



DRIFT-Net: Drive-State-Aware Weighted Interpolation for Error Forecasting and Trajectory Correction Network

Downloaded from: <https://research.chalmers.se>, 2026-10-23 22:42 UTC

Citation for the original published paper (version of record):

Jeong, J., Quan, Y., Choi, W. (2026). DRIFT-Net: Drive-State-Aware Weighted Interpolation for Error Forecasting and Trajectory Correction Network. IEEE Access, 14: 148490-148500.
<http://dx.doi.org/10.1109/ACCESS.2026.3737241>

N.B. When citing this work, cite the original published paper.

© 2026 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, or reuse of any copyrighted component of this work in other works.

Received 25 August 2026, accepted 21 September 2026, date of publication 24 September 2026,
date of current version 30 September 2026.

Digital Object Identifier 10.1109/ACCESS.2026.3737241

RESEARCH ARTICLE

DRIFT-Net: Drive-State-Aware Weighted Interpolation for Error Forecasting and Trajectory Correction Network

JAE WON JEONG¹, (Graduate Student Member, IEEE), YING SHUAI QUAN², (Member, IEEE),
AND WOO YOUNG CHOI³, (Member, IEEE)

¹Department of Intelligent Robot Engineering, Pukyong National University, Busan 48513, Republic of Korea

²Mechatronics Group, Department of Electrical Engineering, Chalmers University of Technology, 412 58 Gothenburg, Sweden

³Department of Control and Instrumentation Engineering, Pukyong National University, Busan 48513, Republic of Korea

Corresponding author: Woo Young Choi (wychoi@pknu.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by Korean Government [Ministry of Science and Information and Communications Technology (MSIT)] under Grant RS-2023-00213640; in part by Korea Agency for Infrastructure Technology Advancement (KAIA) grant funded by the Ministry of Land, Infrastructure and Transport under Grant RS-2026-25524573; in part by the Basic Science Research Program through NRF funded by the Ministry of Education under Grant RS-2025-25432737; and in part by the Global Joint Research Program funded by Pukyong National University under Grant 202505790001.

ABSTRACT Accurate vehicle localization is essential for autonomous driving. However, vehicle position estimation becomes challenging when localization information from sensors such as the Global Navigation Satellite System (GNSS) and Light Detection and Ranging (LiDAR) is unavailable, degraded, or unreliable. In such situations, dead reckoning can provide continuous localization, but accumulated yaw estimation errors often result in significant position error. To address this issue, this paper presents DRIFT-Net, a DRive-state-aware Weighted Interpolation for error Forecasting and Trajectory correction Network, to improve dead-reckoning-based vehicle localization. DRIFT-Net employs a Convolutional Neural Network–Long Short-Term Memory (CNN–LSTM) architecture to extract driving features and incorporates an expert layer that performs state-based weighted interpolation of expert predictions to estimate yaw error. A Kalman filter is subsequently applied to suppress noise and refine the final vehicle position estimate through smoothing. Experimental results in low-speed and high-speed driving scenarios demonstrate that the proposed approach improves localization accuracy. The proposed method achieved yaw Root Mean Square Error (RMSE) values of 0.339 deg and 0.148 deg in the respective scenarios, confirming improved performance compared with existing methods.

INDEX TERMS Autonomous driving, dead reckoning, deep learning, error estimation, localization.

I. INTRODUCTION

Accurate vehicle localization is a critical requirement for ensuring the safety and reliability of autonomous driving systems. In general, vehicle position information is obtained using sensors such as the Global Navigation Satellite System (GNSS), cameras, Light Detection and Ranging (LiDAR), and Inertial Measurement Units (IMU) [1], [2]. However,

vehicle localization becomes significantly more challenging in localization-degraded environments, such as tunnels, dense urban canyons, and underground parking structures, where external localization information may be blocked, degraded, or unavailable. In such environments, vehicle localization often relies on dead reckoning using onboard sensors, including IMU and wheel odometry [3]. However, small measurement errors and biases in the yaw rate accumulate through temporal integration, resulting in heading drift. The accumulated heading error continuously changes the

The associate editor coordinating the review of this manuscript and approving it for publication was Domenico Rosaci¹.

propagation direction of dead reckoning and consequently increases the longitudinal and lateral position errors during prolonged operation without external correction [4]. Therefore, reliable prediction and compensation of the accumulated yaw error are essential for maintaining vehicle localization accuracy during prolonged GNSS outages.

