Assured time series forecasting using inertial measurement unit, neural networks, and state estimators

Ashvini Kulkarni[1,3], Augusta Sophy Beulet Paul[1,2]
[1]School of Electronics Engineering, Vellore Institute of Technology, Chennai, India
[2]Centre for Nano-electronics and VLSI Design, Vellore Institute of Technology, Chennai, India
[3]Department of Electronics and Telecommunication, International, Institute of Information Technology, Pune, India

Article Info

Article history:

Received Feb 13, 2024
Revised Feb 17, 2025
Accepted Mar 15, 2025

Keywords:

Convolution neural network
Kalman filter
Long-short term memory
Multi-layer perception
State estimators

ABSTRACT

Pedestrian dead reckoning (PDR) technology has be come an important method for predicting the position of an object or person. Sensor-based positioning is widely used because of its readily available hardware and acceptable accuracy, especially with PDR algorithms integrated with machine learning and deep learning. There are two challenges in this context. Conventional state-estimator methods suffers from dynamics, making the deployment and management of nonlinear dynamics become difficult. Training an effective neural network model with a few inertial measurement unit (IMU) samples is also challenging. This study investigates the integration and comparison of advanced state estimation algorithms such as the Kalman filter (KF), extended Kalman filter (EKF), and sigma point Kalman filter (SPKF) with deep neural networks, including multi-layer perceptron (MLP), convolutional neural network (CNN), and long short-term memory (LSTM). The aim is to improve the reliability of forecasting and prediction tasks, particularly when processing IMU data. This study conducts a comprehensive performance comparison between state estimators integration with deep learning models, evaluating their effectiveness in addressing the challenges of estimation and prediction. The preliminary results show that the feature forecasting rate of the proposed method can reach a root mean square error (RMSE) value of 0.31 (EKF-LSTM) and 1.50 (SPKF-LSTM).

Corresponding Author:

Augusta Sophy Beulet Paul
School of Electronics Engineering, Vellore Institute of Technology
Chennai, Tamilnadu, 600127, India
Email: augustasophyt.p@vit.ac.in

1. INTRODUCTION

This pedestrian dead reckoning (PDR) technology has recently gained significant attention owing to its low cost and potential for noninvasive navigation. The development of accurate and efficient navigation systems is crucial for various applications, including autonomous vehicles, and robotics in indoor and outdoor environments. In indoor environments where GPS signals may be unavailable or unreliable, accurate position estimation with PDR allows users to navigate with precision for personal navigation in large buildings, logistics operations in warehouses, and guiding autonomous robots in indoor spaces.

Several studies have explored the integration of state-estimation algorithms using deep learning models for localization and navigation. However, there is still scope for improvement in terms of the accuracy and efficiency. Indoor positioning systems monitor and identify objects for surveillance and activity recognition using wireless technology, optical sensors, ultrasonic sensors, radio frequency identification (RFID), and inertial measurement units (IMUs). These sensors and location-based navigation services are commonly used [1]. A KF-based state estimator predicts and aids intelligent-sensor-based monitoring and navigation. It precisely locates objects for mining, self-driving cars, and location-based applications by using the least-squares estimate [2]. The challenge is to develop a method that combines traditional state estimation methods with advanced deep learning algorithms to enhance the navigation and localization accuracy in dynamic situations. This integration enables real-time navigation guidance, collision avoidance, and navigation-related tasks using forecasting techniques. This study seeks to fill this gap by exploring the integration of state estimation algorithms with deep learning models to enhance future forecasting, estimation, and prediction in IMU-based navigation and localization systems. Sensors such as accelerometers, gyroscopes, cameras, light detection and ranging (LiDAR), radar sensors, and similar devices collect data on the moving object and environment during sensor-based navigation [3].

The usefulness of a sensor is determined by its application, and data fusion or integration is required for multisensor navigation applications. This method combines sensor data to improve the accuracy, completeness, and dependability of the navigation system with environmental interpretation [4]–[7]. These systems were analytically modeled using moving-object dynamics and kinematics [8], [9]. Accurate position estimation and prediction contribute to providing a seamless and immersive user experience for indoor navigation applications. This considers sensor calibration errors, internal measurement errors, mechanical development imprecision, friction coefficients, inertia, and other factors [10]. To determine whether a pedestrian is in a stationary or nonstationary phase, periodic footsteps and strap-down inertial calculations are used to rectify the zero-velocity update (ZUPT) and adjust location inaccuracies [11], [12]. In some applications, low-cost ultrasonic sensors are employed, resulting in erroneous stride length measurements, which affect tracking performance and produce errors. State estimators, such as the EKF, are used to minimize positional errors [13], [14]. Sensor-based research occasionally lacks precision, resilience, and response time. These limitations are caused by the noise, sensor range, and sensor interference. These constraints can be overcome through sensor redundancy, sensor fusion, enhanced algorithms, and deep-learning techniques [15]–[17].

In navigation, dead reckoning is a process in which particle filtering and map matching accurately estimate the trajectory of a nonstationary object. The most difficult aspect of inertial dead reckoning is the estimation of acceleration noise, bias, and orientation direction. The particle filter provides a high-fidelity estimate of the present state based on sensor data, whereas the map matching method can correct for any drift or irregularity in the projected trajectory by aligning it with the road network. Inconsistent road network map data impact the map matching accuracy [18]. Incorrect data associations can lead to localization errors and affect the overall performance of the navigation systems.

Sensor-based navigation collects data from sophisticated sensors; hence, deep learning models can quickly handle complex data from such sensors. Artificial neural networks (ANNs) advantages over traditional neural networks include ease of use, quick learning, strong generalization, few training-related errors, and lower standard weights. Only a few deep learning prediction methods include deep and recurrent neural networks (RNN), and most solutions are application specific and involve static data analysis and simple time-series modeling [19]. Incorporating time-series models in navigation systems helps bridge the gap between static map-based approaches and dynamic real-world navigation tasks, which are uncertain in nature.

By explicitly modeling the temporal dynamics, uncertainty, and adaptability, time-series modeling enables navigation systems to achieve high accuracy, reliability, and usability in various applications and environments. This time-series modeling is critical in sensor-based applications, such as IMU-based navigation, because it requires the systematic collection and analysis of past observations. Because of the intricate distribution of extensive time-series data, hybrid navigation systems are essential for enhancing the accuracy and utility of hybrid forecasting models [20], [21]. Various strategies have been proposed to address these problems, such as employing Kalman filters (KF) and time-series forecasting models. However, achieving greater prediction performance and robustness in real-world scenarios, including dynamic navigation and localization, requires a more integrated approach to effectively merge these two models. The primary contributions of this study are as follows:

- The study aims to determine the suitability of traditional state estimators and advanced neural network architectures in dynamic environments.
- The proposed algorithm enhances the precision of predictions and estimations by applying KF expansion to the expression of position and velocity estimation variants.

– This study evaluates and compares the performance of state estimators and neural network models to determine their effectiveness in blending estimation and forecasting tasks.
– The hypothesis is that the proposed method can improve the accuracy, resilience, and efficiency of navigation and localization in dynamic situations compared with traditional state estimation methods.

The paper is organized as follows. Section 2 explores the existing research in the domain of time-series forecasting for diverse navigation environments and the proposed solution. Section 3 discusses the methods used and section 4 discusses the results of the position and velocity forecasting. Section 5 concludes the study and discusses the scope of future research.

