36th International Technical Meeting of the Satellite Division of The Institute of Navigation,
11-15 September 2023, Denver, USA, pp. 2624-2643
DOI:10.33012/2023.19271

A Hybrid Deep Learning Approach for Robust Multi-Sensor GNSS/INS/VO Fusion in Urban Canyons

Patrick Geragersian, Ivan Petrunin, Weisi Guo, *Cranfield University*,
Raphael Grech, *Spirent PLC*

BIOGRAPHIES

Patrick Geragersian is a researcher at Cranfield University with a focus on mechanical engineering and digital innovation. He previously worked at Atkins (SNC-Lavalin) on the decommissioning team, where he was involved in developing virtual reality (VR) and augmented reality (AR) technologies for stakeholder engagement, along with automating complex engineering processes. Holding a master's degree in aerospace engineering from the University of Manchester, Patrick's current research at Cranfield involves alternative PNT (position, navigation, and timing) technologies for UAV navigation. His work includes integrating new PNT sources and machine learning algorithms, aiming to enhance accuracy in the rapidly growing UAV industry.

Dr. Ivan Petrunin received an M.Sc. degree in design and manufacturing of electronic equipment from the National Technical University of Ukraine, in 1998, and a Ph.D. degree in signal processing for condition monitoring from Cranfield University, U.K., in 2013. He is currently a Reader in digital signal processing with the Centre for Autonomous and Cyber-Physical Systems, Cranfield University. His current research interests include applied signal processing for autonomous systems with particular emphasis on sensor technologies, perception, data and information fusion, and decision-making for Cyber-Physical Systems.

Professor Weisi Guo received the M.Eng., M.A., and Ph.D. degrees from the University of Cambridge. He was an Associate Professor at the University of Warwick. He is currently the Chair Professor of human–machine intelligence at Cranfield University. He is also a Turing Fellow with The Alan Turing Institute. He has published more than 130 articles and is PI on over $3.5m of research grants from EPSRC, H2020, Royal Society, Innovate U.K., and DSTL. His research has won several international awards, including IET Innovation in 2015 and Bell Labs Prize Finalist in 2014, and Semi-Finalist in 2016 and 2019

Dr. Raphael Grech, with a doctorate in robot vision from Kingston University, is a Thought Leader, Chartered Engineer, and Research Scientist with over 20 years of experience. He provides Technical Leadership, Guidance, and Direction in Innovation Strategy, Business Development, Planning, and Project Management within emerging fields. His expertise spans areas such as Autonomous Vehicles, Digital Engineering, Advanced Robotics, Computer Vision, Sensing and Perception, Machine Learning, and Artificial Intelligence. With a focus on theoretical and practical aspects, he contributes to advancing these rapidly evolving technologies.

ABSTRACT

This paper addresses the significant challenges of robust autonomous navigation in Unmanned Aerial Vehicles (UAVs) within densely populated environments. The focus is on enhancing the performance of Position, Navigation, and Timing (PNT), as specified by the International Civil Aviation Organization, in terms of accuracy, integrity, continuity, and availability. The novel contribution introduces a Robust Multi-Sensor Fusion Architecture (RMSFA) that utilizes a Bayesian-LSTM machine learning algorithm, fusing GNSS, INS, and monocular odometry. Unlike existing solutions that rely on sensor redundancies or methods such as Receiver Autonomous Integrity Monitoring (RAIM), which have limitations in performance, or adaptability to erroneous signals, the proposed system offers improvements in both positioning accuracy and integrity. Furthermore, GNSS data is preprocessed to remove None-Line-of-Sight data (NLOS) to improve positioning accuracy. Additionally, INS data errors are corrected using a GRU-based error correction architecture to improve INS positioning and reduce drifting. The addition of these post-processing steps reduced the 95th percentile horizontal error by 97.4% and 71.5% respectively. A CNN-LSTM architecture is used to obtain a Visual Odometer (VO) from the camera sensor. The Bayesian-LSTM architecture fusion performance was then compared to a GNSS/IMU/VO EKF-GRU architecture. The comparison showed a 95th percentile error improvement in the horizontal direction of 30.1% for the Bayesian-LSTM. The architecture was tested in a realistic simulated environment utilizing Unreal Engine and AirSim for UAV simulation, Spirent GNSS7000 simulator for Hardware-in-the-Loop (HIL) simulation, and OKTAL-SE Sim3D to mimic the effects of multipath on GNSS signals. Overall, this work represents a step toward improving the safety and effectiveness of drone navigation by providing

a more robust navigation system suitable for safety-critical situations, without the stated disadvantages in previously mentioned literatures.

1. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) have the potential to revolutionize various industries, from delivery services to emergency response, but one of the main challenges in their widespread adoption is the ability to ensure robust autonomous navigation in densely populated urban environments (Kuang, Wu, Pan, & Zhou, 2020). Position, Navigation, and Timing (PNT) solutions must be accurate and have high integrity, continuity, and availability to meet the safety requirements set by the International Civil Aviation Organization (ICAO). Integrity, in particular, is a key performance consideration for UAVs operating in low-altitude urban environments, where error tolerances are tighter than in open-sky conditions (Isik, Hong, Petrunin, & Tsourdos, 2020). It is defined as the measure of trust that can be placed in the correctness of the navigation system's information and can be characterized by four parameters: Alert Limit (AL), Integrity Risk (IR), Time to Alert (TTA), and Protection Level (PL). PL is defined as the statistical bound errors computed to guarantee the probability of the position error being smaller than or equal to the targeted integrity. Examples of these are shown in Figure 1 (Zabalegui, et al., 2020) (Zhu, Marais, Betaille, & Berbineau, 2018).

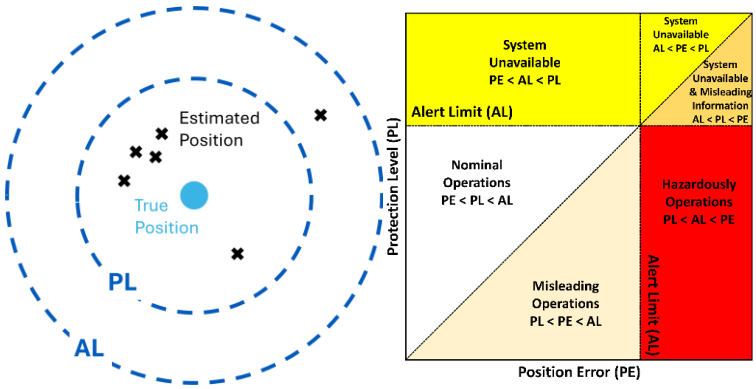

Figure 1: Illustration of the Relationship between Estimated, True Position, and Required Integrity Levels.

One of the main issues facing autonomous navigation in UAVs is the reliability of the sensors used for PNT. Global Navigation Satellite Systems (GNSS) such as GPS are the primary source of positioning information for many UAVs, but they are prone to errors and can be disrupted by various factors such as signal interference and atmospheric conditions (Geragersian, Petrunin, Guo, & Grech, Multipath Detection from GNSS Observables Using Gated Recurrent Unit, 2022). Inertial Measurement Units (IMUs) can provide complementary information to GNSS, but they are subject to drift over time and are not able to provide absolute position information (Unsal & Demirbas, 2012).

Another issue that can affect the accuracy and integrity of GNSS such as GPS is multipath propagation. Multipath occurs when the direct signal from the satellite is reflected or refracted off nearby objects before it reaches the receiver, causing the received signal to be delayed and distorted. This can result in errors in the position and timing estimates obtained from the GNSS receiver (Zhang & Hsu, 2018). Multipath propagation conditions are more likely to occur in urban environments or other areas with a lot of reflective surfaces, such as buildings and bridges. It can also be caused by atmospheric conditions, such as ionospheric and tropospheric delays, which can affect signal propagation and cause errors in the position estimate (Su, Jin, & Hoque, 2019).

One type of error that can affect IMUs is bias error, which is caused by the drift of the sensor's output over time. Bias error can be caused by a variety of factors, such as temperature changes, aging of the sensor, and external forces acting on the sensor (Altinoz & Unsal, 2014). It can result in a systematic offset in the measured acceleration and angular velocity, which can accumulate and cause errors in the position and orientation estimates. Another type of error that can affect IMUs is scale factor error, which is caused by variations in the sensitivity of the sensor's output. Scale factor error can result in errors in the measured acceleration and angular velocity, which can affect the accuracy of the position and orientation estimates (Geragersian, Petrunin, Guo, & Grech, An INS/GNSS fusion architecture in GNSS denied environment using gated recurrent unit, 2021).