To mitigate these errors, various compensation techniques have been extensively investigated. A representative approach is Kalman filter-based sensor fusion, where measurements from sensors such as IMU, wheel odometry, vision sensors, and radar are integrated to reduce dead-reckoning errors [5], [6]. Another widely adopted approach is map matching, which aligns the estimated vehicle trajectory with preconstructed high-definition maps, enabling relatively stable localization even during GNSS outages [7]. However, these methods typically require additional sensing modalities or the construction and maintenance of high-definition maps, leading to increased cost and system complexity.

Recently, deep learning-based approaches have attracted considerable attention as an alternative solution. By leveraging large-scale driving datasets, deep learning models can learn complex sensor characteristics and compensate for time-varying error patterns, and they have been widely applied to autonomous driving and vehicle control systems [8], [9]. To address the degradation of Extended Kalman Filter (EKF) performance during prolonged GNSS outages, a Recurrent Neural Network (RNN)-based framework was proposed in which the RNN learns the discrepancy between Inertial Navigation System (INS) and GNSS measurements and subsequently predicts localization errors when GNSS signals become unavailable [10]. Furthermore, a Convolutional Neural Network–Long Short-Term Memory (CNN–LSTM)-based Visual Inertial Odometry (VIO) framework was proposed to estimate vehicle pose using camera and IMU measurements [11]. In this framework, a CNN extracts visual features from camera images, while an LSTM integrates visual and inertial information to estimate vehicle position and orientation. More recently, transformer-based models have been introduced to predict wheel-odometry uncertainty and compensate for dead-reckoning errors, demonstrating improved long-term prediction performance through real-vehicle experiments [12]. For wheeled vehicles, an IMU-only dead-reckoning framework has also been proposed in which a deep neural network dynamically adapts the measurement-noise parameters of a Kalman filter according to vehicle motion, improving localization robustness under varying driving conditions [13].

Recent hybrid approaches have further combined deep temporal models with model-based navigation estimators. A CNN–GRU model has been employed to predict GNSS pseudo-measurements during signal outages, while an improved robust adaptive Kalman filter was used to account for the uncertainty of the predicted measurements [14]. Another study integrated CNN–LSTM–Attention-based position-increment prediction with factor graph optimization using INS and odometer

measurements to maintain vehicle positioning during GNSS outages [15]. Adaptive trajectory planning has also been investigated in autonomous aerial systems, where spatiotemporal task conditions and current vehicle states are incorporated to dynamically adjust multi-AAV trajectories in response to changing operating conditions [16]. Such studies highlight the benefit of state-aware adaptation in trajectory management under dynamic operating conditions.

These studies demonstrate that deep learning models can represent complex and time-varying navigation-error characteristics. However, many existing approaches either rely on high-dimensional inertial or visual measurements or apply a single global prediction mapping over the entire operating range.

To address this challenge, this paper proposes DRIFT-Net, a DRive-state-aware weighted Interpolation for error Forecasting and Trajectory correction network, for real-time yaw-error prediction and compensation in dead-reckoning localization. DRIFT-Net combines deep learning-based time-series representation learning with a Mixture-of-Experts (MoE) architecture to capture varying error characteristics under different driving conditions.

The proposed framework employs CNN and LSTM modules to learn temporal relationships between vehicle velocity and yaw rate, while multiple expert heads generate candidate yaw-error estimates for different driving states. Rather than relying on a single regression head with one global output mapping, DRIFT-Net employs multiple linear expert heads associated with reference operating states in the normalized velocity–yaw-rate space. Each expert learns an independent yaw-error mapping from the shared CNN–LSTM representation, allowing the model to account for variations in error characteristics across driving states. Since vehicle velocity and yaw rate vary continuously, hard selection of a single expert may lead to abrupt changes in the predicted yaw error near the boundaries between operating regions. Therefore, DRIFT-Net combines the expert predictions using deterministic distance-based interpolation according to the current vehicle state. This formulation provides a continuous transition among expert outputs while maintaining an explicit relationship between the driving state and each expert's contribution, without requiring an additional learnable routing network. The predicted yaw error is then used to correct the vehicle heading and reconstruct the dead-reckoning trajectory. The resulting trajectory is subsequently refined using a constant-velocity Kalman filter to improve physical consistency and suppress measurement noise. Owing to its lightweight architecture and deterministic expert-weighting strategy, the proposed framework offers advantages in computational efficiency, interpretability, and real-time applicability compared with existing deep learning-based localization approaches.

