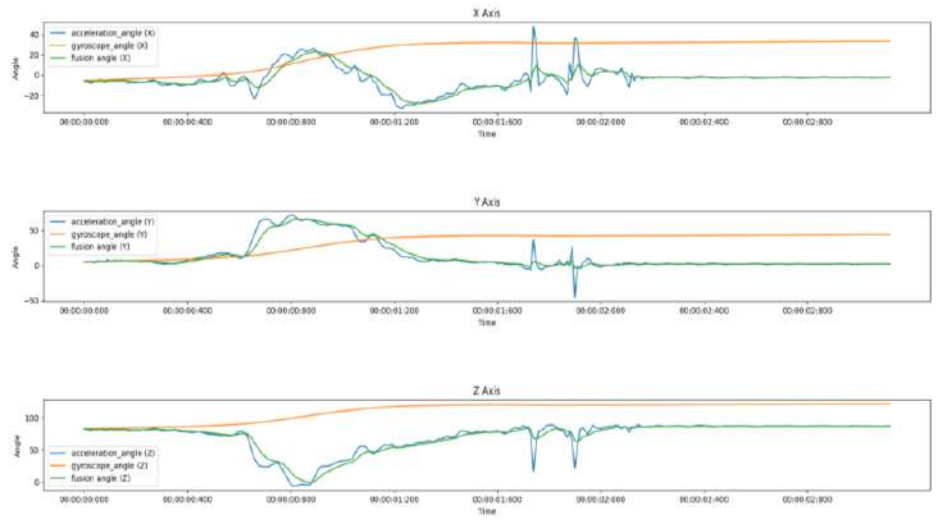

Figure 6: All three axes of Angle vs time plot using complementary filter (Single IMU Sensor).

If we focus on one of the axes, for example, the X-Axis, the gyroscope angle reading has a smooth curve whereas the accelerometer angle has a graph that follows the general trend of the actual angle but with local noise. By applying the complementary filter on the two readings, we get a curve that follows the actual angle at the same time, having minimal local noise.

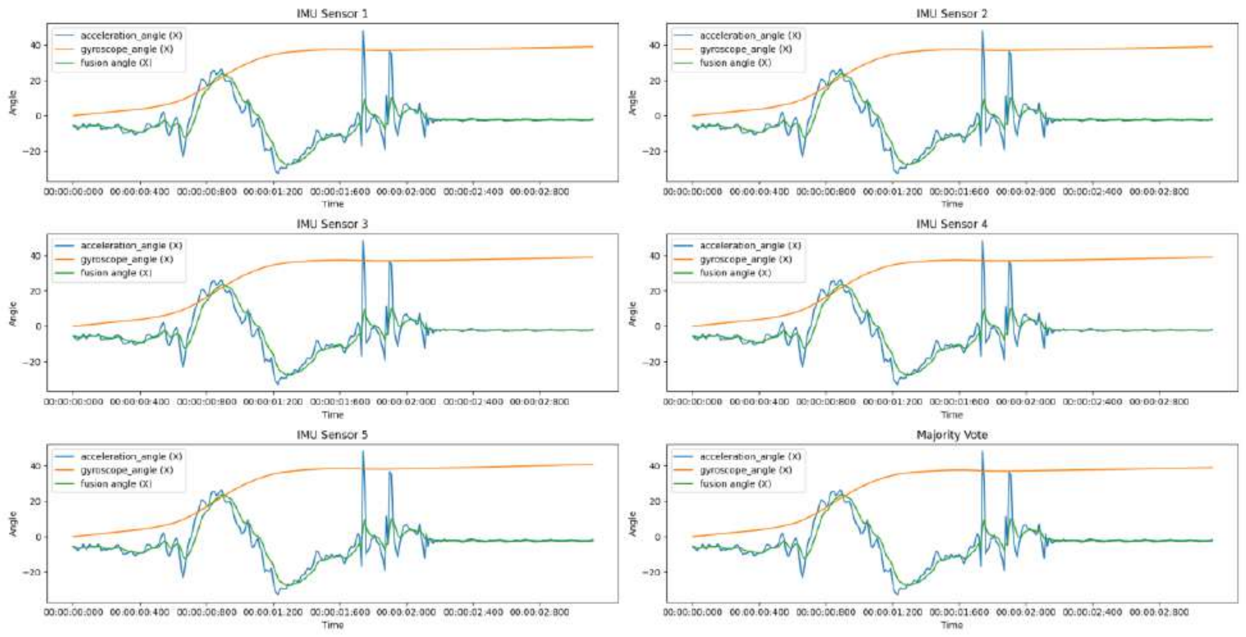

Figure 7: Super plot of Angle vs time of X-axes of five IMU Sensors and combined Majority vote.

