International Journal of Computing and Artificial Intelligence

E-ISSN: 2707-658X
P-ISSN: 2707-6571
IJCAI 2022; 3(2): 66-70
Received: 08-05-2022
Accepted: 10-06-2022
www.computersciencejournals.com/ijcai

Senboyejo Temitope Anthony
Department of Systems Engineering, University of Lagos, Lagos State, Nigeria

Akanbi Lukman Olawale
Research Scholar, Embedded kits Technologies, Osun State, Nigeria

Lawal Akeem Olaide
Department of Electrical and Electronics Engineering, Osun State Polytechnic Iree, Osun State, Nigeria

Corresponding Author:
Senboyejo Temitope Anthony
Department of Systems Engineering, University of Lagos, Lagos State, Nigeria

Biomedical devices calibration using an improved hybridized algorithm

Senboyejo Temitope Anthony, Akanbi Lukman Olawale and Lawal Akeem Olaide

DOI: https://doi.org/10.33545/27076571.2022.v3.i2a.57

Abstract
IMUs, or inertial measuring units, are used in a variety of medical applications. The measurement accuracy of an IMU, however, might deteriorate with time, necessitating re-calibration. Tedaldi *et al*. offered an IMU calibration approach in their study from 2021 that doesn't call for expensive external precision equipment or laborious methods. As a result, the sensors may be re-calibrated by employees or end users without an advanced understanding of inertial measurement by positioning them in a variety of acceptable but loosely defined orientations. Adaptations for low noise accelerometers, a calibration assistance object, and packet loss correction for wireless calibration are just a few of the enhancements to Tedaldi's technique that we provide in this paper. On our custom-built IMU platform, we used the updated calibration technique, and we checked the consistency of the results across many calibration cycles. We examined how the calibration result accuracy declines when fewer calibration orientations are employed in order to reduce the time required for recalibration. We discovered that N=12 distinct orientations are enough to obtain a very successful calibration, and that adding additional orientations only slightly enhanced the calibration. Comparing this to Tedaldi's suggested range of 37 to 50 orientations, it is a huge improvement. As a result, we were able to shorten the time needed to calibrate a single IMU from about 5 minutes to less than 2 minutes without compromising any significant calibration accuracy.

Keywords: Inertial measurement unit, calibration, wireless

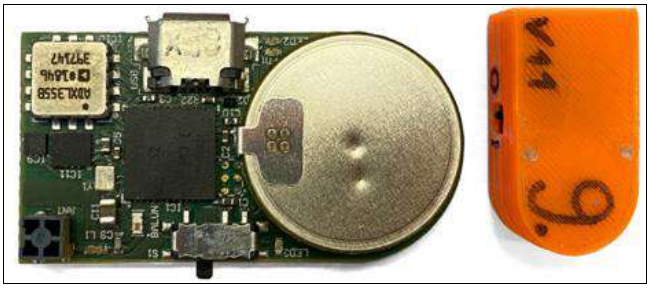
1. Introduction
Inertial measurement units (IMUs) are utilised in a broad range of medical and sporting applications, such as fall detection in senior patients, injury rehabilitation, and training progress tracking [1, 3]. Inertial sensors have also been used for even more difficult tasks, such full-body human motion capture [4, 5]. Correct calibration of each unit is essential in situations when exact measurements and orientation estimations are needed, such as in sports or motion capture. The device will no longer adhere to its declared accuracy limitations when the characteristics of the integrated inertial sensors change over extended periods of time or under variable temperatures, and a re-calibration is required. Users may re-calibrate their IMUs in the field using the approach suggested by Tedaldi *et al*. [6] in 2021 without the need for any specialized and often costly reference tools, such as right angles, turntables, or high precision servo platforms [7]. We encountered several issues when we put this strategy into practice utilizing our proprietary wireless IMU technology, which is explained in Section III. We were able to resolve the problems as indicated in Section IV after carefully examining the algorithm and related processes. The calibration technique described by Tedaldi *et al*. [6] employs unprocessed accelerometer and gyroscope data. The original implementation is in C^{++} [8], but for our tests, we utilised Jianzhu Huai's MATLAB implementation [9], which source code we found to be simpler to use, comprehend, and change.