To address these issues, many approaches have focused on fusing GNSS and IMU data to improve the accuracy and integrity of PNT solutions. Traditional fusion methods include extended and unscented Kalman filters, which require prior knowledge of the system models and assumptions about noise characteristics (Hu, Wang, Zhong, Gao, & Gu, 2018). These methods have shown some success in improving navigation performance, but they are not without their limitations. For example, they may not be able to accurately model complex non-linear dynamics or cope with large errors in the initial state estimate. In addition, they may not be able to detect and mitigate slow-growing failures, such as the drift in IMU measurements over time (Elsanhoury, Koljonen, Valisuo, Elmusrati, & Kuusniemi, 2021).

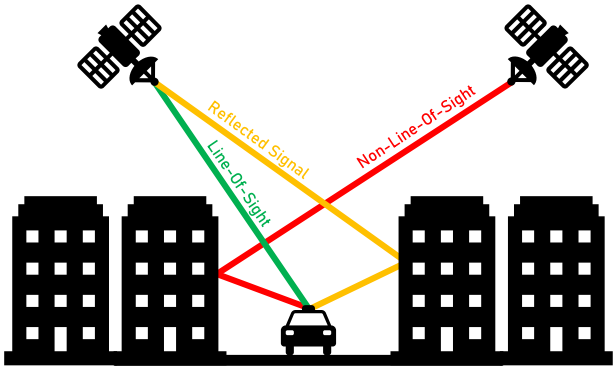

Figure 2: High-Level Concept of Urban Canyon Effects on GNSS Signals

Another approach to improving navigation integrity is through the use of Receiver Autonomous Integrity Monitoring (RAIM) algorithms, which work by classifying GNSS signals as being faulty or not. However, these methods are only effective in situations where most signals are not erroneous and may not be suitable for detecting slow-growing failures (Bhattacharyya & Gebre-Egziabher, 2015).

An additional approach is the use of integrity monitoring techniques that are based on the analysis of the innovation of the Kalman filter. One example of this is the Residuals Chi-square Test Method (RCTM), which is suitable for integrity monitoring of GNSS/Strapdown Inertial Navigation System (SINS) fusion systems. RCTM is based on the analysis of the innovation of the GNSS/SINS Kalman filter, where depending on the χ^2 test statistic, the system will determine if there is a "no-failure" or a "failure" system (Zhu & meng, 2018). However, this approach is not suitable for slow-growing ramp failures such as random walks that, as noisy sensor data is integrated over time, will cause drifting positioning errors (Liu, Zheng, Wang, & Feng, 2010).

A proposed architecture in literature is based on a multi-sensor fusion of GPS/IMU/LIDAR using a multi-sliding window classification adaptive unscented Kalman filter (M-SWCAUKF). This approach can improve the integrity of the navigation solution by continuously adapting to changing conditions and utilizing the complementary information provided by multiple sensors (Zheng, Fu, Li, & Yuan, 2018).

One approach is the use of map-matching techniques, which involves comparing the vehicle's estimated position to a pre-defined map of the environment. By using map constraints and other information, such as the vehicle's heading and speed, it is possible to improve the accuracy and integrity of the navigation solution. Map-matching techniques can be used in conjunction with other sensors, such as GNSS and IMU, to further improve the performance of the navigation system (Tao, et al., 2022).

In this paper, we present a new approach to improving navigation integrity in UAVs operating in urban environments. By leveraging machine learning techniques, our proposed system can continuously monitor and adapt to changing conditions in real time, providing reliable position and velocity information with high integrity. Our approach detects and mitigates both sudden and slow-growing failures, making it suitable for a wide range of mission scenarios.

The effectiveness of this proposed approach will be demonstrated through simulation using Spirent SimGEN, Sim3D, and GSS7000 simulator. The objectives of the research include reviewing the current state-of-the-art in autonomous navigation for UAVs, identifying the limitations of current approaches, proposing a new machine learning-based approach, implementing, and evaluating the approach through simulation. The research will be organized into four chapters: an introduction and literature review, a description of the proposed approach and implementation, an evaluation, and a conclusion.

2. ROBUST MULTI-SENSOR FUSION ARCHITECTURE (RMSFA)

2.1 GATED RECURRENT UNITS

Gated Recurrent Units (GRUs) are a variant of Recurrent Neural Networks (RNNs) introduced by Cho et al. in 2014 (Cho, Merrienboer, Bahdanau, & Bengio, 2014). GRUs were developed as a more computationally efficient alternative to Long Short-Term Memory (LSTM) networks, retaining the capacity to model long-range dependencies in sequential data while utilizing fewer parameters.

The architecture of a GRU consists of recurrent hidden units that include two specific gating mechanisms: the reset gate and the update gate. These gates modulate the flow of information within the network, allowing for the selective retention and forgetting of information across time steps.

1. Reset Gate: This gate determines the degree to which previous hidden states are considered when computing the current hidden state. It is governed by a sigmoid activation function, generating values between 0 and 1 that weigh the contribution of previous hidden states.
2. Update Gate: Operating similarly to the reset gate, the update gate dictates the extent to which the previous hidden state is carried over to the current time step. It acts as a blend between the previously hidden state and a candidate's hidden state, thus controlling the flow of information.

Mathematically, the operations within a GRU are expressed as follows:

$$r_t = \sigma(W_r \,.\, [h_{t-1}, x_t] + b_r)$$

$$z_t = \sigma(W_z \,.\, [h_{t-1}, x_t] + b_z)$$

$$\widetilde{h_t} = \tanh\,(W_h.[r_t \,.\, h_{t-1}, x_t] + b_h$$

$$h_t = (1 - z_t).h_{t-1} + z_t.\widetilde{h_t}$$

Here, r_t and z_t represent the reset and update gate, respectively. The weight matric W and bias vector b are the learnable parameter, σ denotes the sigmoid activation function and $tanh$ is the hyperbolic tangent.

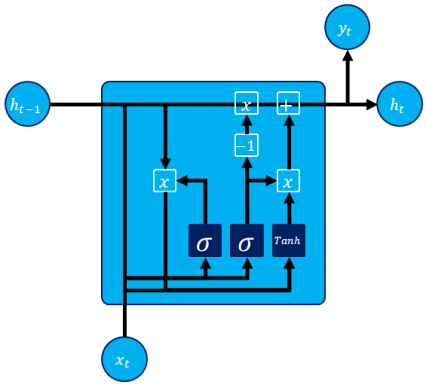

Figure 3: Inside a Gated Recurrent Unit Neural Network Cell

The GRU's simplified architecture compared to the LSTM reduces computational overhead and the number of hyperparameters. Despite this reduction in complexity, GRUs have been found to perform competitively with LSTMs in various tasks such as language modeling, speech recognition, sequence-to-sequence translation and sensor fusion (Xu, Petrunin, & Tsourdos, 2022). One potential drawback of the GRU's design is the possible loss of modeling capacity due to the reduced number of gates, particularly in scenarios that require highly nuanced control over the information flow. The choice between LSTMs and GRUs thus often hinges on the specific demands of the task, the computational resources available, and the nature of the sequential data being processed.

2.2 BAYESIAN-LSTM

LSTM networks are a form of RNNs designed to model sequential data by capturing long-range dependencies. The structure includes input, output, and forget gates, along with a continuous-valued cell state, allowing the network to learn and retain complex temporal relationships within sequences. Integrating Bayesian principles into LSTMs introduces a probability distribution over the model's weights, instead of point estimates. This probabilistic treatment offers a nuanced representation of uncertainty in the model's predictions and parameters, enhancing its robustness and generalization capabilities.

Three principal techniques are commonly applied to realize Bayesian treatment within LSTMs: Markov Chain Monte Carlo (MCMC), Monte Carlo (MC) Dropouts, and Variational Inference (VI). MCMC provides an unbiased estimation of the posterior through a sequence of samples, although it may suffer from slow convergence and high computational cost. MC Dropouts implement dropout layers during both training and prediction, interpreting them as approximate Bayesian inference, which provides a balance between computational efficiency and uncertainty quantification. VI approximates the true posterior with a tractable distribution and offers scalability but may miss subtle characteristics of the posterior (Geragersian, Petrunin, Guo, & Grech, Uncertainty-based Sensor Fusion Architecture using Bayesian-LSTM Neural Network, 2023).

For this paper, MC Dropouts are chosen due to their advantageous balance between accuracy and computational demands. Unlike MCMC, which can be prohibitively slow, or VI, which may require a careful specification of the approximating family, MC Dropouts offer an appealing compromise. The mathematics of MC Dropouts in a neural network context can be described by sampling from the approximate posterior. Given an input x, the predictive distribution is obtained through T stochastic forward passes, each with dropout applied:

$$p(y|x, D) \approx \frac{1}{T}\sum_{t=1}^{T} p(y|x, \theta_t)$$

Here, θ_t represents the weights in the t^{th} forward pass without dropout, and D symbolizes the data. This approach allows Bayesian-LSTMs with MC Dropouts to provide a robust and computationally amenable framework for sequential modeling, preserving the strengths of traditional LSTM networks while incorporating essential uncertainty quantification. By leveraging the advantages of dropouts within a Bayesian framework, MC Dropouts present a practical and effective method, aligned with the specific demands and objectives of the study at hand.