2. LITERATURE SURVEY

There are numerous domains in which sensors can be implemented for navigation purposes, such as in autonomous vehicles and robotics. Sensor-based navigation encounters issues, such as sensor noise, environmental unpredictability, multiple sensor integration, dynamic motion, latency, battery consumption, and error propagation. Eventually, sensor noise and drift can introduce errors in navigation computations, which require the use of calibration and filtering techniques. It is crucial to adjust to various environments and overcome the signal barriers. Error propagation, which can result in accumulated errors and reduced performance, requires error mitigation strategies. Tackling these challenges typically requires a combination of hardware enhancements, sensor fusion algorithms, signal processing methods, deep learning, and thorough testing and validation in real-world scenarios.

Time series forecasting uses statistical models to estimate future values based on historical data. Time series observations are difficult to interpret because of their order and temporal correlations. Time-series models were evaluated after substantial preprocessing to eliminate unordered timestamps, missing values, outliers, and noise. Most machine learning algorithms use hyperparameter optimization, feature selection, preprocessing, and domain expertise for forecasting [22]. Sensor time series data is much larger, and manually tweaking hyperparameters are computationally expensive, so they should be optimized automatically [23]. Time series forecasting has long been employed because of its accuracy and interpretability.

Time series forecasting uses statistical approaches and neural network models. Most statistical methods for forecasting time series employ statistical models to comprehend data. Mathematical models of data behavior can predict future values. This model offers simple auto-regressive and complex integrated moving average models for time series data analysis and forecasting. Many statistical models assume stationarity, linearity, and normality; however, they rarely satisfy these assumptions. Another method is to employ the hidden Markov model (HMM), which shows the probabilistic relationship between observations and hidden states. HMM dynamically learns system outcomes without statistical training [24].

The primary focus of Bayesian optimization (BO) and hyperband (HB) selection is the hyperparameter type in a machine learning model, which is crucial for applying optimization methods to deep learning models [25]. The study of the forecasting process is still in its infancy, owing to patterns including trends, seasonality, outliers, drifts, and unexpected shifts that should be treated differently. Table 1 lists a summary of the time-series forecasting issues.

Table 1. Summary of time-series forecasting issues

Challenges in time series forecasting	Parameters
Data quality	Intermittent data, sparsity, new time series, absent time stamps, and gaps
Data latency	The frequency of historical data received by forecasting models can extend the forecasting horizon and diminish the ability to capture recent effects with autoregressive components
Predict predictors as inputs	Modeling predictors for target variables requires inputs for production forecasting.
Retraining monitoring frequency	Forecasters should decide whether to retrain models frequently or in reaction to feature drift or performance decline.

2.1. Deep learning model for time series prediction

Deep learning uses brain-inspired models in machine learning. Deep neural networks improve the prediction and categorization using the numerous layers of interconnected nodes. Traditional time-series forecasting methods generally provide temporal data modeling and prediction. However, complicated data patterns or abnormalities can pose a challenge to these solutions. Autoregressive (AR), autoregressive moving average (ARMA), and autoregressive integrated moving average (ARIMA) models are frequently employed for time-series forecasting because of their interpretability, computational efficiency, and minimal data requirements compared to deep learning models [26]–[28]. Because navigation applications require large amounts of data, neural networks offer distinct advantages in capturing complex patterns, handling high-dimensional data, and adapting to diverse forecasting tasks. Deep neural networks contain input and output layers. The input layer feeds the deep learning model data, whereas the output layer provides the final prediction. As the function goes backward through the layers, backpropagation adjusts its weights and biases to train the model. Gradient descent is used to calculate the prediction errors before traveling backward through the layers to change the weights and biases of the function. Neural networks forecast and repair errors via forward and backpropagation. The precision of the algorithm increases over time, and further research is required to improve its time-series analysis accuracy, efficiency, and architecture [29], [30].

3. METHOD

This study aims to address the issue of insufficient integration between state estimation methods and deep learning models, which restricts dynamic adaptability, real-time accuracy, and localization precision in sensor-based navigation systems. To overcome this, we propose a novel Kalman filter neural network (KFNN) model, designed to enhance sensor-based forecasting and trajectory estimation. This model incorporates the KF for sensor fusion, specifically for integrating accelerometer and gyroscope data to estimate position and velocity. The proposed architecture includes the prediction of position velocity, the measurement association, the state update and tracking path management performed under the auspices of the KF. The predictions obtained from these processes are subsequently fed as inputs to a neural network to improve prediction accuracy. In turn, the neural network acquires knowledge about the relationship between the sensor data and sensor fusion output derived by the KF. Using the data estimated by the KF to train the model, the proposed approach has the potential to be applied effectively to sensor inputs that have not been previously observed and accurately predict motion parameters. The neural network learns to decipher intricate correlations and patterns in the data, thus enhancing the overall accuracy of the motion estimation system.

Figure 1 depicts the proposed coarse-to-fine hybrid forecast system. It consists of the following major sections: sensor layout, sensor fusion, state estimators, and forecasting model. State estimators or filters are mathematical strategies for estimating the internal state of a dynamic system using noisy or incomplete observations (measurements) of their outputs. The state of a system is often defined as a set of variables that fully characterize its behavior at a given time, such as position and velocity. The methodology was validated through rigorous experimentation and comparative analysis, demonstrating promising results in accurately forecasting motion parameters from accelerometer and gyroscope data. State estimators are important for velocity and position prediction because they employ sensor data to estimate the internal state of a dynamic system. State estimators accurately determine the position and velocity, along with factors such as noise and uncertainty, whereas sensor data estimate motion. KF, EKF, and SPKF are variations of the KF technique used for state estimation in dynamic systems, and their performance predictions are evaluated. The filter enhances the predictions by repeatedly updating the state estimations fed into a neural network architecture designed to operate with a KF output. The KFNN model was trained on the data estimated using the KF. Backpropagation was used as a training approach, adjusting the model parameters to reduce prediction errors. Gradient descent was used to iteratively adjust the weights and biases.

The proposed method assesses the performance of the KFNN model using standard metrics, such as the mean squared error (MSE) and root mean square error (RMSE). The accuracy of the motion parameter predictions was compared to the ground-truth data. Experiments were conducted in both simulated and real-world settings using typical hardware configurations for data collection and processing. The study utilized Keras for the frontend and TensorFlow for the backend, with Sklearn used for the sensor-based models and evaluation metrics. Adjusting the learning rate during training enhances performance and accelerates challenge-specific training. Drop-based learning rates are controlled by four factors: the initial learning rate, drop rate, epoch, and epoch drop. During the training phase, the batch size of the training data with hidden layers was activated using rectified linear units (ReLU). Sensitivity analysis was conducted by perturbing the input features and observing the resulting changes in the predictions. A fixed number of features were identified to compare the overall prediction by employing predictor importance scores for training and validation. A perturbation value of 1e-5 was used, including all multi-layer perceptron (MLP), convolutional neural network (CNN), and LSTM neural network types with all features and selecting a subset of the four features primarily for the available

IMUs, starting from the neural network identified as yielding the best outcome performance. For the latter, the analysis began with the data provided by each sensor in isolation, and included data from pairs of sensors: location (x,y) and velocity ($\ddot{x}$, $\ddot{y}$). After identifying six indoor and outdoor activities, an exploratory analysis of the most significant extracted features was conducted. The KF was employed, incorporating stochastic elements to accommodate stochastic processes and uncertainties, aiding feature generation, and was applied to the standardized neural network owing to its performance and efficiency trade-off.