2. Related work
Ren *et al*. [10] in 2021 put out the idea of spinning the IMU on an inclined plane in order to use the accelerometer to determine the heading. This calibration method does not need any extra equipment. Tedaldi's algorithm's basic idea is similar to the underlying one, but the calibration technique is more involved than just placing the IMU in several random orientations. Although Tedaldi's previous work was noted and remarkable calibration accuracy was claimed by Ren *et al*., they did not compare the accuracy of the two approaches.

A similar strategy to ours was published in 2022 by Peng *et al*. [11], who also employed a 3D-printed icosahedron to position the IMU during calibration. On an integrated microcontroller, they construct a streamlined calibration process based on an iterative weighted Levenberg-Marquardt algorithm. However, they ignore axis misalignment and assume that a pricey multi-axis precision servo stage is available for gyroscope calibration. This renders the technique somewhat ineffective since the accelerometer might be calibrated on the same platform.

3. IMU platform

We created our own wireless sensor device so that we could have an IMU development platform that could be fully customized. The accuracy, size, weight, and power consumption of the sensor data were all optimized for the development platform. Texas Instruments' System-on-Chip (SoC) CC2652R1 was selected as the microcontroller (C). It has a 32-bit ARM Cortex M4F core, an ARM Cortex M0 Bluetooth Low Energy (BLE) transceiver, and a 2.4 GHz processor.



Created a wireless IMU

Fig 1: Left: PCB of the 6th generation of our in-house de

Right: A 3D-printed case that houses the PCB (IMU #9, case version 11), a low-power sensor-controller core for communicating with peripheral components without needing the main core, and the core. A single-chip IMU sensor with a triaxial accelerometer and gyroscope (BMI160 by Bosch), a second high-precision accelerometer (ADXL355BEZ by Analog Devices), and a magnetic sensor are all included in the wireless IMU (MMC3416xPJ by MEMSIC). The magnetic sensor allows for distortion and gyroscope bias drift correction in magnetic, angular rate, and gravity (MARG) sensing applications. For floor-level sensing, a barometric pressure sensor (BMMP388 by Bosch) is incorporated. The embedded 120mAh lithium-ion battery may be charged through wired data transfer with the help of a Serial-to-USB interface (FT230XQ). A four-layer printed circuit board (PCB) with two component sides and measurements of 34.5 mm by 18mm is used to implement component placement and routing. Figure 1 shows the populated PCB and its 3D-printed container. The raw sensor data is fused using a mad wick method that was made available for either IMU or MARG sensing applications [12]. All sensors are sampled at a rate of 100Hz. If desired, sensor fusion may be carried out directly on the microcontroller, which greatly decreases the amount of data that has to be transferred when compared to sending raw sensor data. As a result, many wireless IMUs may operate concurrently at 100Hz frame rates.

4. Calibration improvements

A. Orientation Helper Object

The calibration algorithm requires at least nine different orientations to construct a well-defined optimization problem [6, 13]. In Tedaldi's original work, the IMU is placed in $37 \leq N \leq 50$ distinct static positions, each held for 1-4s. Without support, a cuboid IMU case can only be placed on six faces – or less if some are rounded, as seen in Figure 1. In [6], Tedaldi *et al*. show an IMU with a cable attached resting on the edge of a slab, which is a very unstable position for an IMU weighing only a few grams. The rigid cable can move the lightweight IMU by tiny amounts, preventing the algorithm from identifying static phases. To avoid these problems in our evaluation, we use wireless transmission (see also Section IV-D) and a 3D-printed