2.3 CONVOLUTIONAL NEURAL NETWORK – LSTM

The Convolutional Neural Network-LSTM (CNN-LSTM) model brings together the spatial feature extraction capabilities of CNNs with the temporal sequence modeling strengths of LSTMs, making it particularly suited for applications like video analysis, time-series prediction where spatial characteristics matter, and multivariate sequential data with spatial features. In the CNN-LSTM architecture, the initial layers consist of a series of convolutional and pooling layers, typical of a CNN. The convolutional layers are responsible for the automatic and adaptive learning of spatial hierarchies of features. By applying convolutional filters followed by activation functions, they learn to detect patterns such as edges, corners, and more complex shapes within the data. The pooling layers reduce dimensionality and computational complexity, focusing on the most salient features.

The output of the CNN component is then reshaped and fed into the LSTM layers. LSTMs are a specialized form of RNNs capable of learning long-range dependencies in the data. With their sophisticated gating mechanisms, including input, output, and forget gates, LSTMs can selectively remember or forget information, enabling them to capture temporal patterns over extended sequences. This makes them adept at modeling the temporal or sequential dimension of the data.

The combination of these two paradigms within a single model allows the CNN-LSTM to operate effectively on data that possesses both spatial and temporal dimensions. The CNN component's ability to extract spatial features forms a complementary relationship with the LSTM's capacity to model the dependencies across time (Jiao, Jiao, Mo, Liu, & Deng, 2019).

Mathematically, the convolution operation in the CNN part can be represented as:

$$f_{i,j} = activation\left(b + \sum_m \sum_n w_{m,n} \,.\, x_i + m, j + n\right)$$

Where $f_{i,j}$ is the feature map, b is the bias, $w_{m,n}$ are the weights of the convolutional filters, and $x_i + m, j + n$ are the elements of the input.

The LSTM part can be formulated through the following equations representing the gate activations and state updates:

$$f_t = \sigma\left(W_f \,.\, [h_{t-1}, x_t] + b_f\right)$$

$$i_t = \sigma(W_i \,.\, [h_{t-1}, x_t] + b_i)$$

$$\widetilde{C_t} = \tanh\,(W_c . [h_{t-1}, x_t] + b_c$$

$$C_t = \mathrm{f_t} \,.\, C_{t-1} + i_t \,.\, \widetilde{C_t}$$

$$o_t = \sigma(W_o \,.\, [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t . \tanh(C_t)$$

Here, σ denotes the sigmoid activation function,. The weight matric W and bias vector b are the learnable parameter, while f_t, i_t, o_t represent the forget, input, and output gates, respectively.

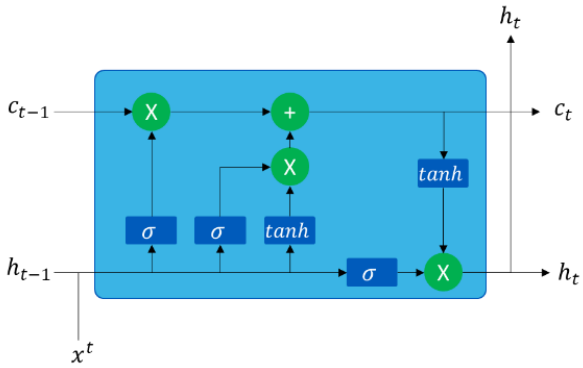

Figure 4: Diagram of an Individual LSTM Cell with Input Gate, Forget Gate and Output Gate

The CNN-LSTM architecture thus facilitates a rich, multi-faceted analysis of data, harnessing spatial and temporal dimensions. Its versatility and capability to learn complex hierarchical features make it a prominent choice in a variety of applications that demand an integrated understanding of spatial and temporal characteristics.

2.4 PROPOSED ARCHITECTURE

2.4.1 HIGH-LEVEL OVERVIEW

Figure 5 presents a high-level overview of the simulated data generation process and proposed architecture for integrity-based multi-sensor positioning. In this architecture design, an accelerometer and gyroscope are used to calculate INS positioning with Direction Cosine Matrix (DCM). Using a GRU-based error correction technique, INS errors caused by the integration of sensor data that contain bias and random errors are corrected by the GRU error predictions.

GNSS signals received from Ublox-F9P are then fed into a GRU multipath detection algorithm that utilizes GNSS observables to classify signals into either Line-of-Sight (LOS), multipath, and None-Line-of-Sight (NLOS) conditions. NLOS signals are then excluded from the Position, Velocity, and Timing (PVT) calculation to improve GNSS positioning performance.

CNN-LSTM architecture is utilized to provide monocular Visual Odometry (VO) information from a camera simulated in AirSim. The data output from this neural network is the change in position. This information is accumulated over time to provide VO position prediction.

Lastly, a Bayesian-LSTM architecture is used to fuse the position data from the previously mentioned parts of the system. This provides an improved position accuracy performance over using individual components of this architecture. The neural network output is the estimated position and the uncertainty in the prediction.

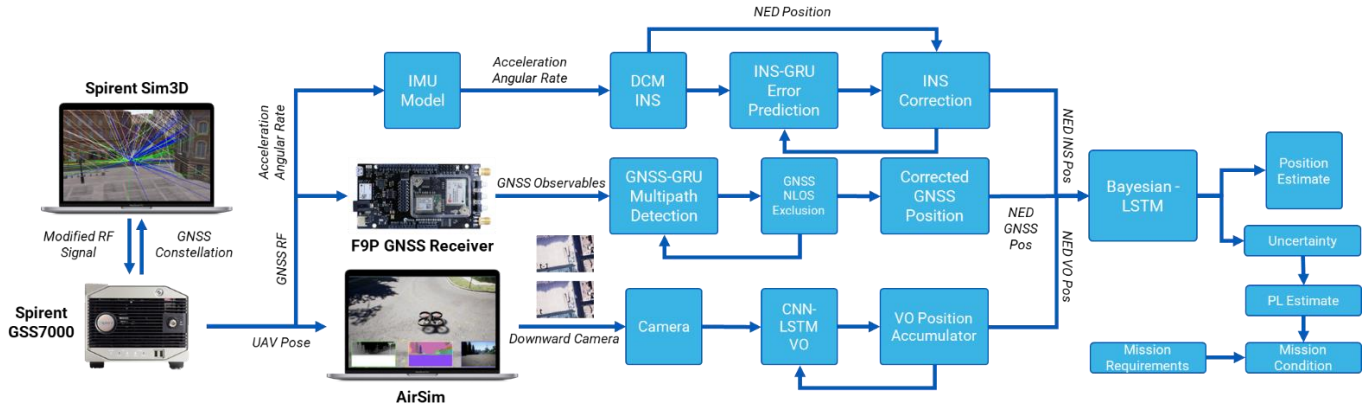

Figure 5: High-Level Overview of Simulated Data Generation Process and Proposed Architecture

2.4.2 INS ERROR CORRECTION USING GRU

A detailed diagram regarding the INS error correction system is shown in Figure 6. The training phase of the GRU involves measuring specific forces and angular velocities using an accelerometer and a gyroscope, which together constitute the system's Inertial Measurement Unit (IMU).

The calculation of the Inertial Navigation System (INS) position and velocity necessitates the utilization of the current system orientation to compute the Direction Cosine Matrix (DCM), which is essential for translating accelerometer and gyroscope readings from the body frame to the navigational frame. Subsequently, these translations are integrated to yield the position, velocity, and updated orientation information, which is employed as the reference INS for error estimation. The feature input for the GRU at each timestep comprises the navigation frame acceleration, angular rate, current orientation, and the time elapsed since the last Global Navigation Satellite System (GNSS) update. The target output for training is the position error, computed as the difference between the INS position and the GNSS ground truth data. The neural network consists of a two-layer GRU architecture with each layer containing 256 and 32 cells respectively with the sequence length input being 20. A fully connected layer is used to provide the output from the neural network. The batch size used is 64, the dropout rate is set at 0.2 and the activation function is a RELU. The ADAMAX optimizer is employed during the training to optimize the neural network weights and minimize the discrepancies between the predicted and actual position errors. Furthermore, the loss is calculated using Root Mean Squared Error (RMSE).