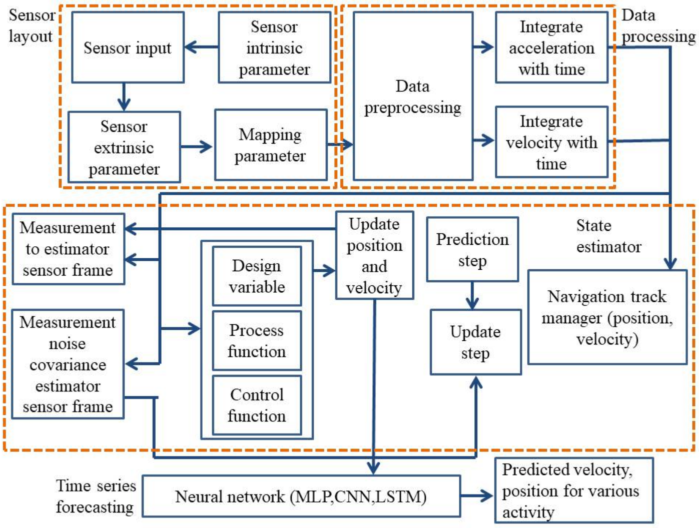

Sensor layout
Sensor input
Sensor intrinsic parameter
Sensor extrinsic parameter
Mapping parameter
Data preprocessing
Integrate acceleration with time
Integrate velocity with time
Data processing
Measurement to estimator sensor frame
Measurement noise covariance estimator sensor frame
Design variable
Process function
Control function
Update position and velocity
Prediction step
Update step
State estimator
Navigation track manager (position, velocity)
Time series forecasting
Neural network (MLP,CNN,LSTM)
Predicted velocity, position for various activity

Figure 1. Proposed coarse-to-fine hybrid forecast system

3.1. Data collection

Experiments were conducted using a realtime IMU sensor, and data were collected from various indoor and outdoor environments to evaluate the performance of the proposed approach. The datasets include diverse scenarios, such as pedestrian navigation and user motion, covering different motion dynamics and environmental conditions. The experiments were conducted on a group of 12 volunteers aged 21–40 years. Each participant wore an integrated circuit consisting of an MPU 6050 sensor, which has a built-in 3-axis accelerometer and a 3-axis gyroscope. The prototype module was tied to the participant's ankle to capture its activities. They are subjected to activities such as walking and cycling (outdoors), ascending stairs, descending stairs, sitting, and walking underground (indoors). Table 2 lists the specifications of the sensors used for data collection.

Table 2. MPU6050 data collection specification

Sensor category	Sensitivity	Frequency response
Accelerometer	Three-axis accelerometer full-scale range ±8g	LPF filter response: 5 Hz min, 260 Hz max
Gyroscope	Three-axis Gyroscope full-scale range ±500 °/s	LPF filter response: 5 Hz min, 256 Hz max

3.2. State estimators

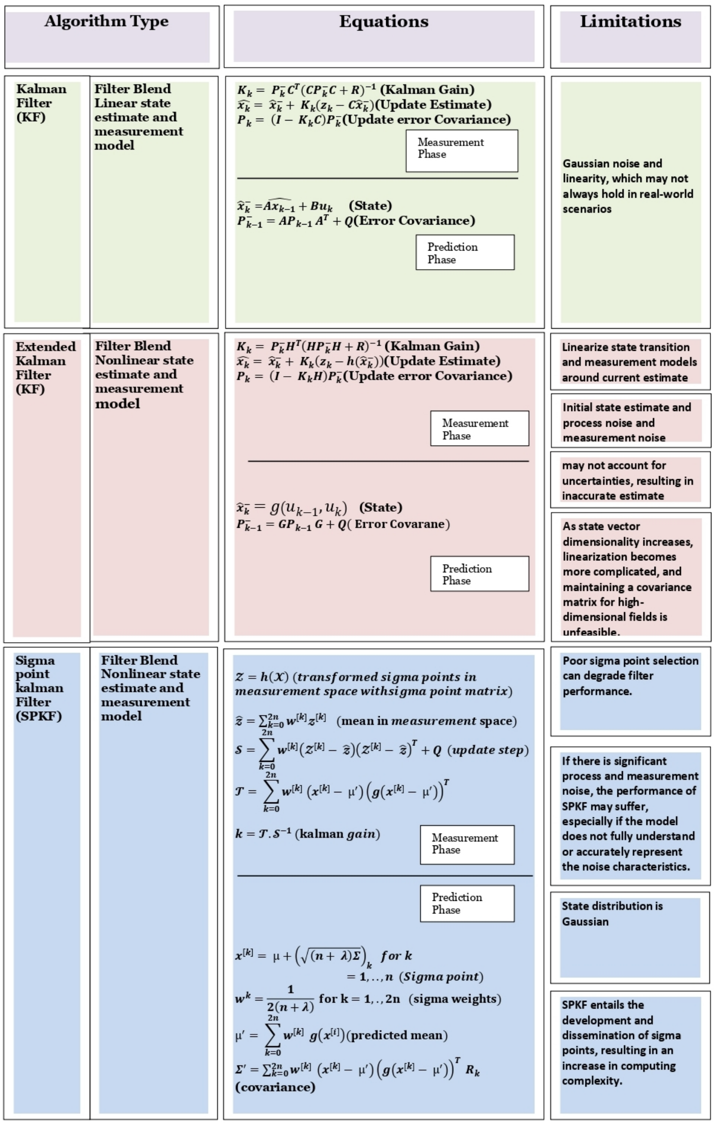
The state estimators KF, EKF, and SPKF plays significant roles in the estimation tasks, which depend on the specific characteristics of the system dynamics, measurement models, and noise attributes. In the context of state estimation, the state vector is represented by (1).

$$x = \left[x, y, \ddot{x}, \ddot{y}\right]^T \tag{1}$$

Where,

- $\ddot{x}$ and $\ddot{y}$ are the velocities in the horizontal and vertical dimensions, respectively.
- x and y are the positions in the horizontal and vertical dimensions, respectively.

The state estimator equations for the KF, EKF, and SPKF are listed in Figure 2.



