New Artificial Intelligence Approach to Inclination Measurement Based on MEMS Accelerometer

Minh Long Hoang, *Member, IEEE*, and Antonio Pietrosanto, *Senior Member, IEEE*

***Abstract*—The article presents a research of angular orientation based on a microelectromechanical system (MEMS) accelerometer by using machine learning (ML) and deep learning (DL) model with architectures of deep neural networks (DNNs). In the industrial environment, artificial intelligence (AI) plays a crucial role in automation which is a potential solution for better performance of inclinometer. This article was carried out to apply this intelligent model on the inertial measurement unit to accomplish the angular position. The experiment shows that the ML model correctly learns the relationship between acceleration and tracking angles via polynomial regression with an *R*-square of 0.98. The employed DL model with four hidden layers of ten neurons achieves an accuracy of 99.99 % and almost a nonerror performance. The acceleration acquisitions were obtained from MEMS accelerometer LSM9DS1 at a frequency of 50 Hz via microcontroller STM32F401RE. The ML and DNN models were designed based on the platform Tensorflow with high processing accuracy. The Pan-Tilt Unit was used as the angle reference for static and dynamic tests. The traditional technique is used for comparison as well as verification of the proposed models. The DL model has better precision over the ML model due to its high structure level with updating weight and error optimization from the neural network structure. Meanwhile, ML shows more stable results in dynamic circumstances.**

***Impact Statement:*— Nowadays, the high accuracy of inclinations (roll and pitch) has become significantly important in multiple applications, especially in industry. The traditional method is to calculate inclination by goniometric function on accelerations with the low-pass filter's support. This method still cannot accomplish the optimized efficiency with overshoot signal or unnecessary variation. This article applies the machine learning and deep learning to overcome this limit with advanced improvement: higher accuracy and better tracking performance stability without other sensors' support. This achievement enhances the safety of the industrial vehicle, which requires high precision of tilt measurement.**

***Index Terms*—Accelerometer, artificial intelligence (AI), deep learning (DL), inertial measurement unit (IMU), machine learning (ML), microelectromechanical system (MEMS).**

I. Introduction

INERTIAL measurement unit (IMU) for the angular position has been widely used for numerous applications such as robotic [1], [2], navigation system [3], [4], and industrial vehicle [5], [6]. Microelectromechanical system (MEMS) [7], [9] accelerometer [8], [10] is the critical factor in orientation tracking, especially for determining inclination of roll and pitch as described in [11]. The roll angle is the rotation around the *x*-axis of the vehicle and the pitch angle represents the rotation around the *y*-axis as demonstrated in Fig. 1. In recent years, the development of inclinometer raised the demand for high precise inclination. However, there is a limited number of research papers, which measure the inclination based on accelerometer solely. Usually, the low-pass (LP) filter [5], [12] is used as a good tool to enhance the accuracy of oriental measurement by removing the external noise. Nevertheless, the LP filter still cannot bring the best measurement to the user since it has the fault tolerance which can be found in any inclinometer specification. Sensor fusion [14] is able to improve the acquired result, but this method requires other sensors' supports, which increases the price as well as the complexity of the system. Therefore, it is necessary to seek out a new approach for enhancement of inclinometer performance. Today, the development of artificial intelligence (AI) in the Industry 4.0 [15], [16] helps manufacturers to drive efficiency, improve quality, and better manage supply chains with mimic "cognitive" functions of the machine that humans associate with the human mind for "learning" and "problem-solving." Machine learning (ML) and deep learning (DL) algorithms provide novel tools to cope with large datasets, learning intricate nonlinear patterns from input information [17], [18]. ML supports predictions using enormous data and, at the same time, makes it imperative to replicate on manufacturing practices based on pattern recognition from the data [16], [19], [20]. DL [21]–[25] uses a layered structure of algorithms called an "artificial neural network" to learn and make intelligent decisions independently. It requires lots of training to get the learning processes correct. However, once it works as an intentional process, it becomes incredibly flexible in predicting data variation like the backbone of real AI. A deep neural network (DNN) is a type of DL with multiple layers between the input and output layers. Each mathematical manipulation occurs between the layer and is, hence, called the name "deep" networks.

This article describes an ML model that learns the relationship between acceleration and an equivalent angle to detect the slope of the object. Accelerations on *X*-axis, *Y*-axis, and *Z*-axis are called *X*acc, *Y*acc, and *Z*acc, which are acquired from MEMS accelerometer LSM9DS1 as an input of the designed model. DNN is made up of three simultaneous tri-axial accelerations, four hidden layers of ten neurons, and two outputs of roll and pitch. Their operating difference is demonstrated via static test

Manuscript received February 8, 2021; revised April 3, 2021, June 15, 2021, and July 5, 2021; accepted August 11, 2021. Date of publication August 18, 2021; date of current version January 20, 2022. This paper was recommended for publication by Associate Editor H. Luo upon evaluation of the reviewers' comments. *(Corresponding author: Minh Long Hoang.)*

The authors are with the University of Salerno, 84084 Fisciano, Italy (e-mail: hoangminhlong94@gmail.com; apietrosanto@unisa.it).

Digital Object Identifier 10.1109/TAI.2021.3105494

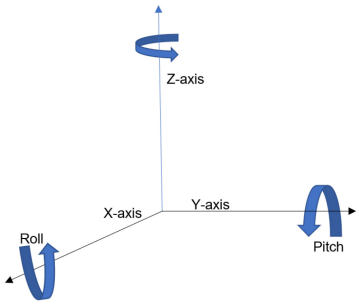
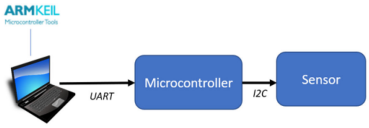


Fig. 1. Orientation visualization.

- Implementation Process



- Operating Process

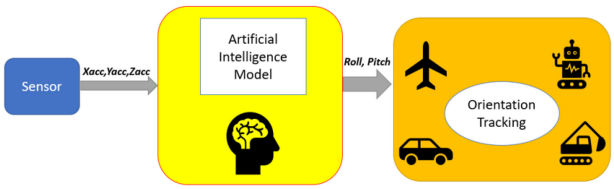


Fig. 2. AI process in orientation tracking.

and dynamic test. The article aims to present a new approach of AI to inclination measurement as illustrated in Fig. 2. Two proposed models were compared with the traditional method of inclination measurement. The accelerations pass through an LP filter to remove the high-frequency noise; then the orientation formula converts these data to tilt information. In this work, the traditional method is called LP filter which demonstrates the pros and cons of the ML and DL models in spatial orientation precisely.

The article is organized as follows. The first part is a brief description of AI on angular position, and then the model and detail architecture of ML and DNN are presented. In the next section, the analysis of the experiment results in both models is deeply described. The dynamic test is carried out repetitively to validate the precision of the algorithms. Finally, the conclusion summarizes the research achievement and future work of the project.