The main contributions of this paper are summarized as follows:

- A learning-based yaw-error compensation framework is proposed for dead-reckoning localization using only onboard vehicle signals, namely vehicle velocity and yaw rate.
- A state-dependent expert interpolation strategy is introduced in the velocity–yaw-rate space to capture varying yaw-error characteristics across different driving regimes.
- Expert outputs are combined through a deterministic state-based weighting scheme instead of a learnable gating network, improving interpretability and reducing model complexity while preserving state-dependent compensation capability.

II. FEATURE EXTRACTION AND LEARNING-BASED CNN-LSTM

A. VEHICLE MODEL AND DATA CONFIGURATION

To estimate the vehicle position, a model that represents the vehicle state is required. The vehicle motion on a two-dimensional plane can be approximated using the kinematic bicycle model [17]. The vehicle state at time t is defined as:

$$x_{t+1} = x_t + v_t \cos(\psi_t) \Delta t, \quad (1)$$

$$y_{t+1} = y_t + v_t \sin(\psi_t) \Delta t, \quad (2)$$

$$\psi_{t+1} = \psi_t + \frac{v_t}{l_f + l_r} \tan(\delta_t) \Delta t \quad (3)$$

where x_t and y_t denote the vehicle position, v_t denotes the velocity, δ_t represents the steering angle, ψ_t is the vehicle yaw, Δt denotes the time interval, and l_f and l_r denote the distances from the vehicle center of gravity to the front and rear axles, respectively. To accurately train the error prediction model, it is necessary to first define the structure and format of the data that will be used as inputs to the model.

$$L_t = \begin{bmatrix} \dot{\psi}_t \\ v_t \end{bmatrix}, \quad (4)$$

$$L_{\text{seq}} = \{L_1, \dots, L_T\}, \quad t = 1, \dots, T \quad (5)$$

where L_t denotes the input data at time t , and L_{seq} represents the input window of length T . For deep learning model training, the data are converted into input features and target labels. The model input consists of the time series of vehicle velocity v and yaw rate $\dot{\psi}$ within a window of size T . These two variables serve as key indicators describing the longitudinal and lateral motion of the vehicle. For stable model training, each variable is normalized using the minimum and maximum values computed over the entire training dataset.

The target value to be predicted is the dead reckoning error computed over a window of length T . The target yaw error e_t is computed as follows:

$$\psi_t^{DR} = \psi_{t-1}^{DR} + \dot{\psi}_t \Delta t, \quad (6)$$

$$e_t = \psi_t^{GT} - \psi_t^{DR}, \quad t = 1, \dots, T \quad (7)$$

where ψ_t^{GT} denotes the ground-truth yaw angle, and ψ_t^{DR} represents the yaw angle estimated by dead reckoning [3].

B. CONVOLUTIONAL FEATURE REPRESENTATION LEARNING

Based on the defined input sequence and target yaw error, Fig. 1 provides an overview of the overall yaw-error compensation framework. The input sequence composed of velocity and yaw rate is first processed by a CNN–LSTM network to extract temporal driving features. Since vehicle driving data are continuous time-series signals, CNN is employed in the first stage to capture local temporal patterns associated with yaw-error characteristics and to transform them into high-level feature representations. CNN is effective not only for image processing but also for time-series analysis, as convolutional kernels can extract local patterns and combine them to represent the overall sequence [18]. The extracted temporal features are then delivered to the expert layer, where the final yaw-error estimate is generated for yaw-error compensation and position estimation.

The CNN module of the model receives the time series input window L_{seq} , defined in Section II-A, and employs two convolutional layers. The first convolutional layer uses a small kernel to capture subtle variations that occur within short temporal intervals of the data. Batch normalization and an activation function are then applied to stabilize the training process and introduce nonlinearity. The feature map produced by the first block is passed through a max pooling layer. Max pooling reduces the dimensionality of the feature map to improve computational efficiency while retaining the most salient features, allowing the model to learn representations that are robust to minor variations. Through this process, the temporal length is reduced by half, and more abstract features are emphasized. The second convolutional block then receives the sequence of features extracted in the first stage and combines them in a hierarchical manner. This hierarchical structure allows the model to capture complex driving patterns over a broader temporal context. The time series data that pass through the CNN module are ultimately transformed into a low-dimensional feature sequence in which the essential dynamic characteristics of the driving behavior are compactly represented. This feature sequence, with reduced noise and enhanced salient information compared with the raw data, serves as an optimized input for the LSTM network to learn long-term temporal dependencies in the sequence.