Algorithm Type		Equations	Limitations
Kalman Filter (KF)	Filter Blend Linear state estimate and measurement model	$K_k = P_k^- C^T (CP_k^- C + R)^{-1}$ (Kalman Gain) $\widehat{x_k} = \hat{x}_k^- + K_k(z_k - C\hat{x}_k^-)$(Update Estimate) $P_k = (I - K_k C)P_k^-$(Update error Covariance) Measurement Phase $\hat{x}_k^- = \widehat{Ax_{k-1}} + Bu_k$ (State) $P_{k-1}^- = AP_{k-1} A^T + Q$(Error Covariance) Prediction Phase	Gaussian noise and linearity, which may not always hold in real-world scenarios
Extended Kalman Filter (KF)	Filter Blend Nonlinear state estimate and measurement model	$K_k = P_k^- H^T (HP_k^- H + R)^{-1}$ (Kalman Gain) $\widehat{x_k} = \hat{x}_k^- + K_k(z_k - h(\hat{x}_k^-))$(Update Estimate) $P_k = (I - K_k H)P_k^-$(Update error Covariance) Measurement Phase $\hat{x}_k^- = g(u_{k-1}, u_k)$ (State) $P_{k-1}^- = GP_{k-1} G + Q($ Error Covarane$)$ Prediction Phase	Linearize state transition and measurement models around current estimate Initial state estimate and process noise and measurement noise may not account for uncertainties, resulting in inaccurate estimate As state vector dimensionality increases, linearization becomes more complicated, and maintaining a covariance matrix for high-dimensional fields is unfeasible.
Sigma point kalman Filter (SPKF)	Filter Blend Nonlinear state estimate and measurement model	$\mathcal{Z} = h(\mathcal{X})$ $(transformed\ sigma\ points\ in\ measurement\ space\ withsigma\ point\ matrix)$ $\hat{z} = \sum_{k=0}^{2n} w^{[k]} z^{[k]}$ (mean in $measurement$ space) $\mathcal{S} = \sum_{k=0}^{2n} w^{[k]} (\mathcal{Z}^{[k]} - \hat{z})(\mathcal{Z}^{[k]} - \hat{z})^T + Q$ $(update\ step)$ $\mathcal{T} = \sum_{k=0}^{2n} w^{[k]} (x^{[k]} - \mu')\left(g(x^{[k]} - \mu')\right)^T$ $k = \mathcal{T}.\mathcal{S}^{-1}$ (kalman $gain$) Measurement Phase Prediction Phase $x^{[k]} = \mu + \left(\sqrt{(n+\lambda)\Sigma}\right)_k$ $for\ k = 1,..,n$ $(Sigma\ point)$ $w^k = \frac{1}{2(n+\lambda)}$ for k = 1,.,2n (sigma weights) $\mu' = \sum_{k=0}^{2n} w^{[k]}\ g(x^{[i]})$(predicted mean) $\Sigma' = \sum_{k=0}^{2n} w^{[k]} (x^{[k]} - \mu')\left(g(x^{[k]} - \mu')\right)^T R_k$ (covariance)	Poor sigma point selection can degrade filter performance. If there is significant process and measurement noise, the performance of SPKF may suffer, especially if the model does not fully understand or accurately represent the noise characteristics. State distribution is Gaussian SPKF entails the development and dissemination of sigma points, resulting in an increase in computing complexity.

Figure 2. State estimator equations for the KF, EKF, and SPKF

3.3. Multi-layer perceptron model implementation

Time series forecasting entails scientific forecasts using historical data organized with timestamps. The process involves creating models based on past studies and using data to make observations and inform strategic decisions. MLP refers to a hierarchical structure of processing units that consists of two or more neuron layers that do not overlap. Multilayer sensory neural networks utilize multiple levels of processing power. MLP training utilizing the least MSE may fail to approximate a function owing to the lack of clarity in the data distribution. Errors in calculating Euclidean distance can impede the training of neural networks. The network was trained using an error back-propagation learning method that incorporated additional error measures. The gradient of the MSE was calculated for every input and output pair in each iteration by propagating the weighted sums through all layers. The weights of the first hidden layer were subsequently adjusted using the value of the gradient to propagate backward change. Backpropagation is an iterative learning approach used by MLP to adjust the weights of the network to minimize the cost function [31]. After the weighted sums were propagated through all layers in the iteration, the gradient of the MSE was calculated for all input and output pairs, as well as the gradient of the first hidden layer. The gradient descent was calculated using (2).

$$\Delta\omega(t) = -\epsilon \frac{dE}{d\omega(t)} + \alpha\Delta\omega(t-1) \tag{2}$$

Where $\Delta\omega(t)$ = The change in the weight ω at time step t. ϵ = Learning rate. $\frac{dE}{d\omega(t)}$ = Gradient of the error function E with respect to the weight $\omega(t)$. α = Momentum learning rate, typically a value between 0 and 1.

3.4. Convolutional neural network model implementation

In sensor-based prediction scenarios, such as monitoring and forecasting machinery health or environmental conditions, advanced deep learning approaches can dramatically improve predictive accuracy and dependability. CNN are especially well suited for this purpose because of their capacity to interpret raw time-series data from sensors and find detailed patterns and abnormalities that may indicate underlying difficulties or environmental changes. CNNs use a series of convolution and pooling layers to automatically and adaptively learn spatial hierarchies of features from input data. These layers collaborate to extract low-to high-level features, allowing the network to capture key characteristics of sensor data without requiring substantial manual feature engineering. This feature makes CNNs extremely efficient and effective in handling complicated, multidimensional sensor data. A CNN is a deep learning algorithm that takes input data, assigns weights and biases that are both learnable and differentiates between various aspects of the data. The CNN design comprises three essential layers: a convolution layer, a pooling layer, and a fully connected layer. Applying a filter or kernel to the input data in the convolution layer generally results in a reduction in the input size. Padding is used to preserve the original size of the input while enabling the extraction of low-level characteristics. The pooling layer acts as an intermediary between the convolution and fully connected layers. Pooling techniques decrease the spatial dimensions of the feature maps produced by the convolution layer. The decrease in dimensions to improves computational performance and mitigates over fitting by prioritizing the most significant characteristics. Ultimately, the results from the pooling layer are passed on to the fully linked layer, where the combined characteristics are utilized to make the final predictions or classifications. The hierarchical architecture of CNN enables efficient processing and learning from raw time-series data, making them highly favorable for jobs involving sequence categorization. CNN can learn directly from the raw input data. This allows them to construct internal representations of time series data and achieve a performance similar to that of models trained on artificial features [32], [33].

3.5. LSTM model implementation

LSTM is a specific form of RNN structure that addresses the shortcomings of regular RNNs in effectively capturing and utilizing long-term relationships in sequential data. LSTM can capture long-term dependencies, handle variable-length sequences, selectively learn and remember relevant features, and adapt to noisy and irregular data. In the LSTM architecture, memory cells play a pivotal role in retaining information over extended periods, thereby facilitating the capture of dependencies within sequential data. These cells possess the unique ability to selectively retain or discard information based on input signals. In addition, LSTM networks incorporate gating mechanisms to regulate the flow of information within a model. These gates include the forget gate, which determines the retention or discarding of information from the previous cell state; the input gate, which is responsible for incorporating new information into the cell state; and the output gate, which governs the information output to the subsequent time step. The cell state, which traverses horizontally across the network, acts as a conduit for transporting information across different time steps, thereby ensuring the preservation of valuable data over long sequences. Complementing the cell state, hidden state, or LSTM output serves as a filtered representation of the cell state containing pertinent information for the current prediction or classification task. Together, these components form the foundation of LSTM networks, enabling them to model and understand sequential data effectively [34]. The LSTM operations involved in the forget gate, output gate, hidden gate, and input gate are given by (3)–(6) respectively as:

$$f_t = \sigma(x_t U_t + H_{t-1} W_t) \tag{3}$$

$$o_t = \sigma(x_t U_t + H_{t-1} W_t) \tag{4}$$

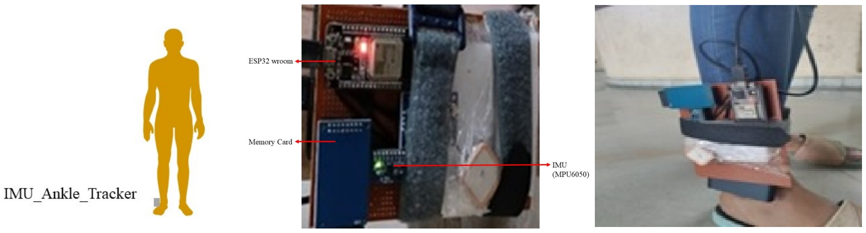
$$H_t = o_t \cdot \tanh(c_t) \tag{5}$$

$$i_t = \sigma(x_t U_t + H_{t-1} W_t) \tag{6}$$

Where W_t = Weight associated with the input, x_t = Input at the current time step, H_{t-1} = Hidden layer at the previous timestamp, and U_t = Weight associated with the hidden state.