II. Related Work

Nowadays, the ML and DL have been applied widely in human activity recognition (HAR). When the application of ML or DL in accelerometer is mentioned, the involved research are usually about activity classification [43]–[45]. Commonly, the sensing method classifies HAR system into two types: vision-based and acceleration-based methods. Vision-based method usually uses one or more cameras to collect data, while acceleration-based method asks the users to wear several accelerometers for data collecting [46]. A combination between the ML method and domain knowledge successfully proceeds the classification from the collected data of several sensors on the subjects' chest and feet [37]. Another research [38] uses K-nearest neighbors algorithm and neural network algorithm to classify activities of people. The accelerometer is put on the user's waist to detect different activities of people, such as run, walk, fall, etc. In addition, the comparison between ML and DL has been carried out in this field popularly. The paper [40] uses accelerometer data to compare the ML classifiers and DL classifiers in HAR from accelerometer data. The results shows that DL exhibits a higher performance than traditional ML classifiers with no hand-crafted feature selection. Another article [41] works on the comparison of ML and DL techniques for activity recognition using mobile devices. The authors find out that DL outperforms traditional ML techniques in many different domains where feature engineering is complex. However, ML is highly competitive with an advanced feature engineering since DL must include the overload of finding a complex nonlinear transformation of raw features adapted to the problem being solved. Moreover, DL might be the most viable approach with complex transformation, but its computational and data requirements are always higher [42]. Many approaches have been implemented in literature with the aim of recognizing human activity by accelerometer, but there are limited articles that deeply analyzed the inclination/orientation tracking based on ML and DL.

In [36], this work adds one more accelerometer to correct the inclination sensor's errors. The acceleration and inclination sensors are obtained by the ARDUINO sd module and then analyzed by WEKA (ML software). Although this combination can minimize the inclination sensor error, the additional accelerometer becomes a significant disadvantage because of the rise of the sensor cost and system complexity. Other research like fingerprinting methods for spatial characterization of the environments monitored [26] vehicle attitude recognition method based on convolution neural network [27]. These works improve the position estimation results or determine what kind of action the car is doing using ML and DL. However, the detailed precision of roll and pitch was not investigated deeply, and additional multiple sensors like gyroscope and magnetometer with complicated algorithms are required.

Practically, the inclination is calculated by using the trigonometric function like discussed in [48]. By using the sine and cosine formula, the acceleration can be converted to the orientation angle. The accelerations of the X-axis and Y-axis are used to calculate the roll and pitch, respectively, as described in [5] and [12]. Although this traditional slope detection method can provide the tilt value, the accuracy is still limited, which is demanded to improve. The additional sensor is not an appropriate solution in economic concerns since many companies desire to use the only accelerometer for inclinometer.

Therefore, this article deeply works on the learning relationship between acceleration and tracking angles based on the ML polynomial regression (PR) and DNN algorithm. The acceleration data are used for regression algorithms and not in classification; so the designed models must accomplish high level of accuracy to guarantee any single output value. The described methods are direct, less complicated, but highly

Fig. 3. Low-pass filter model.

effective. The polynomial model is proposed to train the orientation datasets to fit the complicated and nonlinear functions in the inclination. Meanwhile, the DL model shows a new approach to solve the rapport between three-axis acceleration *X*cc, *Y*acc, and *Z*acc with the angular position. This work exploits the structure and algorithm functions step by step to demonstrate the advance of each trained model and the different propagation as well as performance between them. The ML model is capable of acquiring higher accuracy than the traditional technique after a serial test of inclination. The DL model shows the high potential of tracking the tilt angle with less variation, and the achieved values are almost the same as reference angles. The ML performance is more stable than the DL model since the ML tracking capability always remains the same, while DL may sometimes conduct significant errors during dynamic performance.

III. Traditional Inclination Measurement

To measure roll and pitch, two fundamental formulas are used for oriental inclination. The *X*-axis acceleration (*X*acc) is proportional to the sine of the angle of inclination (roll). The *Y*-axis acceleration (*Y*acc) is calculated by the cosine of the inclination (Pitch) as follows:

$$\text{Roll} = \arcsin(X\text{acc}) \tag{1}$$

$$\text{Pitch} = \arccos(Y\text{acc}) - 90. \tag{2}$$

A single-pole infinite impulse response (IIR) filter is the affective filter for a digital accelerometer. This filter is a useful tool to remove high-frequency noise by allowing only low-frequency signals to pass through.

1) Filter equation in z-transform

$$H(z) = \frac{bz}{z-a} = \frac{b}{1-az^{-1}}. \tag{3}$$

2) Discrete-time signal of the input–output form

$$y[n] = \alpha x[n] + by[n-1] \tag{4}$$

$$\alpha = e^{-2\pi\tau} \tag{5}$$

where τ is the ratio, calculated by the desired corner frequency f_c and sample frequency f_s of the filter as described in [28]

$$\tau = \frac{f_c}{f_s} \tag{6}$$

where $x[n]$ is input and $y[n]$ is the output signal. $\alpha + b = 1$ because of unity gain at dc in a filter. The acceleration data Xacc and Yacc enter the LP filter and go out as X_LP and Y_LP as shown in Fig. 3.

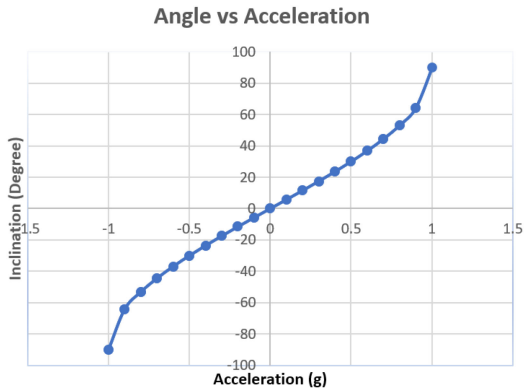


Fig. 4. Angle versus acceleration.

IV. AI on Angular Position

Roll and pitch are measured based on the variation of accelerations from Xacc, Yacc, and Zacc. There are various ways of attitude calculation, but AI brings out the angle based on the training data. In this work, ML and DL are applied separately, and then their prediction accuracy will be compared with each other to demonstrate the pros and cons of each training model. From the learning model, the ML understands the relationship between acceleration and angular position and then predicts the result each time the input data come into the model. Similarly, DL generates the decision model, which is optimized by weight update to accomplish minimized error via hidden layers. Fundamentally, the ML model becomes progressively better at whatever their function is, depending on guidance. In the case of returning an inaccurate prediction, ML requires adjustments from the human. With a DL model, an algorithm can determine if a prediction is accurate or not through its neural network. Therefore, ML is trained by fewer complex data: angle and acceleration, as shown in Fig. 4. Therefore, roll and pitch will be detected following this ideal sinusoidal curve. Meanwhile, the DL learns from three inputs *X*acc, *Y*acc, and *Z*acc to decide the proper angles. In this way, the DL is capable of decreasing the impact of external noise on the sensor. The precise rotation of the Pan-Tilt Unit obtains the training data of the DL model. (PTU-C46) in the range of (–90° to 90°).

V. Machine Learning Positioning

A. Selection of Regression Model

First, ML trains the computer from studying data and statistics of the relationship between acceleration and roll and pitch. In this case, both roll and pitch data need to be trained with an array of acceleration from –1 to 1 to collect their corresponding angles of $-90°$ to 90°. Coherently, all the axial points of the sensor are converted to the inclination thanks to the predicting model.