C. SEQUENCE MODELING WITH LSTM NETWORKS

The feature sequence extracted by the CNN module compactly represents local driving patterns at each time step, yet it lacks information on how these patterns evolve and interact over time. Since the yaw error is influenced not only by the current motion but also by the driving history propagated from the past, it is essential to model the long-term temporal dependencies in the sequence. To address

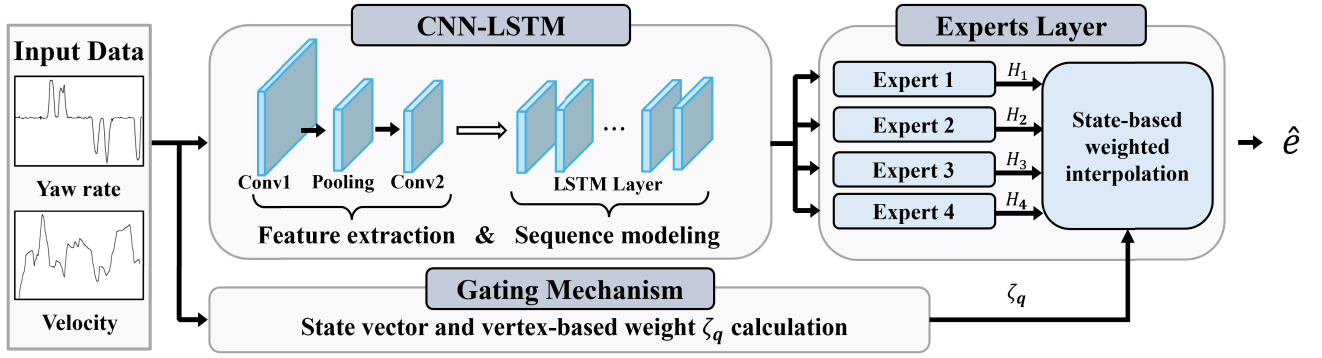


FIGURE 1. Architecture of the proposed yaw-error compensation network. The input sequence composed of vehicle velocity and yaw rate is processed by the CNN-LSTM backbone, and the final yaw error is estimated through state-based weighted interpolation of expert outputs.

this, an LSTM network is employed to capture these long-range dependencies. An LSTM consists of an internal cell state and a set of input, forget, and output gates. Through this gated structure, the network is able to retain important information from the past within the cell state while selectively discarding information that is irrelevant to the current prediction. As a result, LSTM is robust to the vanishing gradient problem commonly observed in conventional RNNs and can effectively capture the complex temporal dynamics of time series data. The equations for each gate of the LSTM can be expressed as follows [19]:

$$i_t = \sigma(W_i F_t + U_i h_{t-1} + b_i), \quad (8)$$

$$f_t = \sigma(W_f F_t + U_f h_{t-1} + b_f), \quad (9)$$

$$o_t = \sigma(W_o F_t + U_o h_{t-1} + b_o), \quad (10)$$

$$\tilde{c}_t = \tanh(W_c F_t + U_c h_{t-1} + b_c), \quad (11)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \quad (12)$$

$$h_t = o_t \odot \tanh(c_t) \quad (13)$$

where σ denotes the sigmoid function, and i_t , f_t , and o_t represent the input, forget, and output gates, respectively. F_t denotes the feature representation obtained from the CNN and is used as the input to the LSTM at time step t , while h_t and c_t denote the hidden state and the cell state, respectively. The matrices W_i , W_f , W_o , and W_c denote the weights applied to the input, whereas U_i , U_f , U_o , and U_c represent the weights applied to the hidden state. The terms b_i , b_f , b_o , and b_c correspond to the biases added to each linear transformation.

In the proposed model, the feature sequence with temporal structure generated by the CNN module is fed into a two-layer LSTM network. The LSTM processes the sequence step by step, propagating a hidden state that summarizes information from previous time steps. Through this mechanism, the network can capture complex driving contexts in which temporal ordering is crucial, such as steering behavior following a rapid deceleration. Ultimately, the LSTM layer produces a temporal feature sequence in which past contextual information is embedded across time, and this representation is used as the input to the expert layer.