3.6. Comparing neural network architectural design for forecasting

The hardware configuration used for sensor-based activity tracking is illustrated in Figure 3. Figure 3(a) shows the IMU ankle tracker mounted on the subject's leg for capturing motion data. Figure 3(b) presents the custom hardware module comprising various sensors used for data acquisition. Finally, Figure 3(c) demonstrates the subject wearing the complete hardware setup while performing various physical activities. The MLP, CNN, and LSTM structures and hyper-parameters were evaluated using the accelerometer, gyroscope datasets gathered for the six activities in indoor and outdoor environments using sensor data. The parameters for the three variants of neural networks used in this study are listed in Table 3. All models employed the Adam variant of stochastic gradient descent and optimized the MSE loss function. The regression models require metrics such as mean absolute error (MAE), MSE, RMSE, and coefficient of determination (R²) to assess accuracy, as it produce continuous output and require a difference between the expected and ground truth. The residuals, which represent the discrepancies between the actual values and the predictions made by the KF and neural network models, were examined. The effectiveness of the KF and the neural network model in estimating the 2D position and velocity was examined by analyzing these metrics and visuals.



(a) (b) (c)

Figure 3. Hardware setup and subject demonstration: (a) IMU ankle tracker mounted on the human body, (b) hardware module with sensors for data collection, and (c) subject wearing hardware for data collection for various activities

Table 3. Forecasting model parameters

Parameter	Forecasting model		
	MLP	CNN	LSTM
Input	3-axis accelerometer gyroscope data	3-axis accelerometer gyroscope data	3-axis accelerometer gyroscope data
Activation function	ReLU	ReLU	ReLU
Loss function for training	MSE	Softmax cross-entropy	Softmax cross-entropy with logic
Learning rate	0.001	0.001	0.0025
No. of hidden layers	1	1	1
Neurons in the first layer	128	256	128
Neurons in the second layer	64	128	64
Neurons in the third layer	32	32	32
Neurons in the fourth layer	128	128	128
Training epochs	20	25	50
Batch Size	128	10	1024

3.7. Model performance parameter

Accuracy assessment is the backbone of any deep-learning model. The choice of proper error measures determines the accuracy of the prediction model. For instance, in the regression analysis, the model's efficacy was measured using MSE, MAE, RMSE, and R². These parameters were used to compare the performance of the three different neural network variants in making the predictions. The MAE is the average absolute difference between the actual and projected values in a dataset. It measures the mean residuals in the dataset and calculates them as:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}| \tag{7}$$

Where y_i = mean value of velocity, position. $\hat{y}$ = predicted value of velocity, position. The square root of the MSE is the RMSE. It measures the distance of the residuals from the mean represented in (8):

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2} \tag{8}$$

Where y_i = actual value of velocity, position. $\hat{y}$ = predicted value of velocity, position. N = number of observations.

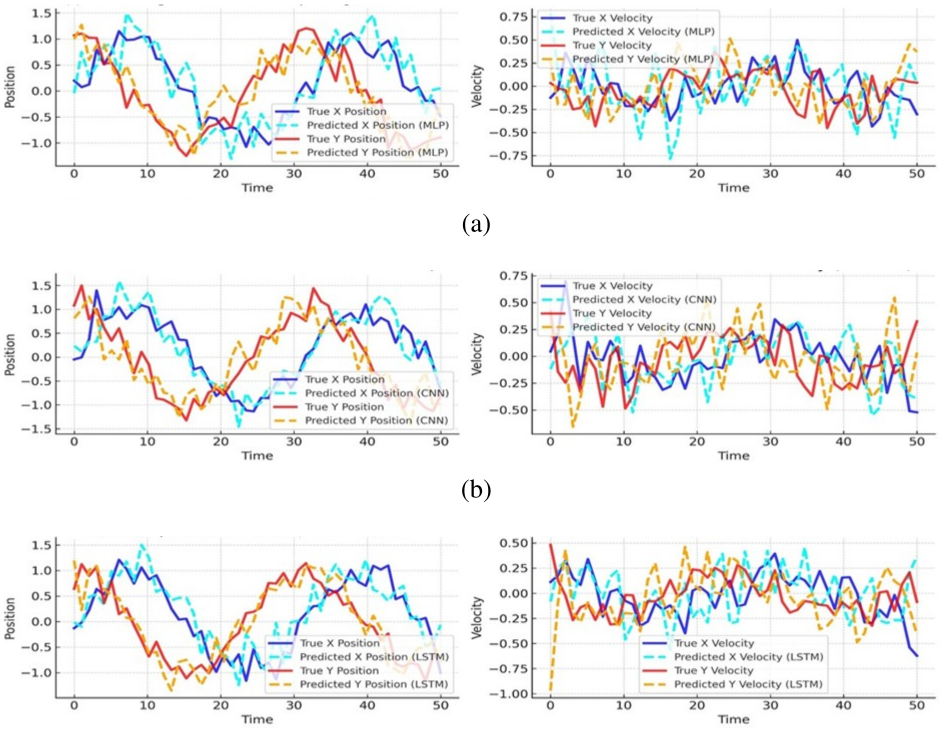
The R² represents the fraction of the variance in the dependent variable explained by the linear regression model. It is a scale-free score, so regardless of whether the numbers are low or high, R² will always be less than or equal to 1. The R² formula is:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y})^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$$

After assessing their performance, simulations show that for neural networks, an input vector is provided, the resulting feature map is processed through the ReLU activation function for nonlinearity, and a pooling operation is conducted to introduce translation invariance [35]–[37].

4. RESULTS AND DISCUSSION

A comparative analysis was conducted to evaluate the effectiveness of the state estimation techniques with and without deep learning models. Metrics such as MSE, RMSE, MAE, and R² were employed to gauge the accuracy by utilizing ground truth information. The visualization contributed to the analysis of the results, and the predicted position and velocity for outdoor activities using the neural network models are shown in Figures 4(a) to 4(c). These predictions were calculated using the most recent data gathered solely by MPU6050. In this plot, the comparison between the estimated 2D positions and velocities and the actual values indicates a strong performance of the KF-LSTM predictions that accurately plot the actual values.