Figure 7 shows the X-axis reading plot from each of the five sensors and the plot from the majority vote algorithm. The result obtained looks similar to that obtained without the majority vote algorithm. This is expected because the 5 sensors in simulation have valid readings and the majority vote adds redundancy should one or more of the sensors fail.

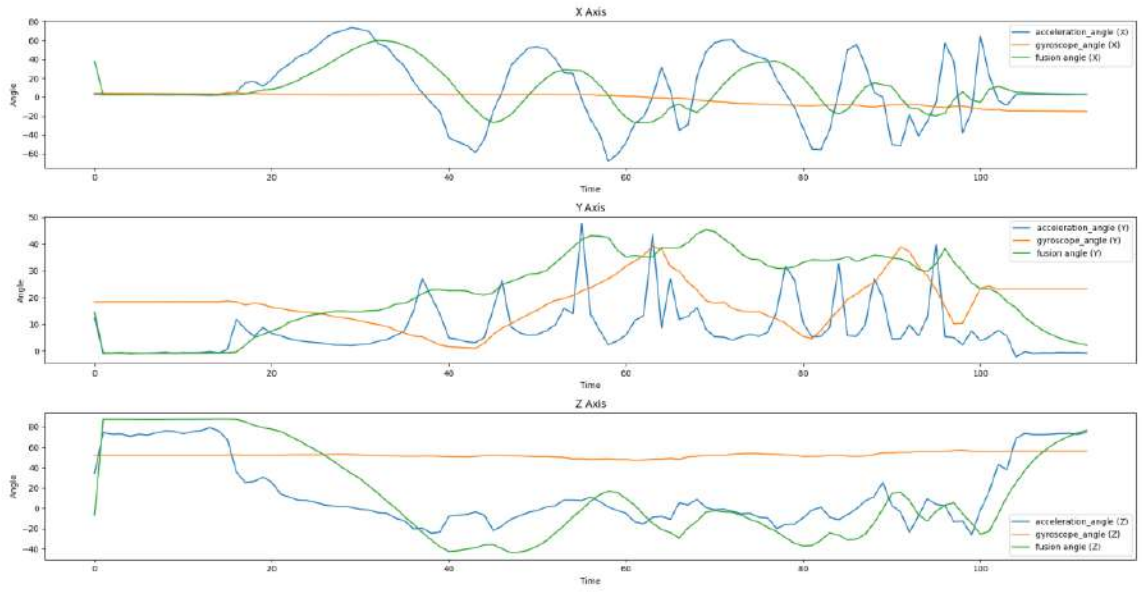

Figure 8: All three axes of Angle vs time plot using complementary filter (Single IMU Sensor).

The plots in Figure 8 show the results from running the model on the actual hardware, that is, on the STM32 Nucleo-F401RE board with three IMU sensors. The plots here show the same characteristics as described earlier. If we focus on one of the axes, let us say the X-Axis, the gyroscope angle reading is smooth whereas the accelerometer angle has a curve that follows the general trend of the actual angle but with a lot of local noise. By applying the complementary filter on the two readings, we get a curve that follows the actual angle at the same time, having minimal local noise. Since this was done on actual hardware, the accuracy was limited by the polling rate of the sensor and the processing capabilities of the used microcontroller.