Regression algorithms [39] are supervised learning for the prediction of actual values by learning the relationship between inputs and outputs. There are four potential regression types, which can be utilized for orientation tracking: linear regression (LR), PR, support vector regression (SVR), random forest regression (RFR). LR generates a straight line for the relationship

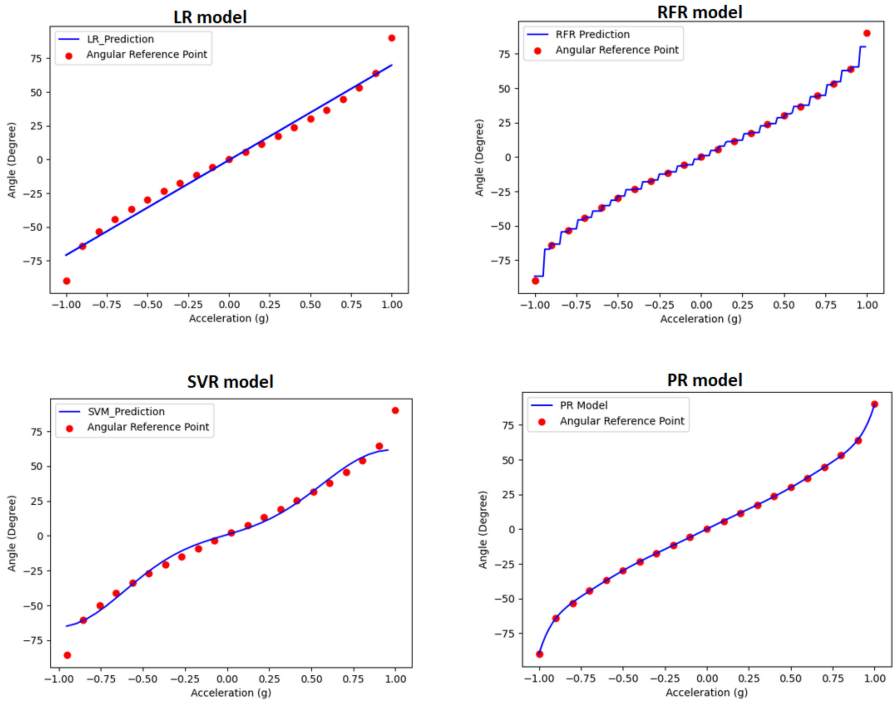


Fig. 5. Regression models in orientation.

TABLE I
ML MODEL VALIDATION

Metric	LR	PR	SVR	RFR
MAE	3.5	0.12	0.13	1.59
R2	0.61	0.98	0.92	0.88
MSE	7.88	0.01	0.02	2.76

between data points. About the PR algorithm, a polynomial line is trained to be fit data-points by ML, which describes the relationship via PR. SVR is capable of transforming linearly inseparable data to separable data by adding more dimensions to it. The RFR idea is to combine multiple decision trees to determine the final output rather than relying on individual decision trees.

As visualized in Fig. 5, LR is not so fit for this case since the relationship between acceleration and inclined angle follows the curve instead of linear. SVR with the kernel of radial basis function creates complex regions within the feature space in higher dimensional space where each data in the training set represents its own dimension. This method can generate curving relationship, but its predictions still have considered distance with the reference value, especially at minimum and maximum points. RFR calculates many different averages from its decision tree predictions in each interval, resulting in multiple steps in each interval. The visualized line shows the increment of angle, respected to acceleration in step by step that does not ideally match the angular curve relationship in Fig. 4. Meanwhile, the PR is more direct and optimally matches the curve relationship of the concerned data as shown in Fig. 7. In addition, Table I reports the evaluation metrics from the mentioned regression

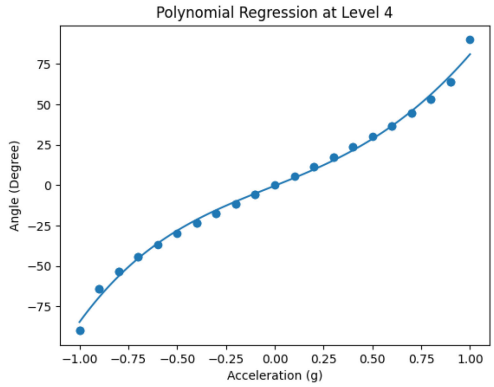


Fig. 6. Polynomial regression at level 4.

models: mean absolute error (MAE), $R2$ score, and mean square error (MSE). As the evaluated data, the PR model accomplishes the less error and the highest $R2$ score, compared with other models. Hence, the PR fits the best of the nonlinear relationship between data points from two variables in this case.

B. Polynomial Regression Models

The ML model contains two main stages: model training and test process.

1) In regression training, the ML model has to determine the connection between input and output of the provided data. Like human study, it learns the mapping between the features of a dataset row and the target value. Depending on the selected algorithm, the ML model will develop its knowledge to fit the nonlinear curve in this case. For the PR, ML determines the relationship between a

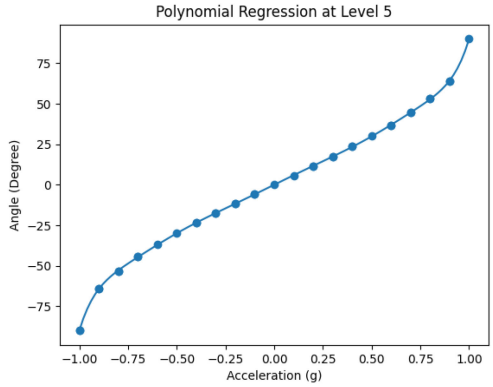


Fig. 7. Polynomial regression at level 5.

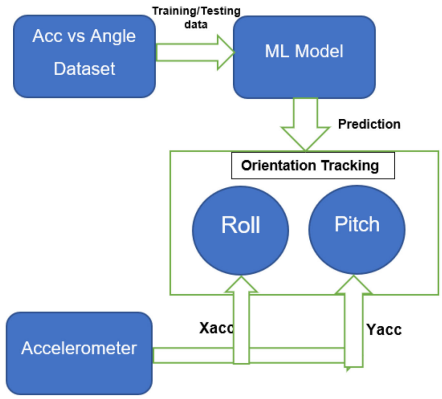


Fig. 8. Orientation tracking by ML model.

dependent(y) and independent variable(x) with nth degree polynomial as described in the following equation:

$$y = b_0 + b_1 * x + b_2 * x^2 + b_3 * x^3 + \cdots + b_n * x^n \tag{7}$$

where $b_0, \ldots, b_n$ are constants, whose model trains to find out their fittest value for input and output data.

2) ML test is the process that evaluates the efficiency of ML performance after the training period. The test data must not be included in the training process for reliable validation. It is usually from 10% to 20% of the provided data. This stage is similar to giving an exam to the students for judging them by score, up to the number of error.

The acceleration data is the acquisition input of Xacc or Yacc, and the ML will provide the orientation angle for the user. The ML polynomial model is trained as the following steps.

Step 1) Implement the dataset. The total of data is imported to the ML model.

Step 2) Split into train/test. The training set includes 85% of the original data which are the acceleration and corresponding angle. The testing set covers all the remaining data.

Step 3) Fit the dataset. Acceleration and the corresponding angle are displayed on the relationship chart to guarantee the precision of two variables as displayed in Fig. 5.