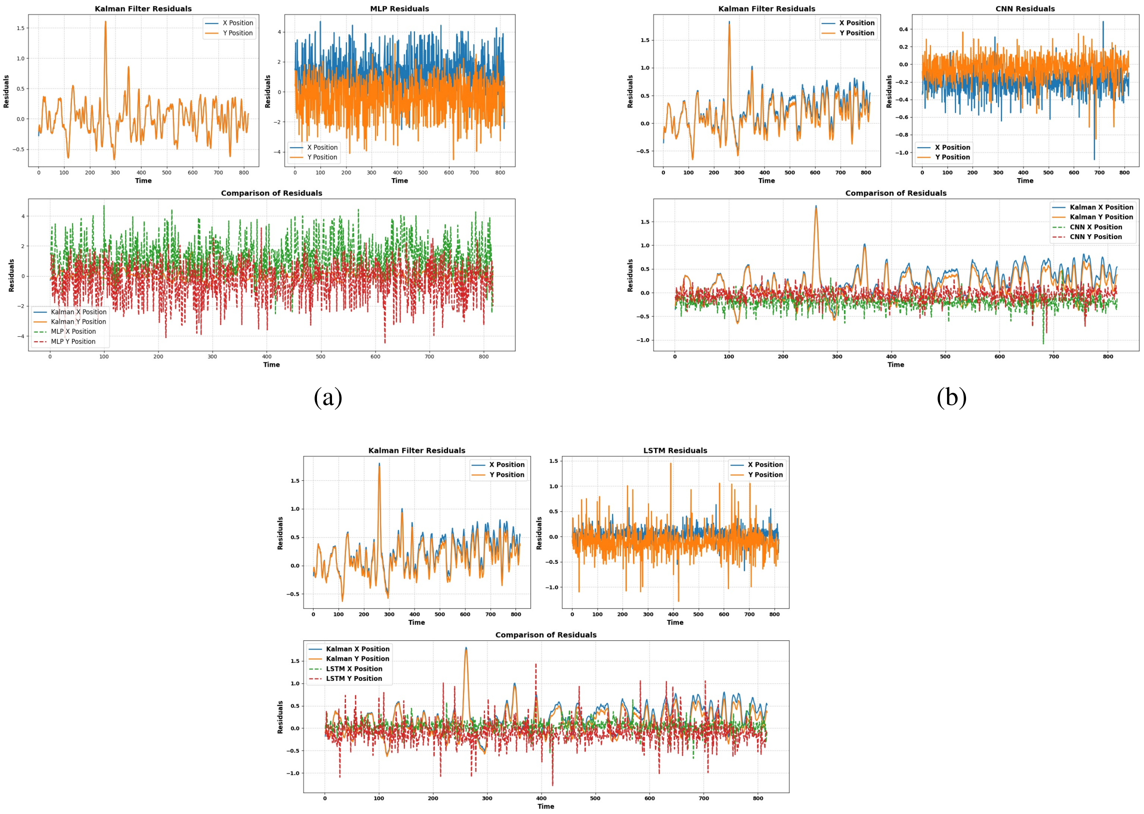
(a)

(b)

(c)

Figure 4. Predicted position and velocity values from: (a) MLP, (b) CNN, and (c) LSTM

In the context of all three neural network models, a residual plot is often used to assess the alignment between the model's forecasts and observed values. Residuals refer to discrepancies between the observed and estimated values. The residuals correspond to deviations between the anticipated and actual values. The residuals for both the KF and the neural network models were obtained by subtracting the anticipated positions and velocities from the actual positions and velocities. The residual plot displays the individual predictions of the KF and the features generated by the KF provided to the neural network models. The purpose of the residual plot is to evaluate the quality of the regression analysis fit, as shown in Figure 5(a) to 5(c) shows the KF predictions and comparison of KF and MLP, CNN and LSTM residuals in time-series forecasting. This comparative study confirms that LSTM outperforms both MLP and CNN in minimizing positional errors, making it a more robust choice for real-time localization and tracking applications.



(a)

(b)

(c)

Figure 5. Residual error plot of Kalman prediction and neural network model prediction for (a) MLP, (b) CNN, and (c) LSTM for a set of targets

The performance of the navigation system was evaluated based on the IMU data and the estimates were compared with the ground truth to determine accuracy, reliability, and efficiency, as shown in Table 4. The table presents a performance comparison of forecasting models that utilize various Kalman filtering techniques, namely KF, EKF, and SPKF. And the challenges posed by the time series forecasting is mentioned in the Table 5. The effectiveness of these models was evaluated on the basis of three metrics: MAE, RMSE, and R². The MLP model exhibited a modest performance, whereas the CNN model demonstrated larger error values. The LSTM model outperformed the MLP and CNN models, effectively capturing data patterns. The EKF model demonstrated a satisfactory level of accuracy, whereas the SPKF model exhibited superior performance, characterized by a smaller MAE and a higher R². LSTM models have improved performance compared to MLP and CNN models in various configurations, particularly when used in conjunction with the KF and SPKF. This suggests that LSTM models are particularly useful for handling temporal dependencies in the data. LSTM performed better than MLP and CNN in this category, with a MAE of 5.70, a RMSE of 1.31

and a positive R² of 0.5. However, the SPKF-MLP and SPKF-LSTM combinations showed the best overall performance with the lowest errors and the highest R² values, indicating excellent fit and predictive accuracy. Compared to the initial study [38], the performance of sensor-based state estimators, such as KF, EKF, and SPKF, was also evaluated and improved results were demonstrated in terms of velocity, RMSE, and finally position as shown in Table 6.

Table 4. Comparison of forecasting models: MLP, CNN, and LSTM

Model performance	KF with forecasting models			EKF with forecasting models			SPKF with forecasting models		
	MLP	CNN	LSTM	MLP	CNN	LSTM	MLP	CNN	LSTM
MAE	2.8	3.0	1.3	7.24	8.9	5.7	0.3	2.1	0.4
RMSE	5.6	9.9	4.4	2.12	6.2	1.91	1.2	1.4	1.03
R^2	-2.7	-1.1	0.7	0.8	-1.91	0.5	0.9	-1.8	0.9

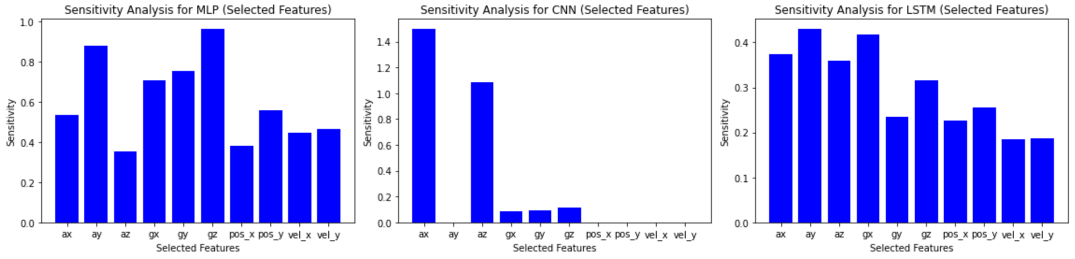
Table 5. Challenges and parameters in time series forecasting

Challenges in time series forecasting	Parameters
Data quality	Intermittent data, sparsity, new time series, absent time stamps, and gaps
Data latency	The frequency of historical data received by forecasting models can extend the forecasting horizon and diminish the ability to capture recent effects with auto regressive components
Predict predictors as inputs	Modeling predictors for target variables requires inputs for production forecasting.
Retraining monitoring frequency	Forecasters should decide whether to retrain models frequently or in reaction to feature drift or performance decline.

Table 6. Comparison of RMSE and RMS from other articles and the proposed method

Article	Method	RMSE	RMS
Jamil *et al.* [39]	KF, ANN	2.5	6.388
Kim *et al.* [40]	MARG, CNN	4.1	-
Ribeiro *et al.* [38]	LSTM	1.9	-
Proposed method	EKF-LSTM	1.91	0.3
Proposed method	SPKF-LSTM	1.03	1.5

The benchmark model was compared to obtain overall performance indicators. The proposed strategy produces improved results not only in terms of location accuracy, but also in sensor data forecasting. Using both extended Kalman filtering and time-series forecasting, collaborative forecasting can optimize the benefits of both strategies, resulting in a better performance measured by the RMSE, which is the difference between the predicted and observed values after new measurements are considered. This approach also improves forecast accuracy, especially in scenarios with noisy or missing measurements. Forecasting models also require sensitivity analysis to investigate the impact of input features on the output of a model. The objective was to comprehend the impact of alterations in the input features on the model predictions. This study identifies features that have a large impact on a model's output and those that have a smaller impact. Sensitivity plots or charts display the results of the sensitivity analysis. Input perturbations are employed to calculate the sensitivity, which involves deliberately altering the input features and analyzing the effect on the model's output. Visualization of sensitivity involves showing the impact of variations in each feature on the model predictions. The training and validation of the sensitivity are specified for all neural network algorithms in Figures 6(a) to 6(c).