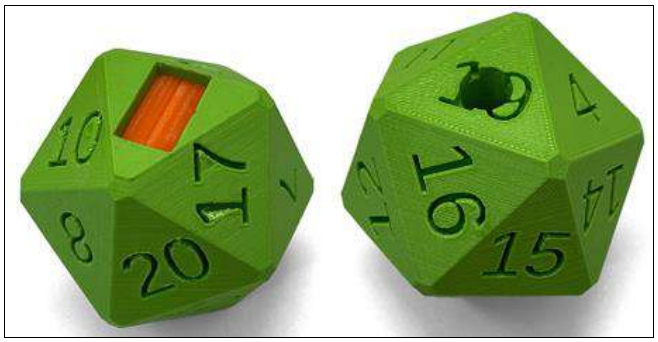


Fig 2: The icosahedral enclosure allows the IMU to rest in

20 different orientations. It can hold the IMU case (orange) and has a whole opposite for pushing the IMU back out. Icosahedral orientation helper object based on Tim Edwards' Opens CAD model [14] as shown in Figure 2. The 3Dprinted regular icosahedron has a distance of 66mm between opposing vertices. The enclosed IMU is press-fit into a slot on one of the triangular faces and can be released by pushing through a hole on the opposing face. The orientation helper object allowed us to place the IMUs to be calibrated in 20 easily reproducible orientations with at least 42 degrees rotation between them and without interfering cables. This enabled a more systematic approach to capturing calibration sequences, as explained in Section V.

B. Improved static phase selection

Running the calibration algorithm [9] on our captured sequences of the low noise accelerometer revealed a weakness of the static phase detection algorithm. This issue was found in both the original C++ code [8] and the MATLAB implementation [9]. The variance of the long static phase in the beginning ςinit is used as a baseline for finding a suitable variance threshold for the short static segments later in the sequence. However, ςinit is so small due to the low noise floor of the ADXL355 that even tiny perturbations in the short sequences are above the maximum threshold of 10ςinit variance. Increasing the maximum variance threshold factor from 10 to 225 solved this problem for our case but revealed another problem: v In Tedaldi's algorithm, the best static phase threshold (an integer multiple k • ςinit) is determined by performing a nonlinear least-squares minimization of the accelerometer cost function L (θacc) for each k and selecting the k with the smallest residual. As stated in [6], this approach does not require parametrization, but favors calibration sequences in which the same orientation is repeated multiple times. Thus, the algorithm would select a k for which the long sequence at the beginning was split into multiple smaller segments of

length>1s because the variance ς (t) was too close to kçinit, crossing the threshold multiple times. This problem can be fixed by rejecting segments where the acceleration vector direction did not change relative to the previously accepted segment, minimizing the number of redundant segments.
However, the selection of the k with the minimum residual was still susceptible to the slightest perturbation of the accelerometer data. When truncating the calibration sequence even slightly, the algorithm would seemingly at random select very different values of k, which led to wildly varying numbers of segments: Fewer segments for smaller k because static phases would be broken into multiple parts, which were then rejected for being shorter than 1s. Thus, we modified the static phase selection to always select the k which produced the largest number of usable static segments, excluding segments that were duplicates or too short. If there are multiple k, we select the one with the lowest residual.

C. Division by zero
The MATLAB implementation [9] contained a division by zero in the function from Omega to Q when the angular rates were [0, 0, and 0] in a single packet. This is a very unlikely event due to the measurement noise, but it occurred in at least one calibration sequence, so we corrected the issue.

D. Wireless Packet Loss Correction
For data transmission between IMU and host PC, we used Bluetooth 5.0 Low Energy (LE) Generic ATT tribute Profile (GATT) notifications since they provide a significant improvement in data transmission speed compared to indication messages [15]. Even though the Bluetooth link layer L2CAP provides acknowledgments and retransmissions, any wireless transmission includes a risk of packet loss. Depending on the application, the loss of a single quaternion may not be critical, but the loss of raw gyroscope data in a calibration sequence will result in integration errors. Loss of accelerometer data is not critical because it is only sampled in static periods.
Thus, we implemented simple and a power-efficient erasure code (EC) for forward error correction of lost gyroscope samples in order to prevent this problem. The EC data Ei is the same size as the raw gyroscope data i (6 bytes) and consists of the XOR of the previous M raw gyroscope values Ei = Gi−1LGi−2L...LGi−M. This allows the receiver to reconstruct arbitrary packet losses in a window of length M when followed by a sequence of M correctly received packets. Computing Ei for a new packet consists of only two XOR operations: One for adding the next Gi−1 and one for removing the oldest element Gi−M from the running XOR sum. Apart from the running sum Ei, only an additional ring buffer for storing Gi...Gi−M+1 is required on the IMU.