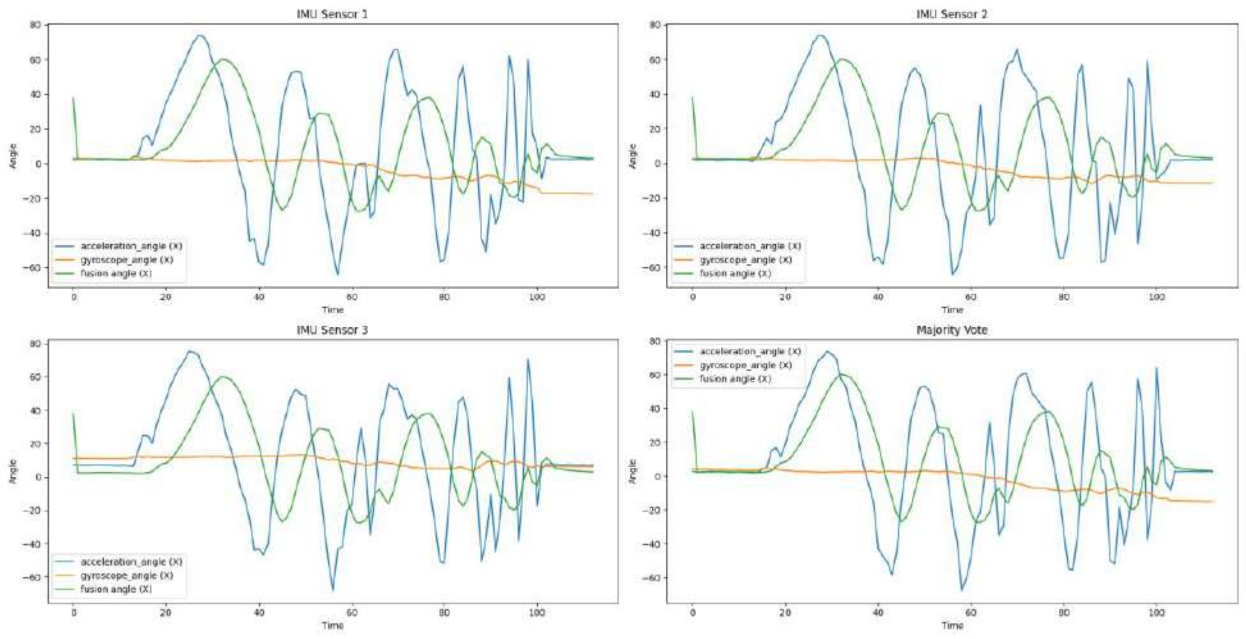

Figure 9: Super plot of Angle vs time of X-axes of three IMU Sensors and combined Majority vote sensor.

Figure 9 shows the X-axis readings from all three sensors, followed by the output from the majority vote algorithm. The gyroscope angles drift as time progresses and the accelerometer angles do follow the angle features, but they have noise and thus readings are not useful as such. So, upon looking at the fusion algorithm output, the output is much better than the other two outputs and does not drift, nor does it have noise inside it.

5. CONCLUSION

We introduced the application of SAFE to improve the reliability of measurements obtained from Inertial Measurement Unit sensors. For the SAFE component, the sensor layer contains groups of sensors that measure external inputs, and it then employs a competitive sensor fusion configuration where each sensor sends its output to the sensor fusion algorithm. The sensor fusion algorithm examines the values to decide which measurements to keep or discard. This algorithm takes raw measurements from the sensor as input, performs principal component analysis calculations, and returns a single output. The failure checking layer checks for irregular readings that occur when most fusion layer input sensor readings are different. We evaluated the functionality of combining sensor fusion algorithms proposed in the SAFE framework by combining two different algorithms: the complementary filter and the majority vote algorithm, which were simulated and later deployed into a hardware prototype. This allowed for testing different aspects of the SAFE framework for improvement and to evaluate the applicability of the proposed framework and showed the potential for use in other fields of application.

REFERENCES

Aggarwal, P. 2010. *MEMS-based Integrated Navigation*. Norwood, MA: Artech House.

Belloli, L., Vicino, D., Ruiz-Martin, C. and Wainer, G. 2019. "Building DEVS Models with the Cadmium Tool". In *Proceedings of 2019 Winter Simulation Conference*, edited by Navonil Mustafee, Ki-Hwan G. Bae, Sanja Lazarova-Molnar, Markus Rabe, C. Szabo, Peter Haas, and Young-Jun Son, 45-59. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Boi-Ukeme, J. and Wainer, G. 2020. "A Framework for the Extension of DEVS with Sensor Fusion Capabilities". In *Proceedings of 2020 Spring Simulation Conference*, 15th-18th, May, Virtual.

Earle, B., Bjornson, K., Ruiz-Martin, C. and Wainer, G. 2020. "Development of a Real-time DEVS Kernel: RT-Cadmium". In *Proceedings of 2020 Spring Simulation Conference*, 15th-18th, May, Virtual.

Gui, P., Tang, L. and Mukhopadhyay, S. 2015. "MEMS Based IMU for Tilting Measurement: Comparison of Complementary and Kalman Filter-based Data Fusion". In *Proceedings of 2015 IEEE 10th conference on Industrial Electronics and Applications* 15th-17th June, Auckland, New Zealand.

Han, S., Meng, Z., Omisore, O., Akinyemi, T. and Yan, Y. 2020. "Random Error Reduction Algorithms for MEMS Inertial Sensor Accuracy Improvement—A Review". *Micromachines 11*(11):1021.

Islam, T., Islam, M.S., Shajid-Ul-Mahmud, M. and Hossam-E-Haider, M. 2017. "Comparison of Complementary and Kalman Filter Based Data Fusion for Attitude Heading Reference System". *AIP Conference Proceedings* 1919(1):020002.