(a) (b) (c)

Figure 6. Sensitivity analysis of selected features for different models: (a) MLP, (b) CNN, and (c) LSTM

5. CONCLUSION

This study contributes to the field of time-series forecasting for multi-input data by introducing innovative methodologies that combine state-of-the-art state-estimation techniques with neural-network approaches. This study evaluates the efficacy of three unique time-series forecasting frameworks (MLP, CNN, and LSTM) and compares their performance. This study demonstrates that incorporating Kalman-based state estimators with time-series forecasting models enhances the accuracy and stability of predictions in dynamic navigation and localization tasks. The presented models exhibit promising results in accurately predicting the location and speed of various activities using IMU sensor data. These outcomes substantially contribute to the field of dynamic navigation and activity monitoring. IMU sensors are employed in time-series prediction to assess trends in historical data, enabling more informed decision making. Effectively utilizing IMU sensors can facilitate the collection of relevant sensor data for precise navigation-state predictions. Initially, a feature extractor was trained using a dataset, and the neural network-based regression produced by the feature extractor was evaluated using a dataset collected under more realistic conditions. The study found that CNN and MLP-based approaches could not compete satisfactorily with the best, whereas the LSTM-based approach outperformed the other approaches. The experiment provided a comprehensive synopsis of the application of LSTM to examine the impact of critical hyperparameters on the differentiation of target activities within the feature space. Pretrained neural network models are expected to be used more extensively, similar to deep-learning applications. Comparative research revealed that the features derived using the KF perform excellently in sensor identification. Specifically, a single LSTM model achieves a prediction accuracy of 1.91 RSME when using the EKF filter and 1.03 RSME when using SPKF. The study also suggests that future research should aim to emphasize a more extensive and varied dataset, incorporating diverse environmental settings, user behaviors, and sensor conditions to improve generalizability. Additionally, projecting user activity data coordinates onto a digital map can serve as a real-time navigation aid, enabling enhanced situational awareness and route optimization. Future research focus on using more diverse datasets and mapping user activity coordinates to enhance navigation and spatial awareness

FUNDING INFORMATION

The authors did not receive financial support for the conduction of this research.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Ashvini Kulkarni	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓			
Augusta Sophy Beulet Paul	✓	✓		✓		✓				✓		✓	✓	

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal Analysis

I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting

Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject Administration
Fu : **Fu**nding Acquisition

CONFLICT OF INTEREST STATEMENT

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

ETHICS APPROVAL AND CONSENT TO PARTICIPATE

The authors have obtained the necessary permissions from the institutional ethics committee to con duct this work. Consent was obtained from all participants involved in the study.

DATA AVAILABILITY

The datasets generated and analyzed during the current study are available from the corresponding author upon reasonable request.

REFERENCES

[1] H. Koyuncu and S. H. Yang, ”A survey of indoor positioning and object locating systems,” *IJCSNS International Journal of Computer Science and Network Security*, vol. 10, no. 5, pp. 121–128, 2010.

[2] P. A. S. B. Ashvini Kulkarni, “Sensor-based adaptive estimation in a hybrid environment employing state estimator filters,” *Intelligent Automation & Soft Computing*, vol. 37, no. 1, pp. 127–146, 2023, doi: 10.32604/iasc.2023.035144.

[3] R. Harle, “A survey of indoor inertial positioning systems for pedestrians,” *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1281–1293, 2013, doi: 10.1109/SURV.2012.121912.00075.

[4] W. Zhang, X. Li, D. Wei, X. Ji, and H. Yuan, “A foot-mounted PDR system based on IMU/EKF+HMM+ZUPT+ZARU+HDR+compass algorithm,” in *2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, Sep. 2017, pp. 1–5, doi: 10.1109/IPIN.2017.8115916.

[5] M. Kwak, Y. Park, J. Kim, J. Han, and T. Kwon, “An energy-efficient and lightweight indoor localization system for internet-of-things (IoT) environments,” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 2, no. 1, pp. 1–28, Mar. 2018, doi: 10.1145/3191749.

[6] X. He, D. Aloi, and J. Li, “Probabilistic multi-sensor fusion based indoor positioning system on a mobile device,” *Sensors*, vol. 15, no. 12, pp. 31464–31481, Dec. 2015, doi: 10.3390/s151229867.

[7] M. Kozlowski, D. Byrne, R. Santos-Rodriguez, and R. Piechocki, “Data fusion for robust indoor localisation in digital health,” in *2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, IEEE, Apr. 2018, pp. 302–307, doi: 10.1109/WCNCW.2018.8369009.

[8] S. Han and J. Wang, “A novel method to integrate IMU and magnetometers in attitude and heading reference systems,” *Journal of Navigation*, vol. 64, no. 4, pp. 727–738, Oct. 2011, doi: 10.1017/S0373463311000233.

[9] G. Girard, S. Côté, S. Zlatanova, Y. Barette, J. St-Pierre, and P. Van Oosterom, “Indoor pedestrian navigation using foot-mounted IMU and portable ultrasound range sensors,” *Sensors*, vol. 11, no. 8, pp. 7606–7624, Aug. 2011, doi: 10.3390/s110807606.

[10] P. Mirowski, T. K. Ho, S. Yi, and M. MacDonald, “SignalSLAM: Simultaneous localization and mapping with mixed WiFi, Bluetooth, LTE and magnetic signals,” in *International Conference on Indoor Positioning and Indoor Navigation*, IEEE, Oct. 2013, pp. 1–10, doi: 10.1109/IPIN.2013.6817853.

[11] S. Yamagishi and L. Jing, “Pedestrian dead reckoning with low-cost foot-mounted IMU sensor,” *Micromachines*, vol. 13, no. 4, Apr. 2022, doi: 10.3390/mi13040610.

[12] D. Xie, J. Jiang, P. Yan, J. Wu, Y. Li, and Z. Yu, “A novel three-dimensional positioning method for foot-mounted pedestrian navigation system using low-cost inertial sensor,” *Electronics*, vol. 12, no. 4, Feb. 2023, doi: 10.3390/electronics12040845.

[13] R. Wu *et al.*, “IOAM: A novel sensor fusion-based wearable for localization and mapping,” *Remote Sensing*, vol. 14, no. 23, Nov. 2022, doi: 10.3390/rs14236081.

[14] X. Guang, Y. Gao, P. Liu, and G. Li, “IMU data and GPS position information direct fusion based on LSTM,” *Sensors*, vol. 21, no. 7, 2021, doi: 10.3390/s21072500.

[15] Z. Yang, H. Jing, B. Kuang, S. Zhong, J. Qin, and Y. Tong, “GPS and lidar fusion positioning based on unscented kalman filtering and long short-term memory cosidering gps failure,” in *2021 IEEE 4th International Conference on Automation, Electronics and Electrical Engineering (AUTEEE)*, 2021, pp. 18–23, Shenyang, China, 2021, pp. 18-23, doi: 10.1109/AUTEEE52864.2021.9668630.

[16] M. Ok, S. Ok, and J. H. Park, “Estimation of vehicle attitude, acceleration, and angular velocity using convolutional neural network and dual extended kalman filter,” *Sensors*, vol. 21, no. 4, Feb. 2021, doi: 10.3390/s21041282.