4. Evaluation
With each of our four operational IMUs (units #1, #2, #6 and #9), we recorded five calibration sequences with N 37 in order to identify the bare minimum necessary number of static positions N for a calibration precise enough. We captured the packet index, ADXL355 acceleration, BMI160 acceleration, and BMI160 gyroscope measurements for each sequence. The first static phase lasted for 40s, and thereafter there were static intervals of around 3s. We arranged the icosahedron on all 20 faces of the first 20 orientations, with increasing numbers pointing upward. After the increasing sequence, we randomly selected faces on which to position the icosahedron until at least 37 postures were captured. We deleted all IMU #9 results for N>34 as there were only 34 valid postures in one calibration session. Using our modified MATLAB code, we fully calibrated each IMU for each sequence that was taken with a maximum of N, yielding 4 times 5 sets of the 18 calibration parameters (= [acc, gyro], as reported in [6], for each IMU. We determined the average parameters for the five sets for each IMU, and these values served as the ‘reference' calibration coefficients for that IMU. After that, we gradually cut short each sequence, lowering the effective N = Neff. We ran a second calibration run using Neff segments for each trimmed sequence and then calculated the mean absolute difference for each group of coefficients to compare the resultant Neff to mean. This eliminates the necessity for an exact "gold standard" reference of the calibration parameters and enables evaluation of how the calibration quality degrades with decreasing N, compared to the "complete" calibration suggested by Tedaldi.
By simply mounting the IMUs in four different horizontal positions on the five orthogonal sides of their casing, we were able to record an extra set of calibration runs without the need of an icosahedron. These sequences allowed for a successful calibration, but since the calibration error for these runs was substantially higher than the mean, we did not include the data.

5. Results
The evaluation's findings are shown in Figures 3, 4, and 5. As the conclusion is the same for the BMI160 accelerometer, we only show the calibration run results using the ADXL355 values. The accelerometer-related.