Based on this shared temporal representation, the expert layer generates yaw-error estimates for weighted interpolation. For online compensation, only the final output corresponding to the most recent step is used as the effective yaw-error prediction.

III. WEIGHT INTERPOLATION WITH FILTERING

A. STATE-BASED WEIGHTED INTERPOLATION

MoE is a framework that decomposes a complex problem into multiple experts and improves prediction performance by taking a weighted combination of the outputs from the experts that are most suitable for a given input. In the conventional setting, a shared backbone network is coupled with several experts, and a router probabilistically determines the contribution of each expert based on the characteristics of the input [20], [21]. This structure has the advantage of effectively partitioning highly nonlinear problems and significantly enhancing the representational capacity of the model. Nevertheless, probabilistic routing introduces an additional learnable module, which may increase training overhead and make the expert selection process less transparent.

In the proposed model, the benefits of MoE are preserved while reducing the instability and training overhead associated with probabilistic routing. The probabilistic router is replaced with a deterministic state-based gating mechanism defined using the vehicle velocity v and yaw rate $\dot{\psi}$. By identifying the driving condition from v and $\dot{\psi}$ and combining the outputs of the experts through weighted interpolation, the model achieves a more stable and interpretable prediction structure. Let the parameters be defined as $\theta_1 = |\dot{\psi}|$ and $\theta_2 = v$ then the formulation can be expressed as follows:

$$|\dot{\psi}| = \theta_1 \in [\underline{\theta}_1 \ \bar{\theta}_1] = [\dot{\psi}_{\min} \ \dot{\psi}_{\max}], \quad (14)$$

$$v = \theta_2 \in [\underline{\theta}_2 \ \bar{\theta}_2] = [v_{\min} \ v_{\max}]. \quad (15)$$

The vertices, denoted by $\mathbf{V}$, serve as reference points for computing the expert mixture weights and are defined as follows:

$$\mathbf{V} = [V_1 \ V_2 \ V_3 \ V_4] = \begin{bmatrix} \underline{\theta}_1 & \underline{\theta}_1 & \bar{\theta}_1 & \bar{\theta}_1 \\ \underline{\theta}_2 & \underline{\theta}_2 & \underline{\theta}_2 & \bar{\theta}_2 \end{bmatrix} \quad (16)$$

where V_1, V_2, V_3 , and V_4 denote the four vertices. The four vertices represent the combinations of the minimum and maximum absolute-yaw-rate and vehicle-velocity states. Accordingly, four experts are employed to associate one local yaw-error predictor with each vertex of the two-dimensional operating-state space. To prevent the different physical scales of absolute yaw rate and vehicle velocity from disproportionately affecting the distance calculation, the current-state and vertex coordinates are independently min-max normalized to $[0, 1]$ using the parameter bounds defined in (14) and (15). Let D_q denote the Euclidean distance between the normalized current state and the normalized q -th vertex, and let $D_{\text{sum}} = \sum_{i=1}^4 D_i$. The expert mixture weights are then computed as follows:

$$\xi_q = \frac{D_{\text{sum}}/D_q}{\sum_{i=1}^4 (D_{\text{sum}}/D_i)}, \quad q = 1, \dots, 4 \quad (17)$$

$$\sum_{q=1}^4 \xi_q = 1, \quad \xi_q \geq 0 \quad (18)$$

The term ξ_q represents the expert mixture weight corresponding to each vertex. If the current state coincides with the q -th vertex such that $D_q = 0$, the corresponding expert weight is set to $\xi_q = 1$, while the remaining weights are set to zero. This state-based gating mechanism enables an interpretable assignment of which expert becomes active under a given state.

B. EXPERT LAYER INTERPOLATION FOR ERROR PREDICTION

The four experts operate as parallel predictors that receive the shared temporal feature representation generated by the CNN-LSTM backbone. All four expert heads have the same architecture but possess independent trainable parameters. Each expert is implemented as a fully connected linear layer that maps the 64-dimensional LSTM feature at each time step to a scalar yaw-error estimate. During training, the learning signal received by each expert is modulated by its state-dependent mixture weight. Consequently, an expert associated with a vertex closer to the current state receives a stronger learning signal, encouraging the experts to learn different local yaw-error mappings despite having identical architectures.