[17] J. P. S. D. M. Lima, H. Uchiyama, and R. Taniguchi, “End-to-end learning framework for IMU-based 6-DOF odometry,” *Sensors*, vol. 19, no. 17, Aug. 2019, doi: 10.3390/s19173777.

[18] A. Vagale, L. Šteina, and V. Vēciņš, “Time series forecasting of mobile robot motion sensors using LSTM networks,” *Applied Computer Systems*, vol. 26, no. 2, pp. 150–157, Dec. 2021, doi: 10.2478/acss-2021-0018.

[19] A. Poulose, O. S. Eyobu, and D. S. Han, “An indoor position-estimation algorithm using smartphone IMU sensor data,” *IEEE Access*, vol. 7, pp. 11165–11177, 2019, doi: 10.1109/ACCESS.2019.2891942.

[20] R. Xu, W. Chen, Y. Xu, and S. Ji, “A new indoor positioning system architecture using GPS signals,” *Sensors*, vol. 15, no. 5, pp. 10074–10087, Apr. 2015, doi: 10.3390/s150510074.

[21] F. Bashir and H.-L. Wei, “Handling missing data in multivariate time series using a vector autoregressive model-imputation (VAR-IM) algorithm,” *Neurocomputing*, vol. 276, pp. 23–30, Feb. 2018, doi: 10.1016/j.neucom.2017.03.097.

[22] T. Lin, M. Wang, M. Yang, and X. Yang, “A hidden Markov ensemble algorithm design for time series analysis,” *Sensors*, vol. 22, no. 8, Apr. 2022, doi: 10.3390/s22082950.

[23] L. Yang and A. Shami, “On hyperparameter optimization of machine learning algorithms: Theory and practice,” *Neurocomputing*, vol. 415, pp. 295–316, 2020, doi: 10.1016/j.neucom.2020.07.061.

[24] A. Mannini and A. M. Sabatini, “Machine learning methods for classifying human physical activity from on-body accelerometers,” *Sensors*, vol. 10, no. 2, pp. 1154–1175, Feb. 2010, doi: 10.3390/s100201154.

[25] J. G. De Gooijer and R. J. Hyndman, “25 years of time series forecasting,” *International Journal of Forecasting*, vol. 22, no. 3, pp. 443–473, 2006, doi: 10.1016/j.ijforecast.2006.01.001.

[26] E. U. Warriach and K. Tei, “A comparative analysis of machine learning algorithms for faults detection in wireless sensor networks,” *International Journal of Sensor Networks*, vol. 24, no. 1, pp. 1–13, 2017, doi: 10.1504/IJSNET.2017.084209.

[27] V. Prema and K. U. Rao, “Time series decomposition model for accurate wind speed forecast,” *Renewables: Wind, Water, and Solar*, vol. 2, no. 1, Dec. 2015, doi: 10.1186/s40807-015-0018-9.

[28] D. S. D. O. S. Júnior, J. F. L. D. Oliveira, and P. S. G. D. M. Neto, “An intelligent hybridization of ARIMA with machine learning models for time series forecasting,” *Knowledge-Based Systems*, vol. 175, pp. 72–86, Jul. 2019, doi: 10.1016/j.knosys.2019.03.011.

[29] T. Koutroumanidis, K. Ioannou, and G. Arabatzis, “Predicting fuelwood prices in Greece with the use of ARIMA models, artificial neural networks and a hybrid ARIMA–ANN model,” *Energy Policy*, vol. 37, no. 9, pp. 3627–3634, Sep. 2009, doi: 10.1016/j.enpol.2009.04.024.

[30] N. Kimura, I. Yoshinaga, K. Sekijima, I. Azechi, and D. Baba, “Convolutional neural network coupled with a transfer-learning approach for time-series flood predictions,” *Water*, vol. 12, no. 1, Dec. 2019, doi: 10.3390/w12010096.

[31] S. Indolia, A. K. Goswami, S. P. Mishra, and P. Asopa, “Conceptual understanding of convolutional neural network- a deep learning approach,” *Procedia Computer Science*, vol. 132, pp. 679–688, 2018, doi: 10.1016/j.procs.2018.05.069.

[32] A. Apicella, F. Donnarumma, F. Isgrò, and R. Prevete, “A survey on modern trainable activation functions,” *Neural Networks*, May 2020, doi: 10.1016/j.neunet.2021.01.026.

[33] G. Lin and W. Shen, “Research on convolutional neural network based on improved Relu piecewise activation function,” *Procedia Computer Science*, vol. 131, pp. 977–984, 2018, doi: 10.1016/j.procs.2018.04.239.

[34] N. K. Manaswi, “RNN and LSTM,” in *Deep Learning with Applications Using Python*, Berkeley, CA: Apress, 2018, pp. 115–126, doi: 10.1007/978-1-4842-3516-4_9.

[35] M. Shiblee, P. K. Kalra, and B. Chandra, “Time series prediction with multilayer perceptron (MLP): A new generalized error based approach,” *Advances in Neuro-Information Processing*, 2009, pp. 37–44, doi: 10.1007/978-3-642-03040-6_5.

[36] C. Gao, J. Yan, S. Zhou, B. Chen, and H. Liu, “Long short-term memory-based recurrent neural networks for nonlinear target tracking,” *Signal Processing*, vol. 164, pp. 67–73, Nov. 2019, doi: 10.1016/j.sigpro.2019.05.027.

[37] S. Feng, X. Li, S. Zhang, Z. Jian, H. Duan, and Z. Wang, “A review: state estimation based on hybrid models of Kalman filter and neural network,” *Systems Science & Control Engineering*, vol. 11, no. 1, Dec. 2023, doi: 10.1080/21642583.2023.2173682.

[38] P. M. S. Ribeiro, A. C. Matos, P. H. Santos, and J. S. Cardoso, “Machine learning improvements to human motion tracking with IMUs,” *Sensors*, vol. 20, no. 21, Nov. 2020, doi: 10.3390/s20216383.

[39] F. Jamil, N. Iqbal, S. Ahmad, and D.-H. Kim, “Toward accurate position estimation using learning to prediction algorithm in indoor navigation,” *Sensors*, vol. 20, no. 16, Aug. 2020, doi: 10.3390/s20164410.

[40] S. U. Kim, J. Lee, J. Yoon, S. Ko, and J. Kim, “Robust methods for estimating the orientation and position of IMU and MARG sensors,” *Electronics Letters*, vol. 57, no. 21, pp. 816–818, Oct. 2021, doi: 10.1049/ell2.12263.

BIOGRAPHIES OF AUTHORS

Ashvini Kulkarni holds a masters in communication engineering as well as a Bachelor of Engineering in Electronics. She currently works as an assistant professor at the Department of Electronics and Telecommunications of the International Institute of Information Technology in Pune, Maharashtra. She is doing a Ph.D. at Vellore Institute of Technology in Chennai and is interested in embedded systems, IoT, and AI. She can be contacted at email: ashvini.kulkarni11@gmail.com.

Augusta Sophy Beulet Paul being a lecturer for more than 25 years, she has developed a passion towards teaching Engineering subjects and guiding projects. The passion has naturally been extended towards developing the students too. She handled more than 20 different subjects in electronics and communication engineering and guided more than 50 projects in UG and PG levels. Her major contribution as 24 research papers in international conferences and journals. She did her research work in the field of VLSI Signal Processing in Anna University, in 2015. Received her M.E. degree from Anna University and finished her B.E. degree in Government College of Engineering, Tirunelveli. Now she is working as Associate Professor in VIT University, Chennai. She is the corresponding author of this paper. She can be contacted at email: augustasophyt@vit.ac.in.