Step 4) Evaluate the model. R-square ($R2$) is a method for measuring the relationship between acceleration and angle in the scale from 0 to 1. In this case, 0 means no relationship, and 1 means related. Both the training set and testing test are evaluated. This model score achieves approximately 0.98 which means 98% related between value of acceleration and inclination in the PR. Thus, this model has a high accuracy in predicting future values based on the PR.

Step 5) Process the model. The acceleration acquisition is carried out, and the ML model will accomplish the roll and pitch by the learned model. It is necessary to select the appropriate polynomial degree of the regression model. A too low level cannot provide the best fit in data relationship, while a too high level consumes more time of launching model due to more computational complexity. As illustrated in Fig. 6, the PR at level 4 still does not fit all the concerned points while the curve fits perfectly the data in Fig. 7. ML model learns one input as acceleration and gives the output as roll and pitch. Fig. 8 describes the process of angular prediction by the ML model. The ML model learns from the datasets to acknowledge the link between two parameters to predict the angles from the input of acceleration on the X-axis and Y-axis.

VI. Deep Learning Position

DL imitates the working of the human brain in processing data and creating the pattern for decision making. Different from the ML model, the data for the training model of DL has three inputs Xacc, Yacc, Zacc, and two outputs of inclination at the same time as Fig. 8.

Step 1) Load the data. The first step defines the functions and classes in the orientation project. There are three input variables and two output variables. The model learns to map rows of input variables (Xacc, Yacc, Zacc) to output variables roll and pitch.

Step 2) Define the DL architecture and compile the model. This neural network contains three inputs, four hidden layers with ten neurons for each, and two outputs. If the number of hidden layers and neurons is not enough, the result will have large errors. In contrast, a very large number of hidden layers require a more extended time of the operational process and weight update that slows down the neural system. The rectifier is an activation function. A unit employing the rectifier is also called a rectified linear unit (ReLU). The activation function is parametric ReLUs (PReLUs) which makes the coefficient of leakage into a parameter. This feature is learned along with the other neural-network parameters

$$f(x) = \begin{cases} x, & \text{if} x > 0 \\ a \cdot x, & \text{otherwise} \end{cases} \tag{8}$$

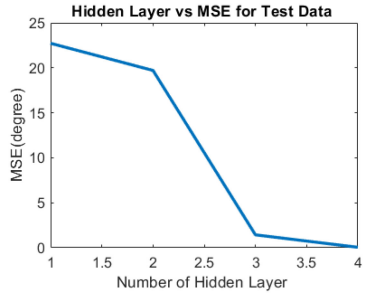


Fig. 9. Hidden layer and error graph.

where a is preset to 1 to acquire the real data of orientation. The loss functions are calculated as mean squared error, demonstrating the error of the model. The optimizer is selected as the efficient stochastic gradient descent (SGD) algorithm "adam" (adaptive movement estimation algorithm) [46]. The initial values of the moving averages and beta1 and beta2 values are 0.9 and 0.95, which are close to 1 and result in a bias of moment estimates toward zero. This is a popular version of gradient descent because it automatically tunes itself and gives good results in a wide range of problems.

In the DL construction, the number of neurons should be suitable for the model operation. Underfitting occurs when the neuron number is too few. In contrast case, an excessive number of numbers causes overfitting. Furthermore, an inordinately large number of neurons in the hidden layers increases the network training time. In many cases, the amount of training time can increase to the point that it is impossible to train the neural network adequately. Thus, the compromise of neurons number must be compromised. In this case, ten neurons per each hidden layer is selected. About the number of hidden layers [47], there are two fundamental rules to start the selection idea: There are many rule-of-thumb methods for determining the correct number of neurons to use in the hidden layers, such as the following.

1) The number of hidden neurons should be two-third the size of the input layer, plus the size of the output layer.
2) The number of hidden neurons should be less than twice the size of the input layer.

In addition, it is necessary to make the various tests on the concerned system to find out the appropriate number which minimizes the prediction error. Fig. 9 demonstrates the corresponding MSE with the hidden layer number via test data. If only two hidden layers are utilized, the MSE of the system becomes large, approximately 20°. Since using three hidden layers, the error decreases dramatically and approaches the optimal value with four layers.

Step 3) Fit model. The training data number is about 70% of the total data, and the remaining 30% is used for the test stage. The training process runs for a fixed number of iterations through the dataset called epochs, which is specified by using the epochs argument. The number of dataset rows is set before updating model weights within each epoch, called the batch size.

Higher value of batch size helps the model to propagate faster, but the precision is compromised. Therefore, the batch size is chosen as 1, which is the minimum value for the highest accuracy. The learning rate controls how quickly the model is adapted to the problem. A lower learning rate needs more training epochs, which leads to the smaller changes made to the weights of each update. Meanwhile, larger learning rates result in rapid changes and require fewer training epochs. However, if the learning rate is too high, the model may cover too quickly to a suboptimal solution or conduct poor prediction performance. Time-based learning rate schedule is utilized, which reduces the learning rate over time by decreasing the learning rate gradually based on the epoch as demonstrated in the following formula:

$$\mathrm{LR}[n] = \frac{\mathrm{LR}[n-1]}{1 + \mathrm{decay} * N\mathrm{epoch}}. \tag{9}$$

In this model, the initial learning rate is 0.1, which is proportional to the batch size since the bigger the batch size, more the immense learning rate and vice versa. Decay the coefficient to decrease the learning rate after each epoch, which is set to 0.002. *N*epoch is the current number of lunching epoch. LR[n] and LR[n-1] are the learning rate of the current epoch and the previous epoch. Learning rate schedule for training models makes large weight update at the beginning of the training procedure with greater learning rate and then reduces the learning rate value; therefore, smaller training updates are generated to weights later in the training process. This function was implemented by the SGD optimization algorithm of Keras.

The number of epochs must be significant enough to reach the best state of the neural network because the weight is updated after every epoch to reduce the error until the saturated point where there is no more change after the epoch process. A high number of batch sizes is processed faster than the smaller size, but its accuracy may not be good enough. In these DL neural networks, the number of epochs is 50 and uses a relatively small batch size of 1 to guarantee high precision of propagation as follows: $Model\_fit \quad (Accelerations, Angles, epochs = 50, batchsize = 1)$.

The loss function is the error function, which shows how big the DL model's error is. Fig. 10 shows the error optimization of 50 epochs. The error function is decreased following each epoch because the model learns more about the concerned data relationship. After epoch 40, the error reduced approximately to 0. This feature indicates that the model works quite

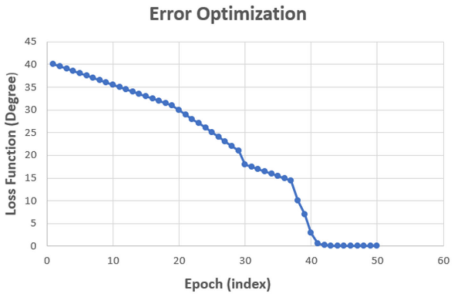


Fig. 10. Error versus epoch.