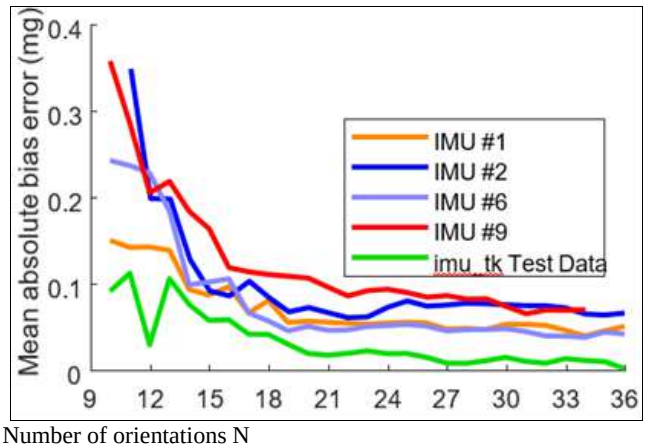
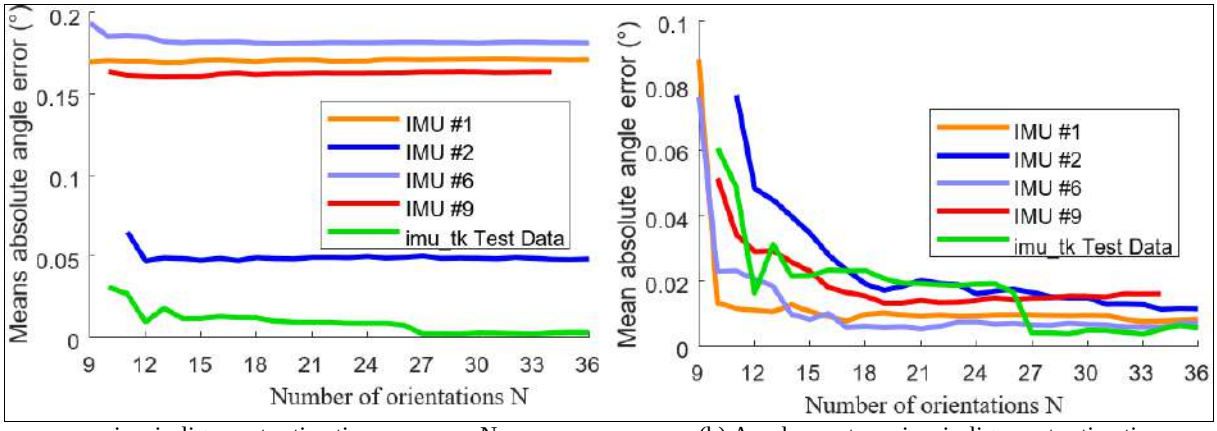
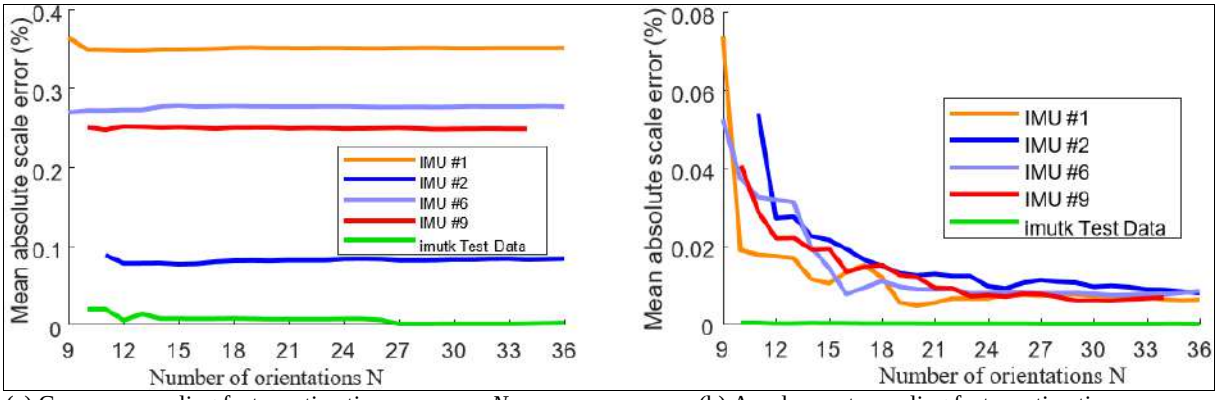


Fig 3: Mean absolute difference of accelerometer bias estimation to θ mean for varying N. Mean of 5 calibration runs for each IMU except imu tk Test Data (only 1 calibration run). Units are 1mg = 9.807 × 10−3 m/s2



(a) Gyroscope axis misalignment estimation error over N (b) Accelerometer axis misalignment estimation error over N

Fig 4: Axis misalignment estimation errors (including non-orthogonality) for different N, when compared to θ mean. Typical mean absolute misalignment angles of θ mean for our IMUs were in the range between 0.4 and 0.6 degrees



(a) Gyroscope scaling factor estimation error over *N* (b) Accelerometer scaling factor estimation error over *N*

Fig 5: Scaling factor estimation errors for different numbers of orientations N, when compared to θ mean. Values are given in percent of the measured quantity (angular rate and acceleration respectively).

Calibration, with greater N, we see a modest decrease in inaccuracy. The errors near N=12 orientations, though, are already less than 0.1% of the mean values for scaling and misalignment. The calibration error for bias is already very close to the ADXL355's noise floor (200 g at 100 Hz). Therefore, it is debatable whether the extra effort required for three orientations is necessary. The outcomes are significantly more evident for the gyroscopes: When more positions are taken into account after sufficient orientations have been collected to calculate a successful calibration, the accuracy barely improves at all. Because the mean is calculated from just one calibration run rather than five, the calibration sequence is essentially compared to itself, which causes the results for imu tk Test Data to be unreasonably low. Despite this, the major finding for N also applies to this IMU.