The second mode of the Neural Network is the prediction phase, which is active throughout the simulation. During this phase, the trained GRU predicts the positioning and velocity errors based on the given navigation frame acceleration, angular rate, current orientation and the time elapsed since the last GNSS update. The predicted errors are then deducted from the INS velocity and position readings to generate the system prediction. For evaluation purposes, the updated position prediction will be deducted from the ground truth information available in the simulation to assess system performance.

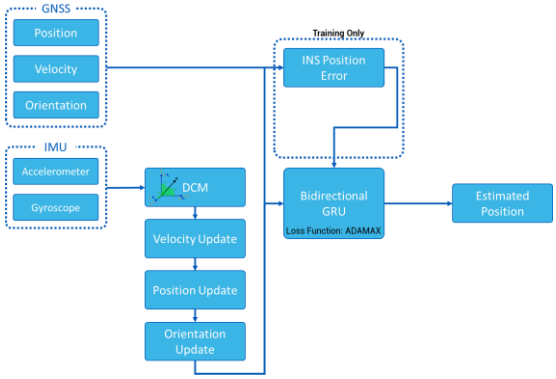

Figure 6: INS Error Correction Algorithm

2.4.3 MULTIPATH DETECTION AND EXCLUSION USING GRU

Figure 7 presents a classification algorithm based on the Gated Recurrent Unit (GRU) that leverages Global Navigation Satellite System (GNSS) to ascertain the presence of multipath conditions for signals. The algorithm utilizes pseudorange, ephemerides, Doppler shift, Carrier-to-Noise ratio (C/N0), and elevation data from each satellite. Employing a GRU facilitates the identification of nonlinear relationships and enables the utilization of past error dependencies for predicting multipath conditions. The aforementioned features serve as the input to the Neural Network, which then processes the data using various weights, biases, and non-linear functions to establish a connection between the input features and the output decision. The selection of these inputs was informed by a review of the current literature and existing systems. For instance, C/N0 was successfully employed in NLOS classification, while an elevation-based C/N0 was highlighted in (Kubo, Kobayashi, & Furukawa, 2020) as a critical factor in determining the presence of multipath. Moreover, Doppler shift and pseudorange were employed to enhance the classification performance of Support Vector Machines (SVM).

In the proposed system, the information received from each satellite is processed and classified every second into one of three distinct categories: LOS, multipath, and NLOS. LOS refers to signals that travel directly from the satellite to the receiver without any obstructions or reflections. Signals affected by multipath, on the other hand, are those that reach the receiver after reflecting off various surfaces, such as buildings or the ground, often resulting in a delay in signal reception and leading to inaccurate positioning. These signals are typically mixed with LOS signals and therefore can provide GNSS positioning information in situations where most are affected by interference. NLOS signals are those that do not have a direct path from the satellite to the receiver, usually because of obstructions like buildings, trees, or other structures.

If a signal is classified as NLOS, it is excluded from the Position, Velocity, and Time (PVT) calculations to ensure more accurate positioning. This classification is crucial for applications that require precise positioning, as it helps in filtering out inaccurate signals that could potentially degrade the system's performance. To compute the PVT, a minimum of four satellites is required, with the pseudorange and satellite positions information fed into the MATLAB 'receiver position' function to determine the receiver's position. When a multipath condition is detected and the corresponding signal is excluded, the corresponding satellite is flagged for a duration of one second. Consequently, the pseudorange measurement and position of that particular satellite are not utilized for receiver position calculation for that one-second duration. The neural network consists of a single-layer GRU architecture with 128 cells and a sequence length input of 60. A fully connected layer is used to provide the output from the neural network. The batch size used is 64, the dropout rate is set at 0.2 and the activation function is a RELU. The ADAMAX optimizer is employed during the training to optimize the neural network weights and minimize the discrepancies between the predicted and actual position errors. Furthermore, the loss is calculated using cross-entropy loss.

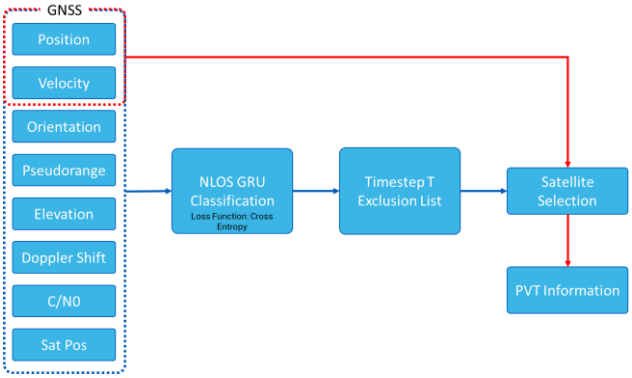

Figure 7: Multipath Detection and Exclusion

2.4.4 CNN-LSTM BASED VISUAL ODOMETRY

Figure 9 outlines an advanced visual odometry algorithm, leveraging a hybrid architecture that combines CNN and LSTM networks. Instead of processing single images independently, this approach concatenates two sequential images, thus including temporal information directly into the input data. The architecture consists of a modified ResNet-based CNN followed by two LSTM layers.

The CNN extracts spatial features using a modified ResNet architecture. This step transforms the raw image data into a more compact and informative representation. The spatial features are then fed into two LSTM layers, which are designed to model temporal dependencies and predict the vehicle's position difference between the two timesteps. The final prediction, representing the position difference, is made by a fully connected layer receiving the output of the LSTM layers. The predicted position differences from the model are accumulated over time to estimate the overall trajectory of the vehicle. This model is particularly beneficial for applications such as autonomous navigation and robotics, which require real-time and precise trajectory estimations. The hybrid CNN-LSTM architecture enables the model to capture both spatial and temporal dependencies in the data which can lead to improved positioning performance.

The neural network consists of a series of convolutional layers based on a modified ResNet-18 architecture, followed by a flattening layer, two LSTM layers, and a fully connected layer. Each LSTM layer contains 100 cells, a dropout rate is at 0.2 and the activation function is based on RELU. ADAMAX optimizer is employed during the training to optimize the neural network weights and minimize the discrepancies between the predicted and actual position differences.

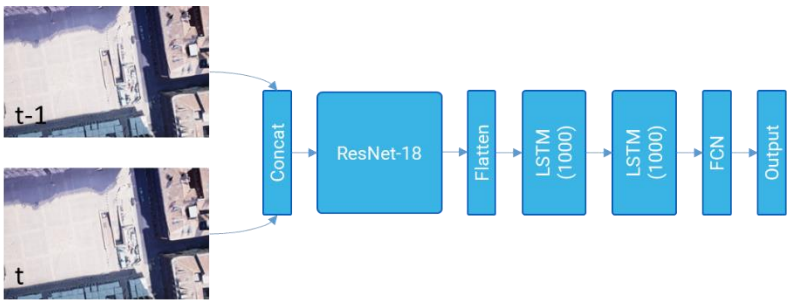

Figure 8: CNN-LSTM VO

2.4.5 BAYESIAN-LSTM SENSOR FUSION AND PROTECTIONAL LEVEL ESTIMATION

Once post processing for each sensor has been carried out, the data is then fused using a Bayesian-LSTM architecture. The input to the architecture is in form of a matrix which contains the input features: GNSS Position NED, INS Position NED and VO Position NED. Each input has a sequence length of 20 and a batch size of 64. The Bayesian-LSTM uses two layers of 128 and 64 cells each with a RELU activation function layer and a fully connected layer at the end for a 3-dimensional vector output in the form or NED frame positioning. Furthermore, to provide uncertainty, a Monte Carlo dropout layer is added after the LSTM layer to provide

randomized dropouts during the prediction phase of the algorithm. Once training is completed, the architecture provides 100 predictions of each timestamp. These are then used to calculate the mean and standard deviation of each prediction. To calculate the protection levels estimation, the below equation is used.

$$\boldsymbol{HPL = K_H d}$$

$$d = \sqrt{\frac{\sigma_E^2 + \sigma_N^2}{2} + \sqrt{\left(\frac{\sigma_E^2 - \sigma_N^2}{2}\right)^2 + (\sigma_E \sigma_N)^2}}$$

Where σ_E is the standard deviation in the East direction (m) and σ_N is the standard deviation in the North direction. K_H is the scalar factor that are both set to one in this scenario which represents an integrity risk of 0.1%.