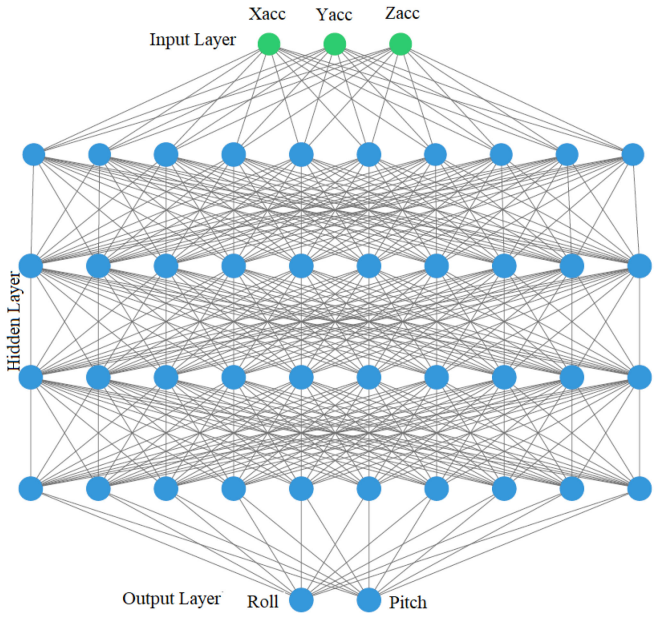


Fig. 11. Overview of DL architecture.

perfectly when the learning epoch occurs more than 40 times with almost zero error function.

Step 4) Evaluate the model. The neural network is evaluated based on the accuracy percentage and the mean squared error from the last epoch update. The model accuracy achieves approximately 99.99% with almost zero error, which means the model has been learned very well.

Step 5) Process the DL model. Accelerations from three axes enter the DL model, and the coherent angles will be acquired. Finally, the DL is the constructed model with architectures of DNNs as shown in Fig. 11.

VII. Experimental Setup

A robust setup was built up for the operating system as shown in Fig. 12. The sensors involved in acquiring data for the experiments are the LSM9DS1 from STMicroelectronics. This device is a system-in-package featuring a tri-axial digital linear acceleration sensor with 16 bit resolution and selectable full-range scale from 2 to ± 16 g, a 3-D digital angular rate sensor with 16 bit resolution, and a full range scale of ± 245, ± 500, and ± 2000 dps. The device also includes a 16 bit tri-axial

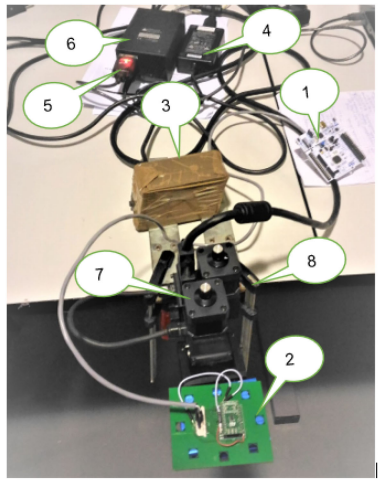


Fig. 12. Test bench for experiment.

magnetic sensor and a magnetic field full scale of ±4, ±8, ±12, and ±16 gauss[29]. The magnetometer and gyroscope were not used for this project. The output data rate of the accelerometer was set to 50 Hz for the stability of transferring signal between the sensor and the computer.

The implementation of the algorithms has been made on an ARM Cortex-M4 based microcontroller STM32F401RE by STMicroelectronics [30], [31]. This device provides 96 kB of RAM and 512 kB of embedded programmable FLASH memory, and the clock frequency is up to 84 MHz. The MCU is assembled on its development board ST NUCLEO-F401RE [32], [33] for easy accessibility to all the required connections.

The sensor is mounted on the STEVAL-MKI159V1 adapter board [34] and connected to the Microcontroller (MCU) development via an inter-integrated circuit (I2C) communication line. A Pan-Tilt Uni Controller (PTU-C46) [35] with resolution 0.051^{o} per position provides fast and accurate positioning of cameras that were manipulated to verify the algorithm performance. The LSM9DS1 was assembled on PTU-C for tracking this device orientation. All the acquisition data were sent to the host computer for signal analysis via a USB cable from the nucleo-board. ML model and DNN model were designed by Python environment with Scikit-learn module and Keras, which is a powerful Python library for development and evaluation of DL models. It wraps the efficient numerical computation libraries Theano and TensorFlow to define and train neural network models [5]. The acceleration acquisitions enter the ML and DL models as real-time input; then, oriental angles are detected practically:

1) NUCLEO-F401RE board;
2) LSM9DS1 sensor mounted on a printed circuit board;
3) counterweight for balance;
4) ac/dc power supply;
5) RS232 cable;
6) PTU-C controller;
7) PTU-C46 pan-tilt unit;
8) heavy clamps.

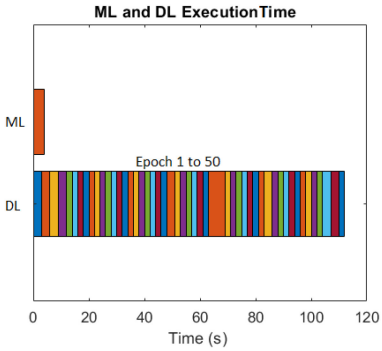


Fig. 13. Model time execution.

TABLE II
ML AND DL MODELS VALIDATION

Parameters	ML	DL
MSE (^{o})	0.187	0.077
MAE (^{o})	0.21	0.191
R2 score	0.98	0.99
Lunch time (s)	4.04	130.12
Training / Test data number	126	13800
Training Time (s)	3.80	100.68
Testing time (s)	0.24	14.44

VIII. EXPERIMENTAL ANALYSIS

A. Designed Model Validation

This part describes the validation of the ML and Dl model based on the training and test sets. The models were operated by a PC processor: Intel(R) Core(TM) i7-10850H CPU @ 2.70 GHz.

As illustrated in Fig. 13, the DL model includes 50 epochs based on gradient descent algorithm; so it consumes a longer time to converge the appropriate result. Meanwhile, the ML model only requires about 4 s to learn the polynomial algorithm and test it. The ML model takes less time than the first two launched epochs of DL execution.

Table II reports the training and testing time of both models. The DL model consumes longer time for the training and testing time due to its complex structure and more extensive data size. Nevertheless, this system acknowledges better performance in general with smaller MSE, MAE, and higher $R2$ score. Hence, the compromise between time and performance from these designed models is what the user should take into account. In the following parts, the real experiment will be described for the validation in the real operation since the sensor is mounted on the mechanical setup. External interference such as mechanical vibration can also challenge the forecast capability of each model.

B. Static Test

In the static test, the evaluation of ML and DL models is demonstrated by the numerous samples on the slope. The inclination was adjusted in the range of 10^o–30^o. LP filter data were acquired to show the comparable observation of ML and DL models. For each inclination level, roll and pitch are controlled alternately and then 900 samples are extracted for the characterized procedure, as shown in Fig. 14.

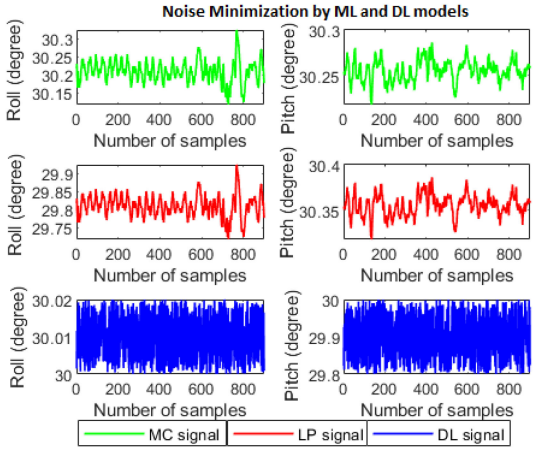


Fig. 14. Model comparison.