The interpretability of the experts can be understood in terms of the proximity of the current state coordinates $(|\dot{\psi}_t|, v_t)$ to the four vertices in the $(|\dot{\psi}|, v)$ plane. For each vertex indexed by q , the corresponding q -th expert is associated with that operating region, and the mixture weight ξ_q is determined by the proximity of $(|\dot{\psi}_t|, v_t)$ to the q -th vertex, thereby controlling the contribution of the q -th expert output in the final prediction. This approach inherits the advantages of interpolation-based methods that have been well validated in control applications [22], [23] while achieving real-time applicability and interpretability without relying on a learnable router. Using the expert mixture weights ξ_q computed from the four vertices, the final

prediction can be obtained as follows:

$$\hat{e}_t = \sum_{q=1}^4 \xi_q H_q \quad (19)$$

where $\hat{e}_t$ denotes the predicted yaw error at time step t , and H_q denotes the scalar yaw-error estimate produced by the q -th expert. The predicted $\hat{e}_t$ is used to compute the corrected yaw angle employed for vehicle position estimation.

$$\hat{\psi}_t = \psi_t^{DR} + \hat{e}_t \quad (20)$$

where $\hat{\psi}_t$ denotes the compensated heading after applying the predicted yaw-error correction. The corrected yaw angle is then used in the dead reckoning process to compute the vehicle position x_t and y_t . The corrected dead-reckoning position is obtained by integrating the corrected heading with the vehicle velocity:

$$\hat{x}_{t+1} = \hat{x}_t + v_t \cos(\hat{\psi}_t) \Delta t, \quad (21)$$

$$\hat{y}_{t+1} = \hat{y}_t + v_t \sin(\hat{\psi}_t) \Delta t. \quad (22)$$

C. POSITION ESTIMATION WITH KALMAN FILTER

The heading corrected using the predicted yaw error is integrated with the vehicle velocity in (21) and (22) to generate the corrected dead-reckoning trajectory. To suppress local discontinuities and high-frequency fluctuations in the resulting position estimates, a constant-velocity Kalman filter is applied as the final post-processing stage [24]. The state space model of the Kalman filter can be expressed as follows:

$$X_t = [x_t \ y_t \ \dot{x}_t \ \dot{y}_t]^\top \quad (23)$$

where X_t denotes the state vector at time t , x and y represent the vehicle position on the two dimensional plane, and $\dot{x}$ and $\dot{y}$ correspond to the vehicle velocity components. The Kalman filter adopts a constant velocity model, and the discrete time state space representation can be expressed as follows:

$$X_{t+1} = \Phi X_t + w_t. \quad (24)$$

The state transition matrix Φ is given by:

$$\Phi = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad w_t \sim \mathcal{N}(0, Q_t) \quad (25)$$

where w_t denotes the process noise and Q_t is the process noise covariance matrix. The corrected yaw $\hat{\psi}_t$ and the velocity v_t are used to compute the corrected dead-reckoning-based position $(x_t^{\text{corr}}, y_t^{\text{corr}})$ at each time step. The resulting dead-reckoning trajectory is then used as the measurement input to the Kalman filter, and the measurement vector is defined as:

$$Y_t = \begin{bmatrix} \hat{x}_t \\ \hat{y}_t \end{bmatrix}. \quad (26)$$

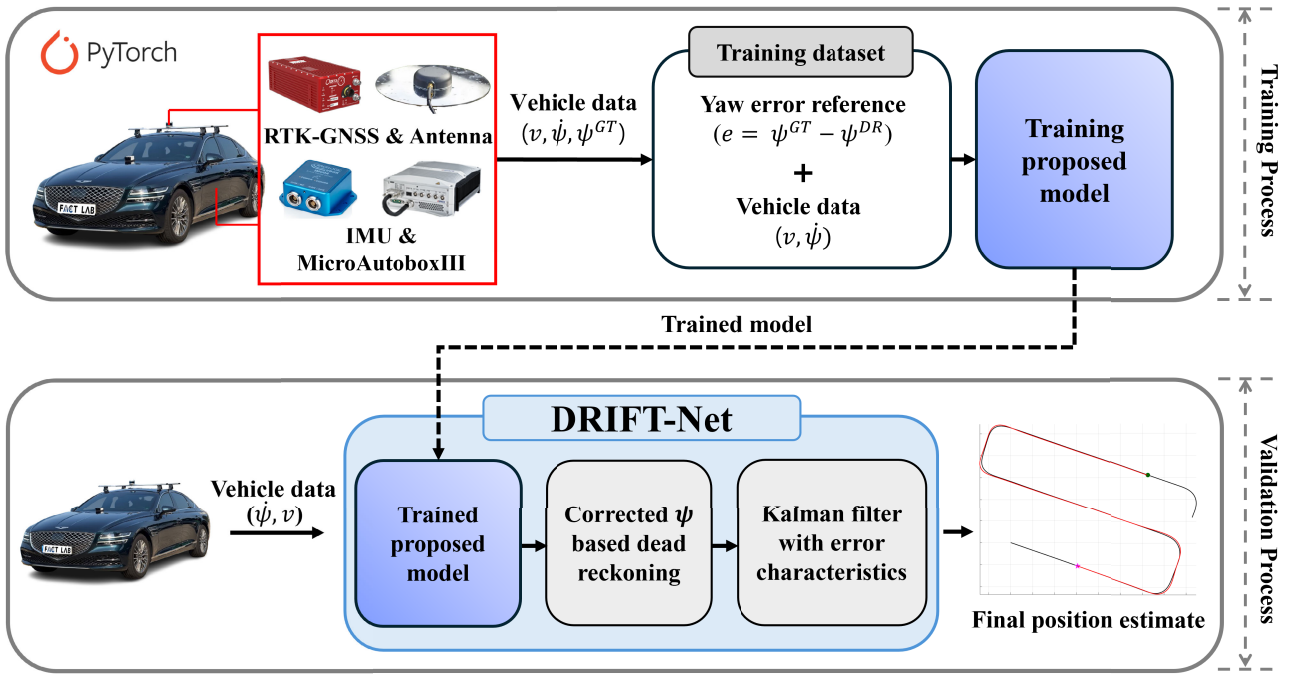


FIGURE 2. Training and validation workflow of the proposed method. In the training stage, vehicle data are used to construct yaw-error references and to train the proposed network. In the validation stage, the trained model predicts yaw error for dead-reckoning correction, and the corrected trajectory is further refined by a Kalman filter.

The measurement model can be expressed as:

$$Y_t = CX_t + v_t \quad (27)$$

where

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \quad v_t \sim \mathcal{N}(0, R_t) \quad (28)$$

where C denotes the measurement matrix, v_t represents the noise included in the measurement process, and R_t denotes the measurement noise covariance.

The error characteristics in the corrected dead-reckoning trajectory can be propagated to the state estimation results, and these characteristics may vary depending on the driving condition. The covariance of the corrected position error with respect to the ground truth is analyzed from the measurement data and used as the measurement noise covariance R_t in the Kalman filter to reduce measurement errors and improve estimation accuracy. The process noise covariance Q_t is tuned following the approaches in [25] and [26], whereas the measurement noise covariance R_t is determined from the error statistics described above. Through this procedure, the Kalman filter mitigates small discontinuities that may arise when the deep learning based correction is applied and suppresses high frequency noise possibly contained in the model output, thereby producing a physically smooth and stable final trajectory.

IV. APPLICATION RESULTS

A. EXPERIMENT SETUP

Fig. 2 illustrates the training and validation workflow used in this research. As shown in the figure, vehicle data are

first used to generate the yaw-error reference, and the resulting input–target pairs are then employed for network training, validation, and trajectory reconstruction. The experiments were conducted using data acquired through the vehicle’s Controller Area Network (CAN) communication. The training dataset consists of vehicle velocity v and yaw rate $\dot{\psi}$ sampled at 100 Hz. The ground-truth yaw angle ψ^{GT} was obtained from the heading output of an Oxts RT2002 dual-antenna RTK-GNSS/INS system. Two GNSS antennas were rigidly mounted on the vehicle with a fixed baseline, and RTK differential corrections were provided by an RT-Base station. The system operated at 100 Hz, corresponding to a sampling period of 10 ms. The mounting offset between the RT2002 coordinate frame and the longitudinal axis of the vehicle was calibrated and compensated during system setup. All measurements were synchronized and processed using a dSPACE MicroAutoBox III. The driving dataset was collected over multiple road sections containing straight segments, gradual curves, and repeated turning maneuvers under both low- and high-speed driving conditions. Accordingly, the dataset captures variations in vehicle velocity, yaw rate, and steering behavior across different driving regimes. In total, approximately 360,000 synchronized samples were collected at a sampling frequency of 100 Hz and divided into training, validation, and test sets at a ratio of 6:2:2. In addition, the two evaluation scenarios described in Section IV-B were obtained from independent driving sessions and were not included in the training dataset. The window length was set to $T = 100$ samples, corresponding to 1 s of vehicle data. The reference vertices