2.5 EXPERIMENTAL SETUP

2.5.1 HARDWARE-IN-THE-LOOP CONFIGURATION

The predominant portion of drones is equipped with a Companion Computer (CC), generally employed for data processing from external sensors, like monocular cameras, or for the deployment of AI-based models to aid the Flight Control Unit (FCU) in managing the computational load associated with the UAV's navigation, control, and stability. A critical step in validating the sensor fusion framework, outlined in the prior section, is conducting a Hardware-In-the-Loop (HIL) simulation. The process commences by integrating a Pixhawk 2.4 board as an FCU, which is directly interfaced with a local computer serving as a CC. Utilizing Unreal Engine and AirSim enables the integration of the FCU into a photorealistic environment. The integration of photogrammetry data into the Unreal Engine facilitates the recreation of urban landscapes, thereby enhancing the level of detail in aspects such as light intensity, authentic weather effects, and material properties. The primary objective of this photorealistic environment is to assist in the validation and testing of optical sensors installed on the UAV, making it possible to define camera intrinsic and distortion parameters. Photogrammetry data from San Francisco was employed in the simulation, as illustrated in the figure below.

Figure 9: San Francisco 3D Environment in Unreal Engine with AirSim

Upon successfully establishing the connection between the FCU, AirSim, and the Unreal Engine, trajectory commands, derived from a Python file linked to the AirSim plugin, are relayed to the Spirent GSS7000 simulator. This enables the execution of HIL tests in an iterative manner, maintaining consistent trajectory assumptions. Concurrently, the trajectory estimated by the monocular camera is documented in an Excel file.

To accurately assess the proposed architecture's performance, it is essential to conduct training sessions using realistic scenarios, derived from the Spirent GSS7000 simulator and SimSENSOR. Both the Spirent GSS7000 and SimGEN are designed to simulate the Global Navigation Satellite System (GNSS) Radio Frequencies (RF) emitted by each constellation's satellites. Adjusting the simulation time and constellation settings enhances the realism of the scenario, enabling precise time-accurate comparisons. The

UAV trajectory chosen for this test is shown in Figure 10. The trajectory is part of a longer journey and represents an area in San Francisco with very dense urban canyons that can reduce GNSS performance. Furthermore, error tolerances in these urban canyons are considerably lower than clear-sky conditions and therefore require any positioning system used to be robust. Additionally, going around a block provides a challenge for both IMU and VO to keep track.

To enhance realism further, particularly in Urban Canyon scenarios, OKTAL-SE Sim3D software was employed to simulate the multipath effect on GNSS signals using ray-tracing method. The software accounts for the specific geometry of the urban environments simulated in this study. Additionally, the distinct material properties that influence signal reflection characteristics were also considered, adding another layer of realism. This simulation generated LOS, multipath, and NLOS conditions for the GNSS signals processed by the receiver. No additional multipath mitigation techniques, aside from those developed in this paper or are already provided by the receiver manufacturer, were used to assist under these conditions.

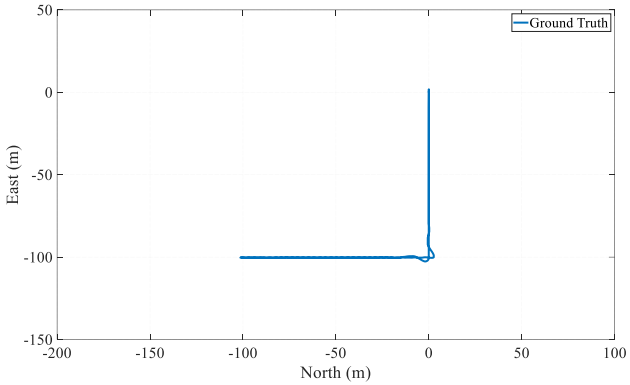

Figure 10: Ground Truth UAV Trajectory for Simulation

Moreover, the repeatability of the simulation setup permits direct algorithm technique comparisons, thereby minimizing the independent variables required for testing. These signals are then fed into a GNSS receiver for position data capture. To generate the required IMU data for training, SimSENSOR is utilized to produce the accelerometer and gyroscope measurements exhibited by the vehicle. SimSENSOR also permits the simulation of realistic sensor errors by modifying the SimSENSOR's deterministic and stochastic error parameters, such as the random walk rate and bias offset. In this study, the IMU stochastic and deterministic errors were adjusted to replicate the characteristics of an Advanced Navigation Orientus IMU sensor, the specifications of which are provided in Table I. This data is then relayed via the User Datagram Protocol (UDP) communication protocol to the AirSim simulator.

Table I: IMU Random Error and Bias for Gyroscope and Accelerometer

IMU Sensor A			
Accelerometer		***Gyroscope***	
Scaling factor (ppm)	600	***Scaling factor (ppm)***	500
Bias (ug)	20	***Bias (deg/h)***	3
ARW (ug/sqrt(Hz))	100	***GRW (deg/s/sqrt(h))***	0.004
Update rate (Hz)	100		

2.5.2 DATA COLLECTION

The sensor and simulation system data were collected and stored in CSV file format, with each timestep's time and date documented. This data was then processed in MATLAB to produce the ground truth data, which was used as the target for training. The input data comprised the IMU sensors' raw output (accelerometer and gyroscope in XYZ) and the GNSS latitude, longitude, and height information, recorded at a rate of 1 Hz. The data was synchronized to the same timestep as the accelerometer and gyroscope data, recorded at 100 Hz, using interpolation.

3. RESULTS AND DISCUSSION

3.1 GRU-BASED ERROR CORRECTION PERFORMANCE

Figure 11 shows a comparison between the Ground Truth data and the GRU Error Correction Algorithm that is applied to the calculated INS position. The plot shows that with the correction applied by the GRU, the INS position follows the ground truth trajectory. This is especially true for sections between 0 m and -80 m in the east direction where the INS error did not exceed greater than 10 m of error. However, there is a greater variation between -10 m and 100 m in the North direction. This may indicate areas that are not represented as well in the training data as earlier segments and therefore lead to an increase in prediction errors.

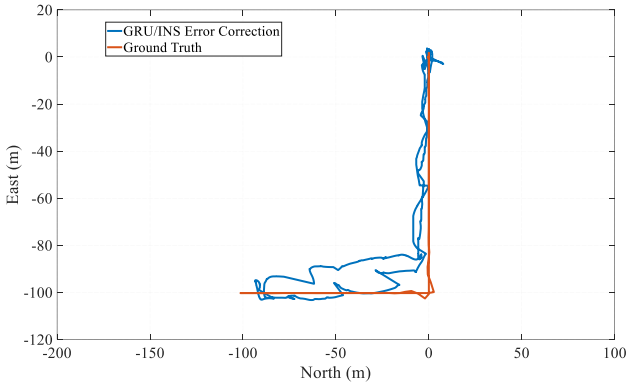

Figure 11: Comparison between Ground Truth and GRU Error Correction Algorithm in the Horizontal Direction

Table II compares the mean, standard deviation, and 95th percentile error between INS and GRU error correction algorithms in the north and east direction. The mean error exhibited in the North and East directions for INS and GRU is 19.9 m, 20.2 m, 2.45 m, and 5.38 m respectively. This indicates that the GRU outperforms the INS positioning estimate by up to 87.7%. This may be due to the biases and random walk effects associated with INS systems due to the constant integration of IMU information over a period leading to a drift in the position prediction from the ground truth if not corrected promptly. In contrast, the GRU architecture can model the highly non-linear relationship between the feature inputs and the output to obtain the estimated position error which is then subtracted from the INS position estimate. A similar story emerges when comparing the standard deviation between INS and GRU with a reduction of up to 86.1% being observed. Lastly, the 95th percentile shows a reduction of up to 86.7% when comparing INS with GRU. In terms of 95th percentile error in the horizontal direction for the whole trajectory, INS exhibits a 77.5 m error compared to GRU with a 22.1m error. This is a reduction of 71.5% in the horizontal direction which indicates the effectiveness of GRU for the prediction of INS position errors.

Table II: Comparison of Mean, Standard Deviation, and 95th Percentile Error between INS and GRU Error Correction in the North and East Direction

	σ error(m)		**μ(m)**		**95th Error (m)**	
	N	*E*	*N*	*E*	*N*	*E*
INS Sensor	19.9	20.2	16.7	18.2	53.3	56.6
GRU Error Correction	2.45	5.38	2.32	7.77	7.09	20.92

3.2 GRU-BASED MULTIPATH DETECTION & EXCLUSION PERFORMANCE

Figure 12 is a confusion matrix result from the testing of the GRU classification architecture with 3 categories. Class 0 represents situations where the signal is LOS. Class 1 represents signals that are Multipath (LOS signal + Reflected Signals), and Class 2 represents signals that are NLOS. The aim of this confusion Matrix is to have a majority of the classified signals along the diagonal of the grid. This would indicate the Predicted Class matches the True Class. In the testing dataset, this is shown to be the case with a robust performance seen for classifying Multipath and NLOS signals correctly. The accuracy of the GRU classification algorithm is 87.1%. The precision regarding LOS was 64.2%, recall (sensitivity) was 62%, and specificity 92.5%. Precision for the Multipath class was 89.7%, recall 91.5%, and specificity 63.4%. Precision for the NLOS class was 90.1%, recall 83.9%, and specificity 98.9%.