The signal behavior of the ML is similar to the LP signal. It is understandable since the ML model learns and trains the output inclination from the ideal sinusoidal curve, based on input accelerations.

The mean absolute percentage error (MAPE) was evaluated to verify the output accuracy of each method. As reported in Table III, the ML model and LP filter obtain the competitive results with a slight difference. Sometimes, ML can predict the closest value to the reference data, but their acquisition form is nearly identical, as shown in Fig. 14.

About DL operation, this model achieves higher accuracy and narrow standard deviation (std). Its outputs are extremely close to the reference angles with a maximum MAPE of 0.14% only. This feature is the advantageous points of the DL model over the ML model.

The DL model with the "adam" algorithm works with sparse gradients and executes step size annealing. The algorithm calculates an exponential moving average of the gradient and the squared gradient. The bias is overcome by first calculating the biased estimates before and then calculating bias-corrected estimates. After numerous epochs, the weights of each hidden layer are updated to optimal values; so the output predictions are very close to the actual value with minimized error. Once the DL structure is constructed correctly, it can offer us a superb performance.

C. Dynamic Test

Two dynamics were carried out: Dynamic 1 as repetitive cycle and Dynamic 2 as step angle.

1) Dynamic 1

In the first dynamic test, PTU rotation was repetitively controlled from 0° to 80° back and forth at a speed of $40^\circ/\text{s}$ as shown in Fig. 15. Ten cycles were recorded to analyze the measurement accuracy and stability of ML and DL models. Each cycle is considered to include two main points 0° and 80°.

As we can see in Table IV, the traditional technique has a significant issue with overshoot and undershoot in the dynamic

TABLE III
STATIC DATA OF ORIENTATION ANGLE

Parameter	LP filter			ML Learning			Deep Learning		
Reference	10^o	20^o	30^o	10^o	20^o	30^o	10^o	20^o	30^o
Roll mean (^{o})	10.12	20.20	29.82	10.12	20.21	30.24	10.01	20.01	30.01
Roll max variation (^{o})	10.27	20.29	29.73	10.26	20.28	30.32	10.02	20.03	30.02
Roll MAPE (%)	1.19	0.99	0.61	1.19	1.034	0.80	0.10	0.14	0.03
Roll std (^{o})	0.013	0.03	0.03	0.013	0.03	0.03	0.005	0.008	0.005
Pitch mean (^{o})	9.82	20.16	30.35	10.23	20.20	30.25	9.99	20.02	30.01
Pitch max variation (^{o})	10.52	20.38	30.38	10.33	20.30	30.26	10.01	20.03	30.03
Pitch MAPE (%)	1.83	0.80	1.15	2.24	0.99	0.82	0.10	0.10	0.03
Pitch std (^{o})	0.14	0.13	0.01	0.12	0.13	0.03	0.004	0.008	0.006

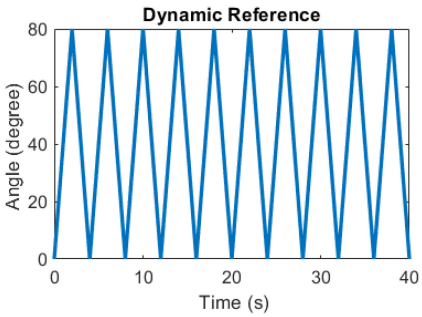


Fig. 15. Dynamic reference for test.

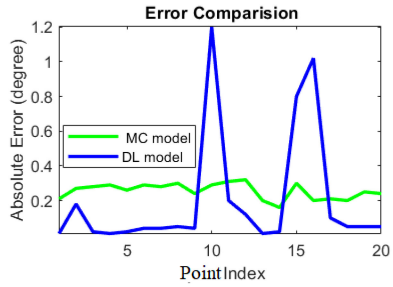


Fig. 16. Error chart of dynamic test.

TABLE IV
DYNAMIC MEASUREMENT RESULT

Cycle index	Reference (^{o})	LP filter (^{o})	ML model (^{o})	DL model (^{o})
Cycle 1	0	0.11	0.21	0.01
	80	81.04	80.27	80.18
Cycle 2	0	-0.44	0.28	0.02
	80	80.99	80.29	80.01
Cycle 3	0	-0.40	0.26	0.02
	80	82.55	80.29	80.04
Cycle 4	0	-0.15	0.28	0.04
	80	81.49	80.30	80.05
Cycle 5	0	-0.30	0.24	0.04
	80	80.70	80.29	78.80
Cycle 6	0	-1.01	0.31	0.2
	80	80.83	80.32	80.12
Cycle 7	0	-0.57	0.20	0.01
	80	82.01	79.84	79.98
Cycle 8	0	-0.04	0.30	0.80
	80	80.37	79.80	81.02
Cycle 9	0	-0.68	0.21	0.1
	80	81.67	80.20	80.05
Cycle 10	0	-0.35	0.25	0.05
	80	81.98	80.24	79.95

case. When the machine just reaches the new angle, it exceeds the target and obtain the overvalue of about 1° or 2° at that moment. The ML model provides more stable data with the error in the range of 0.2°–0.3°. Meanwhile, the DL model can predict higher precision which is almost the same with reference data. However, a big error occurs in cycle 5 and cycle 8, which is more than 1°. This feature shows the similarity with brain operation since it works at very good efficiency in general, but it can work less effectively in some moments. Fig. 16 clearly illustrates the errors of two AI models. The ML model has more stability with repetitive predictions, although it cannot be precise like the DL model.

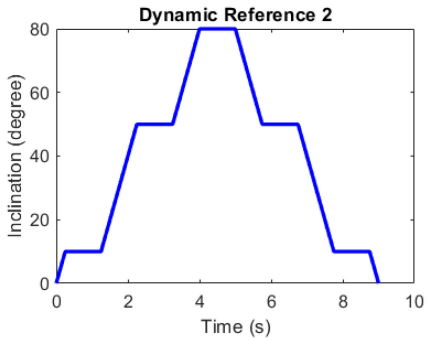


Fig. 17. Dynamic reference for test.

2) Dynamic 2

Another dynamic test was carried out for the comparison between two AI proposed models as shown in Fig. 17. In this test, more random angles are rotated, from (0° to 10°),(10° to 50°), and (50° to 80°), and then rotates back with the same tilt step. Each time a new inclination is reached, the machine stays at that angle for 1 s before moving to the new angle. As reported in Table VI, the ML model generates larger error than DL model in general. However, when the angle decreases from 80° to 50°, DL conducts a bigger error unexpectedly.