Conclusions and outlook

In this study, we described enhancements to a calibration method for wireless IMUs that doesn't need for pricey calibration apparatus. By fixing mistakes that only sometimes occurred and increasing the static phase identification while employing low-noise accelerometers, the processes described in the first paper by Tedaldi *et al.* [6] and its MATLAB implementation [9] were enhanced. Our findings suggest that a dodecahedron would probably be enough, but a low-priced, low-accuracy 3D-printed icosahedron functioned as the sole supplementary, optional calibration equipment and allowed for simple placement of the IMU in 20 different attitudes. The calibration process' evaluation revealed that positioning According to Tedaldi *et al*.'s advice [6], the IMU in N=12 distinct positions already produces a very small error for the accelerometer N and a minimal error for the gyroscope calibration. The methodology given by Tedaldi *et al*. will be implemented in the next step directly on the CC2652R1 SoC of our internal wireless IMU, using a strategy similar to Peng *et al* [11].'S method without compromising calibration for the gyroscope and axis misalignment. It will be necessary to fully redesign the algorithm's structure for this purpose, optimising for computational complexity, programme size, and RAM utilisation.

References

1. Pre-impact fall detection X. Hu X Qu. Bio Medical Engineering on Line. 2022 June 15(1). [Online]. Accessible through this Link:10.1186/s12938-016-0194-x
2. Reviews on different inertial measurement unit (imu) sensor applications, International Journal of Signal Processing Systems. 2020;1(2):256-262.
3. Ahmad NRAR, Ghazilla NM, Khairi V. Kasi. Development of an upper limb exoskeleton for rehabilitation using input from EMG and IMU sensor," Procedia Computer Science. 2021;76;53-59.
4. Ganesan Y, Gobee S, Durairajah V. [Online]. The following link is available: 10.1016/j.procs.2021.12.275
5. Real-time full-body motion capture from video and

imus," in 2017 International Conference on 3D Vision (3DV); c2017. p. 449-457.
6. Malleson C, Gilbert AM, Trumble J, Collomosse A, Hilton M, Volino A. new biomechanical model-aided imu/uwb fusion for magnetometer-free lower body motion capture is described in S. Zihajehzadeh and E. J. Park's paper in IEEE Transactions on Systems, Man, and Cybernetics: Systems. 2017;47(6)927-938.
7. Tedaldi D, Pretto A, Menegatti EA. robust and simple to use approach for imu calibration without external equipments, IEEE International Conference on Robotics and Automation (ICRA); c2021. p. 3042-3049.
8. Marquez-Viloria DL, Castano-Londono J. Botero-Valencia L. Morantes-Guzman Measurement, A low-cost platform based on a robotic arm for parameters estimation of inertial measurement units." [Online]. The URL for this publication is. 2017;110:257-262. https://www.sciencedirect.com/science/article/pii/S0263224117304360
9. Imu Tk C++ Implementation, A. Pretto, 2021. Available online at: bitbucket.org/alberto pretto/imu tk
10. Imu Tk Matlab Implementation by J Huai, 2017. https://github.com/JzHuai0108/imu tk matlab is [available online].
11. A unique self-calibration approach for mimu, IEEE Sensors Journal. 2021;15(10):5416-5422.
12. Ren C, Liu Q, Fu T. Design of an embedded icosahedron mechatronics for robust iterative imu calibration, IEEE/ASME Transactions on Mechatronics. 2022;27(3)1467-1477.
13. An effective orientation filter for inertial and inertial/magnetic sensor arrays is described by S. Madgwick *et al*. in Report x-io and University of Bristol (UK). 2020;25:113-118.
14. A novel multi-position calibration approach for MEMS inertial navigation systems," Measurement Science and Technology. 2007;18(7):1897-1907.
15. Syed ZF, Aggarwal PC, Goodall X, Niu N, El Sheimy. [Online]. Available:\shttps://doi.org/10.1088/0957-0233/18/7/016
    Opens cad polyhedral dice, T. Edwards, 2021. [Online]. The URL for this item is: https://www.thingiverse.com/thing:1043661
16. Bluetooth SIG, Generic Attribute Profile (GATT), 1 2022, GATT.TS.p21.