This shows that the classification algorithm accurately predicts NLOS classes with the ability to classify instances when signals are NLOS and when signals are not NLOS. Therefore, it is of benefit to use this algorithm to filter out signals that are classified as NLOS to improve positioning performance from the GNSS receiver.

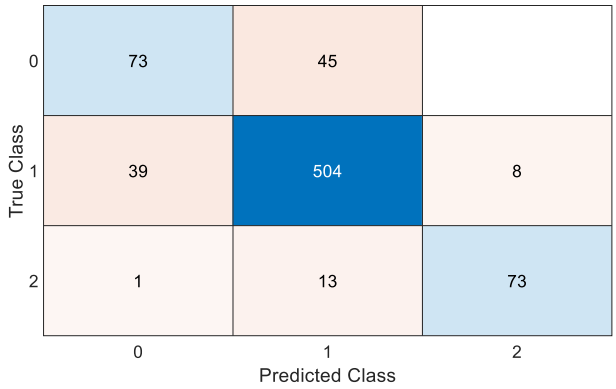

Figure 12: Confusion Matrix Result for GRU-GNSS LOS, Multipath, and NLOS Classification

Figure 13 shows a comparison of the receiver-calculated position estimate with the ground truth trajectory. Some of the position estimates from the GNSS receiver closely match the Ground Truth which indicates areas of low multipath and NLOS interference with the receiver. However, certain sections see large errors of position compared to the ground truth due to NLOS signals interfering with the receiver-based estimated position. This indicates that there is a need for a technique to identify and exclude NLOS signals to use it reliably for GNSS fusion.

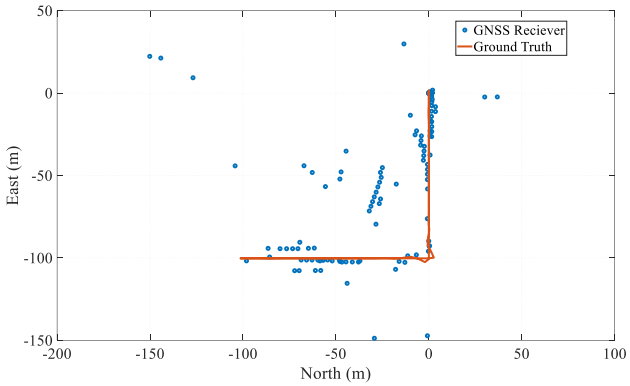

Figure 13: Comparison of Receiver Calculated Position Plot and Ground Truth without Post-Processing.

Figure 14 compares the position estimate with NLOS signals removed using the GRU classification algorithm with the ground truth trajectory. This graph showcases the improvements that can be achieved with a deep learning classification algorithm with temporal dependencies accounted for in the GRU architecture. A majority of the position estimates calculated after removing NLOS signals are close to the ground truth trajectory with only minor points being off the ground truth. However, comparing areas between -10 m and 30 m in the North direction with Figure 13 shows that some NLOS signals have successfully been identified but some may have remained. Furthermore, Multipath signals are still present in the readings which can interfere with LOS signals and reduce positioning accuracy. However, these effects are minor compared to the significance of NLOS which contributes to the overall positioning accuracy. Furthermore, we start to see gaps in areas such as -30 m and -40 m in the North direction. This is due to the number of satellites available after removing NLOS signals dropping below 4 and there cannot calculate a 3D position. This, therefore, requires other sensors such as IMUs to aid in GNSS outages.

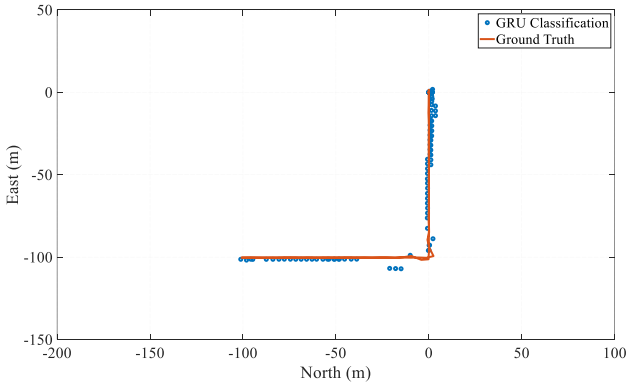

Figure 14: Comparison of Position Estimate with NLOS Signals Removed using GRU and Grund Truth

Table III compares the mean, standard deviation, and 95th percentile error between INS and GRU error correction algorithms in the North and East directions. The mean error exhibited in the North and East directions for INS and GRU is 10.1 m, 4.3 m, 1.64 m, and 1.46m respectively. This shows an improvement in the mean error position performance from GNSS by up to 83.8%. This indicates that the algorithm is effectively able to classify NLOS signals which then leads to an improvement in the positioning accuracy. Regarding standard deviation, the same story is true with the GRU outperforming the receiver. For the 95th percentile in the horizontal direction, a 147 m error is observed by the receiver compared to 3.8 m for the GRU. This is a reduction of 97.4% in the error observed.

Table III: Comparison of Mean, Standard Deviation, and 95th percentile Error between Receiver and GRU Multipath Detection Algorithm in the North and East Direction

	σ error(m)		**μ(m)**		**95th Error (m)**	
	N	*E*	*N*	*E*	*N*	*E*
Receiver	10.1	4.3	49.9	48.3	109	100
GRU Multipath Detection	1.64	1.46	0.41	0.63	2.46	2.9

3.3 CNN-LSTM-BASED VISUAL ODOMETRY PERFORMANCE

Figure 15 shows a comparison between the CNN-LSTM VO output and the ground truth trajectory of the vehicle. In general, the CNN-LSTM VO output closely follows the ground truth trajectory. This illustrates that the CNN-LSTM can predict the position change between two consecutive frames effectively. However, as with INS, there is some drifting occurring. This is due to each new prediction being accumulated to the existing prediction to get a new position. Furthermore, between -50 m and 0 m in the East direction when returning to base, the drift becomes significant as to diverge completely from the trajectory. This may indicate further training that is required to reduce consecutive prediction errors and increase positioning accuracy and longevity for relying on the position estimate.

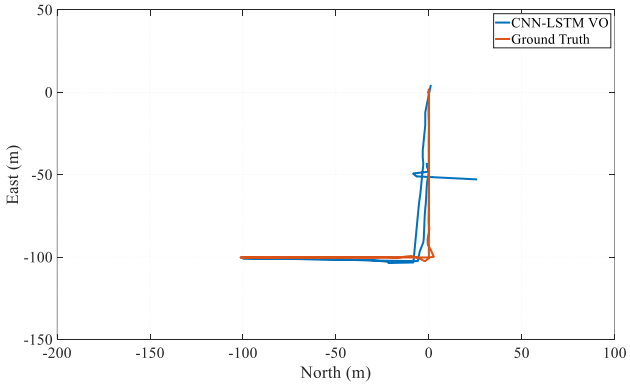

Figure 15: Comparison of CNN-LSTM VO Performance with Ground Truth

Table IV provides two important performance aspects. First, it outlines the CNN-LSTM VO performance regarding the mean, standard deviation, and 95th percentile position error. Here, we can see that the mean error in the north and east direction is 1.12 m and 0.88 m respectively. Furthermore, the horizontal 95th percentile error is 8.13 m. Regarding the second important comparison, the table assesses the CNN-LSTM VO performance with the existing GRU architectures examined above. In regards to the mean error, CNN-LSTM VO outperforms both the GRU-GNSS receiver and GRU-INS position. However, for both standard deviation and 95th percentile error, the GRU-GNSS outperforms both algorithms. This is to be expected as both INS and VO suffer from errors that accumulate over time and therefore diverge from the ground truth whilst GNSS is not affected by such a phenomenon.

Table IV: CNN-LSTM VO position Mean, Standard Deviation, and 95th Percentile Error in the North and East Direction with a Comparison to Previously Mentioned Architectures

	σ error(m)		**μ(m)**		**95th Error (m)**	
	N	*E*	*N*	*E*	*N*	*E*
CNN-LSTM VO	1.12	0.88	1.08	3.32	3.28	7.44
GRU Multipath Detection	1.64	1.46	0.41	0.63	2.46	2.9
GRU Error Correction	2.45	5.38	2.32	7.77	7.09	20.92

3.4 BAYESIAN-LSTM FUSION PERFORMANCE

Figure 16 compares the output of all individual architectures with ground truth vehicle trajectory before fusion using Bayesian-LSTM. This is a complementary visual comparison to Table IV to demonstrate the need for the fusion of sensors to provide cm-level accuracy in Urban Canyons environments where GNSS navigation is affected.