TABLE V
DYNAMIC COMPARISON BETWEEN THREE TECHNIQUES

Test	MAE (LP)	MAE (ML)	MAE (DL)	MSE (LP)	MSE (ML)	MSE (DL)
Dynamic 1	0.884	0.255	0.201	1.262	0.066	0.161
Dynamic2	0.582	0.208	0.155	0.771	0.044	0.002

TABLE VI
DATA COMPARISON BETWEEN TWO AI MODELS

Reference (^{o})	ML error (^{o})	DL error (^{o})
0	0.17	0.03
10	0.2	0.03
50	0.22	0.1
80	0.28	0.09
50	0.22	0.4
10	0.21	0.02
0	0.16	0.02

TABLE VII
GENERAL COMPARISON BETWEEN ML AND DL MODELS

Feature	ML model	DL model
Launch Time	Faster	Slower
Precision	Lower	Extremely higher
Dynamic stability	Higher	Lower

D. Overall Comparison

Table V evaluates the performance of all three techniques in two dynamic tests. In both dynamic tests, the traditional technique suffers quite large errors in terms of MAE and MSE, while the ML outputs the better outcomes apparently. The advantages of ML model over the LP method were clearly demonstrated in the dynamic tests. On the other hand, the DL model still achieves the good prediction with the smallest error with respect to the others except the MSE in the first dynamic test because of the large error in two cycles. MAE is less biased for higher values; so it does not reflect the performance completely when dealing with large error values as MSE because MSE is highly biased for higher values.

Table VII summarizes the comparison between the ML model and the DL model in inclination execution. Overall, the DL model has better command of tracking angle with higher accuracy of acquisition. ML can guarantee higher stability thanks to its repetitive performance. About launch time for starting model, the ML has the advantage thanks to the less complexity of structure.

IX. CONCLUSION

In this article, the two powerful models in AI were successfully designed for the inclination measurement. Both static tests and dynamic tests were analyzed in detail to characterize the pros and cons of each model. The ML model accomplishes a competitive operation to traditional calculation with slight advantage of precision and less overshoot in dynamic case, while the DL technique successfully accomplishes the highest accuracy. The DL model can predict the output almost the same with reference data, while ML assures the permanent propagation for the system. In the dynamic test, there are some points where the DL model conducts more significant error. This article opens a new approach to orientation tracking by AI techniques, which significantly contributes to the development of inclinometer applications. In future work, the optimization of the DL model will be researched further. The aim is to improve its stability to guarantee high accuracy in dynamic cases permanently.

REFERENCES

[1] S. Shin, D. Kim, and Y. Seo, "Controlling mobile robot using IMU and EMG sensor-based gesture recognition," in *Proc. 9th Int. Conf. Broadband Wirel. Comput., Commun. Appl.*, 2014, pp. 554–557.
[2] F. Aghili and A. Salerno, "Driftless 3-D attitude determination and positioning of mobile robots by integration of IMU with two RTK GPSs," *IEEE/ASME Trans. Mechatronics*, vol. 18, no. 1, pp. 21–31, Feb. 2013.
[3] S. Zhu, S. H. Li, Y. Liu, and Q. W. Fu, "Low-cost MEMS-IMU/RTK tightly coupled vehicle navigation system with robust lane-level position accuracy," in *Proc. 26th Saint Petersburg Int. Conf. Integr. Navig. Syst.*, 2019, pp. 1–4.
[4] M. Carratù, S. D. Iacono, A. Pietrosanto, and V. Paciello, "IMU self-alignment in suspensions control system," in *Proc. IEEE Int. Instrum. Meas. Technol. Conf.*, 2019, pp. 1–6.
[5] M. L. Hoang and A. Pietrosanto, "A new technique on vibration optimization of industrial inclinometer for MEMS accelerometer without sensor fusion," *IEEE Access*, vol. 9, pp. 20295–20304, 2021.
[6] M. L. Hoang and A. Pietrosanto, "A robust orientation system for inclinometer with full-redundancy in heavy industry," *IEEE Sens. J.*, vol. 21, no. 5, pp. 5853–5860, Mar. 2021.
[7] M. L. Hoang, S. D. Iacono,V. Paciello, and A. Pietrosanto, "Measurement optimization for orientation tracking based on no motion no integration technique," *IEEE Trans. Instrum. Meas.*, vol. 70, Nov. 2021, Art no. 9503010, doi: 10.1109/TIM.2020.3035571.
[8] M. L. Hoang, M. Carratù, V. Paciello, and A. Pietrosanto, "A body temperature-indoor condition monitor and activity recognition by MEMS accelerometer based on IoT-alert system for people in quarantine due to COVID-19," *Sensors*, vol. 21, no. 7, 2021, Art. no. 2313.
[9] M. L. Hoang and A. Pietrosanto, "Yaw/heading optimization by drift elimination on MEMS gyroscope," *Sens. Actuators Phys.*, vol. 325, 2021, Art. no. 112691.
[10] M. L. Hoang, M. Carratù, M.A. Ugwiri,V. Paciello, and A. Pietrosanto, "A new technique for optimization of linear displacement measurement based on MEMS accelerometer," in *Proc. Int. Semicond. Conf.*, 2020, pp. 155–158.
[11] M. Carratù, S. D. Iacono, M. Long Hoang, and A. Pietrosanto, "Energy characterization of attitude algorithms," in *Proc. IEEE 17th Int. Conf. Ind. Inform.*, 2019, vol. 1, pp. 1585–1590.
[12] M. L. Hoang and A. Pietrosanto, "An effective method on vibration immunity for inclinometer based on MEMS accelerometer," in *Proc. Int. Semicond. Conf.*, 2020, pp. 105–108.
[13] TensorFlow. May 2020. [Online]. Available: https://www.tensorflow.org/api_docs/python/tf /keras
[14] M. L. Hoang, A. Pietrosanto, S. D. Iacono, and V. Paciello, "Pre-processing technique for compass-less Madgwick in heading estimation for industry 4.0," in *Proc. IEEE Int. Instrum. Meas. Technol. Conf.*, 2020, pp. 1–6.
[15] M. A. Kaleem and M. Khan, "Significance of additive manufacturing for industry 4.0 with introduction of artificial intelligence in additive manufacturing regimes," in *Proc. 17th Int. Bhurban Conf. Appl. Sci. Technol.*, 2020, pp. 152–156.
[16] S. Ruhela and S. Riaz, "An intelligent combination: Assessing the impact of harmonized emotional and artificial intelligence for the success of industry 4.0," in *Proc. 10th Int. Conf. Comput., Commun. Netw. Technol.*, 2019, pp. 1–5.