Karimi, A., Zarafshan, F., Al-Haddad, S. and Ramli, A. 2014. "A Novel-input Voting Algorithm for-by-wire Fault-tolerant Systems". *The Scientific World Journal*, article ID 672832.

Kok, M., Hol, J.D. and Schön, T.B., 2017. "Using Inertial Sensors for Position and Orientation Estimation". *ArXiv preprint* https://arxiv.org/abs/1704.06053, accessed September 2023.

Magnusson, N. and Odenman, T. 2012. "Improving Absolute Position Estimates of an Automotive Vehicle Using GPS in Sensor Fusion". Master's Thesis. Chalmers Institute of Technology. https://publications.lib.chalmers.se/records/fulltext/159412.pdf, accessed 27th October 2023.

Mumuni, F. and Mumuni, A. 2021. "Adaptive Kalman filter for MEMS IMU data fusion using enhanced covariance scaling". *Control Theory and Technology 19*(3):365-374.

Narkhede, P., Poddar, S., Walambe, R., Ghinea, G., and Kotecha, K. 2021. "Cascaded Complementary Filter Architecture for Sensor Fusion in Attitude Estimation". *Sensors 21*(6):1937.

Ometov, A., Shubina, V., Klus, L., Skibińska, J., Saafi, S., Pascacio, P., Flueratoru, L., Gaibor, D.Q., Chukhno, N., Chukhno, O. and Ali, A. 2021. "A Survey on Wearable Technology: History, State-of-the-art, and Current Challenges". *Computer Networks* 193:108074

Ruiz-Martin, C and Wainer, G. 2022. "Defining DEVS Models Using the Cadmium Toolkit". In *Proceedings of Winter Simulation Conference*, edited by Ben Feng, Giulia Pedrielli, Yijie Peng, Sara Shashaani, Eunhye Song, Canan Gunes Corlu, Loo Hay

Lee, Ek Peng Chew, Theresa Roeder, and Peter Lendermann, 1356-1370. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Sahawneh, L. and Jarrah, M.A. 2008. "Development and Calibration of Low-cost MEMS IMU for UAV Applications". In *Proceedings of 5th International Symposium on Mechatronics and Its Applications*. Amman, Jordan.

Sukkarieh, S., Nebot, E.M. and Durrant-Whyte, H.F. 1999. "A High Integrity IMU/GPS Navigation Loop for Autonomous Land Vehicle Applications". *IEEE Transactions on Robotics and Automation 15*(3):572-578.

Tseng, S.P., Li, W.L., Sheng, C.Y., Hsu, J.W. and Chen, C.S. 2011. "Motion and Attitude Estimation Using Inertial Measurements with Complementary Filter". In *Proceedings of 8th Asian Control Conference,* Kaohsiung, Taiwan.

Wainer, G. 2017. *Discrete-event Modeling and Simulation: a Practitioner's Approach*. Boca Raton, FL: CRC Press.

AUTHOR BIOGRAPHIES

GABRIEL A. WAINER received the M.Sc. (1993) at the University of Buenos Aires, Argentina, and the Ph.D. (1998, with highest honors) at UBA/Université d'Aix-Marseille III, France. In July 2000, he joined the Department of Systems and Computer Engineering at Carleton University (Ottawa, ON, Canada), where he is now Full Professor. He is one of the founders of SIMUTools, ANNSIM (SCS/IEEE/ACM), the Symposium on Theory of Modeling and Simulation (SCS/ACM/IEEE), and Symposium on Simulation in Architecture and Urban Design - SimAUD (SCS/ACM/IEEE). Prof. Wainer is Editor in Chief of SIMULATION, member of the Editorial Board of Journal of Simulation (Taylor and Francis) IEEE Computing in Science and Engineering, Wireless Networks (Elsevier), Journal of Defense Modeling and Simulation (SCS). He is the head of the Advanced Real-Time Simulation lab, located at Carleton University's Centre for advanced Simulation and Visualization (V-Sim). He is an ACM Distinguished Speaker and a Fellow of SCS. His email address is gwainer@sce.carleton.ca.

JOSEPH BOI-UKEME obtained a Ph.D. in Electrical and Computer Engineering at Carleton University where he researches Cyber-physical Systems. His email address is joseph.boiukeme@carleton.ca.

VEDANT PARANJAPE is a final year Electronics Engineering undergraduate student at VJTI, Mumbai. His research interests are Operating Systems, embedded systems, and Computer Architecture. His email address is vedantparanjape160201@gmail.com.