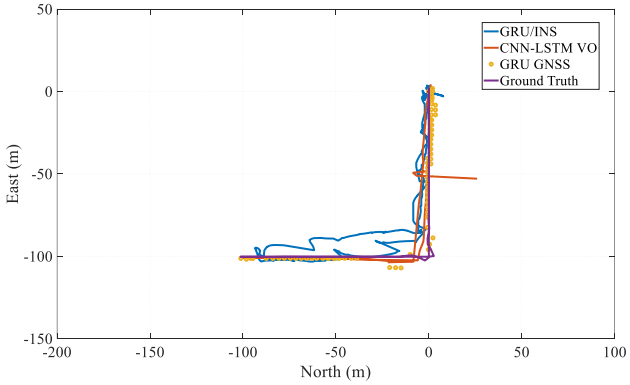

Figure 16: Comparison of GRU-INS, CNN-LSTM VO, and GRU-GNSS with Ground Truth before Sensor Fusion

In Figure 17, the output of the Bayesian LSTM is presented and compared against the ground truth. In contrast to Figure 16, this shows a position performance improvement over any single sensor output provided thus far. It indicates that the sensor fusion can utilize the position information from each sensor after error correction (in the case of INS and GNSS) to provide increased accuracy. However, tiny fluctuations still exist. This is seen between -20 m and -40 m in the north direction and corresponds to the reduced number of available satellites exhibited in Figure 14. The behavior is to be expected in urban canyons with a large degree of satellites blocked. To improve conditions for GNSS further, more constellations can be utilized to improve signal availability for position calculation. However, due to the fusion of these sensors, the Bayesian-LSTM is not fully reliant on accurate GNSS and therefore provides improvement over single sensors for navigation.

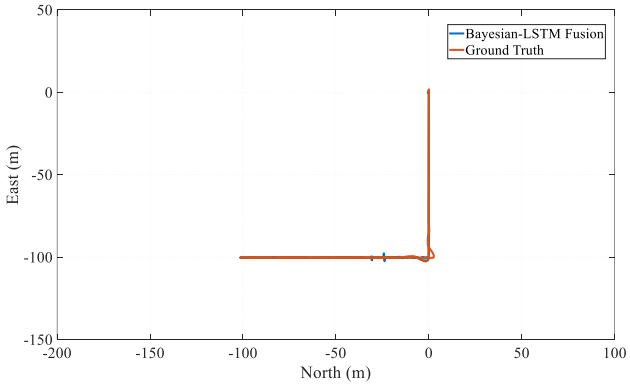

Figure 17: Comparison of Bayesian-LSTM Output with Ground Truth Vehicle Trajectory

Figure 18 highlights the variation in position error from the Bayesian-LSTM prediction over the trajectory of the vehicle. This shows two peaks that correspond to the reduced number of GNSS satellites available for position estimation. This, therefore, provides the estimated impact of the GNSS position accuracy in the fusion architecture (up to 2.5 m in the horizontal direction).

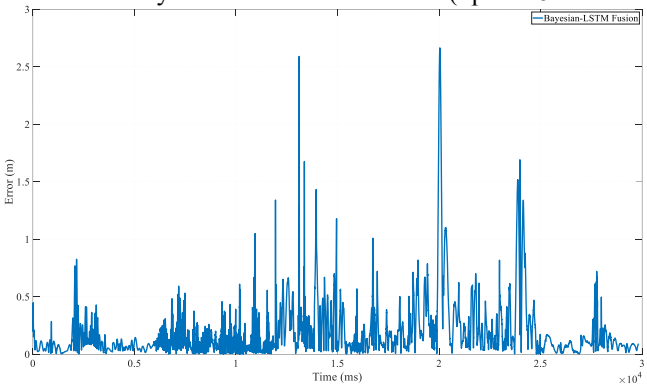

Figure 18: Horizontal Position Prediction Error of Bayesian-LSTM

Table V is a comparison of the mean, standard deviation, and 95th percentile error between Bayesian-LSTM and a GNSS/IMU/VO EKF-GRU algorithm that was developed previously. This architecture uses two local EKF filters to integrate GNSS-IMU and IMU-VO. Then, using a GRU for each navigation frame direction, the data is fused to provide a final position output. The mean error exhibited in the North and East direction for Bayesian-LSTM and the GNSS/IMU/VO EKF-GRU architecture is 0.001 m, 0.0006 m, 0.12 m, and 0.09 m respectively. Improved performance is to be expected as the GNSS signals are filtered to remove NLOS and the use of a CNN-LSTM architecture for VO. The same story is true for the standard deviation and 95th percentile with a horizontal error improvement of 30.1% for the Bayesian-LSTM.

Table V: Comparison of Mean, Standard Deviation, and 95th Percentile Error Between Bayesian-LSTM and State-Of-Art Literature Architecture in the North and East Direction

	σ error(m)		**μ(m)**		**95th Error (m)**	
	N	*E*	*N*	*E*	*N*	*E*
GNSS/IMU/VO EKF-GRU (Negru, Geragersian, Petrunin, Grech, & Busenel, 2023)	0.12	0.09	0.28	0.28	0.68	0.65
Proposed System	0.001	0.0006	0.233	0.231	0.467	0.463

3.5 INTEGRITY ANALYSIS

Figure 19 shows an example output from the Bayesian-LSTM compared to the ground truth trajectory and calculated PL based on the uncertainty output from the architecture. PL is defined as the statistical bound errors computed to guarantee the probability of the position error being smaller than or equal to the targeted integrity. The diagram shows the Bayesian-LSTM position output closely

following the ground truth. Furthermore, any slight differences in these are still within the bounds of the calculated protection level which provides integrity in the sensor fusion architecture.

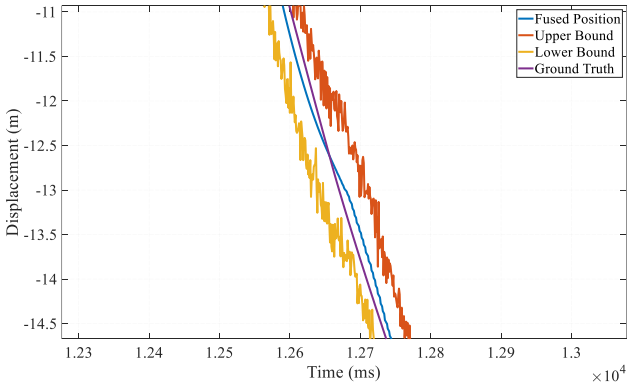

Figure 19: Example output of Bayesian-LSTM Uncertainty with Protection Level Calculation

Figure 20 provides an example case where the Bayesian-LSTM mean position output does not closely track the ground truth, however the protection level is still able to estimate the uncertainty in its position estimate and therefore correctly predicts the bounds that the position will lie between. Furthermore, it can be seen that in areas where the fused position is not tracking the ground truth closely due to increased noise from sensors, the estimated protection level size is greater (timestamp 1.17 to 1.2). However, areas with better tracking (timestamp 1.2 to 1.23) show a reduced protection level size which is the desired behavior.

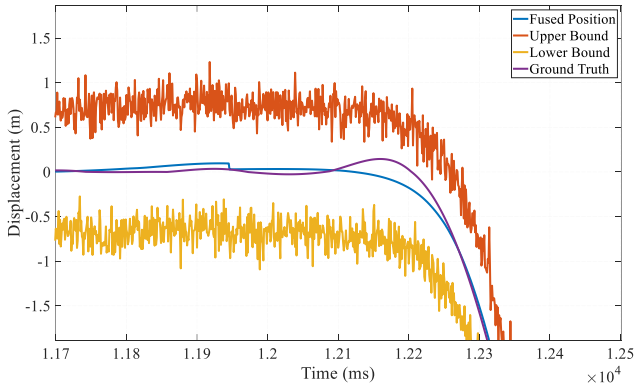

Figure 20: Example of Protection Level Capturing Position Uncertainty

In Figure 21, a comparison is made between the Bayesian-LSTM output, the calculated protection level based on the uncertainty output from the Bayesian-LSTM, the GRU-INS position estimate, and ground truth. In this example, the Bayesian-LSTM does not track the ground truth well. This is also true for the protection level estimates that do not account for the increased uncertainty in this scenario. However, this may be due to several factors including not enough training examples that would aid in better tracking these specific segments or due to increased sensor noise. Compared to just GRU-INS, it can be seen that this does not closely match the ground truth and performs worse than the GRU-INS. This may provide additional insight into the reason why the Bayesian-LSTM did not perform as expected in this scenario. However, as discussed in the previous section, the Bayesian-LSTM architecture performs better than both the individual sensors and the compared algorithm.