[17] R. Choudhary and H. K. Gianey, "Comprehensive review on supervised machine learning algorithms," in *Proc. Int. Conf. Mach. Learn. Data Sci.*, 2017, pp. 37–43.
[18] W. Lv and J. Lei, "Deep learning development review," in *Proc. 3rd Int. Conf. Adv. Electron. Mater., Comput. Softw. Eng.*, 2020, pp. 171–174.
[19] L. Romeo, M. Paolanti, G. Bocchini, J. Loncarski, and E. Frontoni, "An innovative design support system for industry 4.0 based on machine learning approaches," in *Proc. 5th Int. Symp. Environ.-Friendly Energies Appl.*, 2018, pp. 1–6.
[20] R. Shams, "Developing machine learning products better and faster at startups," *IEEE Eng. Manage. Rev.*, vol. 46, no. 3, pp. 36–39, Sep. 2018.
[21] R. Ozdemir and M. Koc, "A quality control application on a smart factory prototype using deep learning methods," in *Proc. IEEE 14th Int. Conf. Comput. Sci. Inform. Technol.*, 2019, vol. 1, pp. 46–49.
[22] M. Miškuf and I. Zolotová, "Comparison between multi-class classifiers and deep learning with focus on industry 4.0," in *Proc. Cybern. Inform.*, 2016, pp. 1–5.
[23] L. Han, Y. Lin, G. Du, and S. Lian, "DeepVIO: Self-supervised deep learning of monocular visual inertial odometry using 3D geometric constraints," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 6906–6913.
[24] S. Cortés, A. Solin, and J. Kannala, "Deep learning based speed estimation for constraining strapdown inertial navigation on smartphones," in *Proc. IEEE 28th Int. Workshop Mach. Learn. Signal Process.*, 2018, pp. 1–6.
[25] H. Yan, J. Wan, C. Zhang, S. Tang, Q. Hua, and Z. Wang, "Industrial big data analytics for prediction of remaining useful life based on deep learning," *IEEE Access*, vol. 6, pp. 17 190–17197, 2018.
[26] A. Belmonte-Hernández, G. Hernández-Peñaloza, D. Martín Gutiérrez, and F. Álvarez, "SWiBlux: Multi-sensor deep learning fingerprint for precise real-time indoor tracking," *IEEE Sens. J.*, vol. 19, no. 9, pp. 3473–3486, May 2019.
[27] J. Huang, Z. Huang, and K. Chen, "Combining low-cost inertial measurement unit (IMU) and deep learning algorithm for predicting vehicle attitude," in *Proc. IEEE Conf. Dependable Secure Comput.*, 2017, pp. 237–239.
[28] L. Cao, "Practical issues in implementing a single-pole low-pass IIR filter [*applications corner*]," *IEEE Signal Process. Mag.*, vol. 27, no. 6, pp. 114–117, Nov. 2010.
[29] STMicroelectronics, "LSM9DS1-iNEMO inertial module: 3D accelerometer, 3D gyroscope, 3D magnetometer - data sheet," Mar. 2015.
[30] STMicroelectronics, "Ultra-low-power arm cortex-m4 32-bit MCU FPU, 105 dmips, 512 kb flash/96 kb ram, 11 tims, 1 adc, 11 comm. interfaces," Jan. 2015.
[31] STMicroelectronics, "Stm32f401xb/c and stm32f401xd/e advanced arm-based 32-bit MCUs," 737 Dec. 2018.
[32] STMicroelectronics, "Stm32 nucleo-64 boards - product specifications," Oct. 2018.
[33] STMicroelectronics, "Stm32 nucleo-64 boards - user manual," Dec. 2017.
[34] STMicroelectronics, "Lsm9ds1 adapter board for a standard dil24 socket," Sep. 2016.
[35] R. R. Burlingame, "Pan-tilt unit (model PTU) user's manual," Directed Perception, Inc., California, CA, USAm Oct. 2000.
[36] G. İrsel, "The machine learning concept for an inclination sensor" in *Proc. 6th Int. Conf. Trends Agricultural Eng.*, Prague, Czech Republic, Sep. 2016, pp. 7–9.
[37] M. Violeta, M. Luštrek, and M. Gams, "Combining domain knowledge and machine learning for robust fall detection," *Expert Syst. J. Knowl. Eng.*, vol. 31, no. 3, pp. 163–75, 2013.
[38] C. Li, M. Lin, T. L. Yang, and C. Ding, "Integrating the enriched feature with machine learning algorithms for human movement and fall detection,"*J. Supercomput.,* vol. 67, no. 3, pp. 854–865, 2013.
[39] G. Shobha and S. Rangaswamy, "Chapter 8 - Machine Learning," in *Handbook of Statistics*. Amsterdam, The Netherlands: Elsevier, 2018, pp. 197–228.
[40] S. R. Shakya, C. Zhang, and Z. Zhou, "Comparative study of machine learning and deep learning architecture for human activity recognition using accelerometer data," *Int. J. Mach. Learn. Comput.*, vol. 8, no. 6, pp. 577–582, Dec. 2018.
[41] B. Alejandro, C. Alejandro, S. Yago, and I. Pedro, "A comparison of machine learning and deep learning techniques for activity recognition using mobile devices," *Sensors*, vol. 19, no. 3, 2019, Art. no. 521.
[42] F. Li, K. Shirahama, M. A. Nisar, L. Köping, and M. Grzegorzek, "Comparison of feature learning methods for human activity recognition using wearable sensors," *Sensors*, vol. 18, no. 2, 2018, Art. no. 679.
[43] S. Luís *et al.*, "Activity classification using accelerometers and machine learning for complex construction worker activities," *J. Building Eng.*, vol. 35, 2021, Art. no. 102001.
[44] G. Eduardo, B. Luciano, C. R. Wagner, P. S. Ana, M. Izabela, and C. Alessandro, "Machine learning algorithms for activity-intensity recognition using accelerometer data," *Sensors*, vol. 21, no. 4, 2021, Art. no. 1214.
[45] F. Esther and A. G. Bonomi, "Accelerometer-based human activity recognition for patient monitoring using a deep neural network," *Sensors*, vol. 20, no. 22, 2020, Art. no. 6424.
[46] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980v9*.
[47] R. D. Sriram, "Neural networks," in *Proc. Intell. Syst. Eng., Springer London*,1997, pp. 471–542.
[48] C. J. Fisher, "AN-1057: Using an accelerometer for inclination sensing," *Analog Device*, One Technology Way, Norwood, MA, USA, 2010.

Minh Long Hoang (Member, IEEE) was born in Hanoi, Vietnam, in 1994. He received the B.S. degree in renewable energy from the University of Science and Technology, Hanoi, Vietnam, in 2015, and the M.S. degree in electronic engineering in 2018 from the University of Salerno, SA, Italy.

Since Nov 2018, immediately after graduation from M.S. degree, he has been working on a Nationally Operative Program (PON) project of European Union about "Industry 4.0 oriented enhancement of Inertial Platform performance" as a Ph.D. student with the University of Salerno, Salerno, Italy. Also within this project, he has been collaborating with Sensor System srl Company (Italy) and Baumer Company (Germany) as Researcher with R&D Department in the field of inclinometer. His research interests include inertial measurement unit sensors, microelectromechanical system, machine learning, deep learning, real-time measurements, Internet of Things, and signal processing.

Antonio Pietrosanto (Senior, Member) was born in 1961. He received the Ph.D. degree in electrical engineering from the University of Naples, Naples, Italy, in 1990.

He was an Assistant Professor of electrical and electronic measurement with the University of Salerno, Salerno, Italy, from 1991 to 1999. He has been a Full Professor of instrumentation and measurement with the University of Salerno, Fisciano, Italy, since 2001. He has been the Founder of three spin-off companies with the University of Salerno (UniSA): SPRING OFF, Metering Research, and Hippocratica Imaging. He has coauthored more than 150 articles in international journals and conference proceedings. His main research activities include the fields of instrument fault detection and isolation (IFDIA), sensors, wireless sensor networks (WSNs), real-time measurements, embedded systems, metrological characterization of measurement software, advanced system for food quality inspection, and image-based measurements.

Prof. Pietrosanto is the President of the Didactic Board of Electronic Engineering of UniSA.