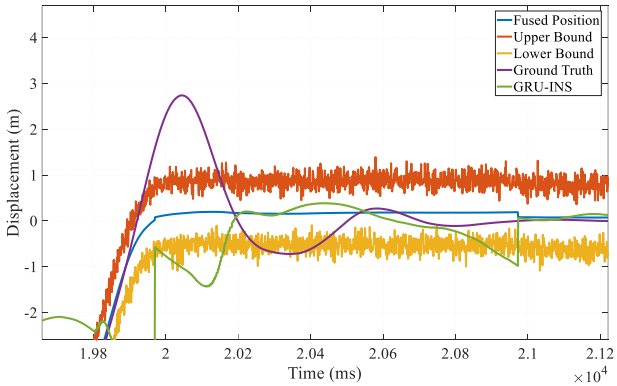

Figure 21: Comparison of Bayesian-LSTM, Protection Level, GRU-INS, and Ground Truth

4. CONCLUSION & FUTURE WORK

In conclusion, the paper demonstrated a proposed robust multi-sensor fusion algorithm using sensor data from GNSS, IMU, and Camera. The data collected from IMU was used to provide INS position data in the local navigation frame. The INS error was then estimated based on the feature inputs and target output using a GRU algorithm. Results showed a reduction of up to 86.7% when comparing INS with GRU. Furthermore, regarding 95th percentile errors in the horizontal direction, INS exhibits a 77.5 m error compared to GRU with a 22.1 m error. This is a reduction of 71.5%. GNSS signals were recorded and classified as either LOS, Multipath, or NLOS. If the signal was classified as NLOS, it would not be used for GNSS positioning. The results showed a reduction in 95th percentile errors in the horizontal direction of 143.2 m. This is a reduction of 97.4% in the error observed. A CNN-LSTM architecture was developed based on a ResNet-18 architecture and LSTM layers. The output from two consecutive images from the camera was the translation of the motion of the vehicle. The outputs were accumulated to obtain the position estimation. Here, the results showed a mean error in the north and east directions of 1.12 m and 0.88 m respectively. The processed data from each sensor was then fed into a Bayesian-LSTM algorithm for data fusion and estimation of the positioning uncertainty. The architecture was then compared to a GNSS/IMU/VO EKF-GRU architecture. The comparison showed a 95th percentile error improvement in the horizontal direction of 30.1% for the Bayesian-LSTM. Lastly, the uncertainty data provided by the architecture was then processed to provide the protection level estimate which was then compared against the ground truth with promising results. The introduction of this architecture provides the foundation for building a robust fusion architecture that can be deployed in UAVs that are traveling in urban canyons scenarios which require an accurate and reliable positioning system to safely navigate complex environments. Future work includes running the architecture on multiple scenarios with an increase in data for machine learning training. Furthermore, expansion into additional complimentary sensors is being explored to improve positioning accuracy further.

REFERENCES

Altinoz, B., & Unsal, D. (2014). Look up table implementation for IMU error compensation algorithm. *2014 IEEE/ION Position, Location and Navigation Symposium - PLANS 2014.* Monterey, CA, USA.

Bhattacharyya, S., & Gebre-Egziabher, D. (2015). Kalman filter–based RAIM for GNSS receivers. *IEEE Transactions on Aerospace and Electronic Systems, 51*(3), 2444 - 2459.

Cho, K., Merrienboer, B., Bahdanau, D., & Bengio, Y. (2014). On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. *arxiv, 1409*(1259).

Elsanhoury, M., Koljonen, J., Valisuo, P., Elmusrati, M., & Kuusniemi, H. (2021). Survey on Recent Advances in Integrated GNSSs Towards Seamless Navigation Using Multi-Sensor Fusion Technology. *Proceedings of the 34th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2021).*

Geragersian, P., Petrunin, I., Guo, W., & Grech, R. (2021). An INS/GNSS fusion architecture in GNSS denied environment using gated recurrent unit. *AIAA SCITECH 2022 Forum.* San Diego, USA.

Geragersian, P., Petrunin, I., Guo, W., & Grech, R. (2022). Multipath Detection from GNSS Observables Using Gated Recurrent Unit. *2022 IEEE/AIAA 41st Digital Avionics Systems Conference (DASC).* Portsmouth, VA, USA.

Geragersian, P., Petrunin, I., Guo, W., & Grech, R. (2023). Uncertainty-based Sensor Fusion Architecture using Bayesian-LSTM Neural Network. *AIAA SCITECH 2023 Forum.* National Harbor, MD, USA: Aerospace Research Central.

Hu, G., Wang, W., Zhong, Y., Gao, B., & Gu, C. (2018). A new direct filtering approach to INS/GNSS integration. *Aerospace Science and Technology, 77*, 755-764.

Isik, O. K., Hong, J., Petrunin, I., & Tsourdos, A. (2020). Integrity Analysis for GPS-Based Navigation of UAVs in Urban Environment. *Robotics, 9*(3).

Jiao, J., Jiao, J., Mo, Y., Liu, W., & Deng, Z. (2019). MagicVO: An End-to-End Hybrid CNN and Bi-LSTM Method for Monocular Visual Odometry. *IEEE Access* , 94118 - 94127.

Kuang, Q., Wu, J., Pan, J., & Zhou, B. (2020). Real-Time UAV Path Planning for Autonomous Urban Scene Reconstruction. *2020 IEEE International Conference on Robotics and Automation (ICRA).* Paris, France.

Kubo, N., Kobayashi, K., & Furukawa, R. (2020). GNSS Multipath Detection Using Continuous Time-Series C/N0. *Sensors*, 4059.

Liu, H., Zheng, G., Wang, H., & Feng, C. (2010). Research on integrity monitoring for integrated GNSS/SINS system. *The 2010 IEEE International Conference on Information and Automation.* Harbin, China.

Negru, S. A., Geragersian, P., Petrunin, I., Grech, R., & Busenel, G. (2023). Realism-oriented design, verification, and validation of novel robust navigation solutions. *Engineering Proceedings*.

Su, K., Jin, S., & Hoque, M. M. (2019). Evaluation of Ionospheric Delay Effects on Multi-GNSS Positioning Performance. *Remote Sensing, 11*(2), 171.

Tao, X., Zhu, B., Xuan, S., Zhao, J., Jiang, H., Du, J., & Deng, W. (2022). A Multi-Sensor Fusion Positioning Strategy for Intelligent Vehicles Using Global Pose Graph Optimization. *IEEE Transactions on Vehicular Technology, 71*(3), 2614 - 2627.

Unsal, D., & Demirbas, K. (2012). Estimation of deterministic and stochastic IMU error parameter. *Proceedings of the 2012 IEEE/ION Position, Location and Navigation Symposium.* Myrtle Beach, SC, USA.

Xu, S., Petrunin, I., & Tsourdos, A. (2022). Experimental Evaluation of GNSS and IMU Fusion Using Gated Recurrent Unit. *2022 Integrated Communication, Navigation and Surveillance Conference (ICNS).* Dulles, VA, USA: IEEE.

Zabalegui, P., Miguel, G., Perez, A., Mendizabal, J., Goya, J., & Adin, I. (2020). A Review of the Evolution of the Integrity Methods Applied in GNSS. *IEEE Access, 8*, 45813 - 45824.

Zhang, G., & Hsu, L.-T. (2018). Intelligent GNSS/INS integrated navigation system for a commercial UAV flight control system. *Aerospace Science and Technology*.

Zheng, B., Fu, P., Li, B., & Yuan, X. (2018). A Robust Adaptive Unscented Kalman Filter for Nonlinear Estimation with Uncertain Noise Covariance. *Sensors*, 808.

Zhu, B., & meng, F. (2018). Integrity monitoring with MAIME upon integrated GNSS/SINS for SGEs. *2017 3rd IEEE International Conference on Computer and Communications (ICCC).* Chengdu, China.

Zhu, N., Marais, J., Betaille, D., & Berbineau, M. (2018). GNSS Position Integrity in Urban Environments: A Review of Literature. *IEEE Transactions on Intelligent Transportation Systems, 19*(9), 2762 - 2778.

2023-09-15

A hybrid deep learning approach for robust multi-sensor GNSS/INS/VO fusion in urban canyons

Geragersian, Patrick

The Institute of Navigation

Geragersian P, Petrunin I, Guo W, Grech R. (2023) A hybrid deep learning approach for robust multi-sensor GNSS/INS/VO fusion in urban canyons. In: Proceedings of the 36th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2023). 11-15 September 2023, Denver, USA, pp. 2624-2643

http://dx.doi.org/10.33012/2023.19271