(a) Experimental route in Scenario 1



(b) Experimental route in Scenario 2

FIGURE 3. Representative experimental environments used for the two evaluation scenarios: (a) Scenario 1 under low-speed driving conditions with multiple steering segments and (b) Scenario 2 under relatively high-speed driving conditions with long straight and gradual-curvature sections.

used to compute the expert weights were defined as follows:

$$[\underline{\theta}_1 \ \bar{\theta}_1] = [\dot{\psi}_{\min} \ \dot{\psi}_{\max}] = [0 \text{ rad/s} \ 0.45 \text{ rad/s}], \quad (29)$$

$$[\underline{\theta}_2 \ \bar{\theta}_2] = [v_{\min} \ v_{\max}] = [0 \text{ m/s} \ 18 \text{ m/s}] \quad (30)$$

where each parameter range represents the minimum and maximum values within the learning environment.

The model was trained using the PyTorch framework on an NVIDIA RTX 3060 GPU. The Mean Squared Error (MSE) between the predicted yaw error and the ground-truth yaw error was employed as the loss function:

$$\mathcal{L} = \frac{1}{N} \sum_{n=1}^N (\hat{e}_n - e_n)^2 \quad (31)$$

where $\mathcal{L}$ denotes the training loss, N is the number of samples in a mini-batch, and $\hat{e}_n$ and e_n denote the predicted and

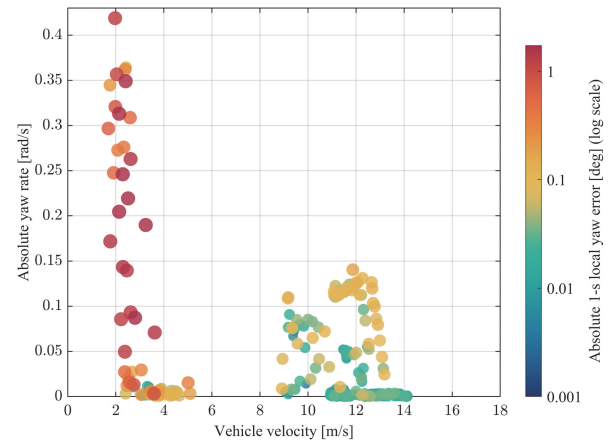


FIGURE 4. Combined operating-state distribution of the absolute 1-s local yaw error across both evaluation scenarios.

ground-truth yaw errors of the n -th sample, respectively. AdamW was used as the optimizer [27].

B. SCENARIO-BASED EXPERIMENT

The proposed method was evaluated using multiple driving datasets exhibiting distinct dynamic characteristics. Each scenario contains different driving patterns, speed variations, and steering behaviors, and the yaw error compensation performance was assessed individually under GNSS outage conditions. In the experiments, a complete loss of GNSS signals was assumed from a specified time instant, and the predicted yaw error $\hat{e}_t$ was used to correct the dead reckoning based yaw estimate in real time during the outage. The corrected yaw and velocity were then used to reconstruct the dead reckoning trajectory at every sampling step, followed by Kalman filtering to refine the trajectory and ensure physical consistency. The computational performance of the proposed framework was evaluated on the same system using a batch size of one. The end-to-end one-step processing time was measured from the update of the vehicle-velocity and yaw-rate input buffer to the output of the constant-velocity Kalman filter. The first 100 inference steps were excluded as warm-up steps, and the processing-time statistics were evaluated separately for each test scenario.

Two evaluation scenarios were considered under distinct driving conditions. Scenario 1 was conducted under low-speed driving conditions at the autonomous driving test track of Chungbuk National University, and Scenario 2 was conducted under relatively high-speed driving conditions at the proving ground of the Korea Intelligent Automotive Parts Promotion Institute (KIAPI). Fig. 3 shows the experimental routes used in the two evaluation scenarios. For each scenario, the reconstructed trajectory over the designated evaluation interval was compared with several baseline methods, including pure dead reckoning without any correction, a CNN-LSTM model without the expert layer, an Autoformer model [28], and an EKF-based

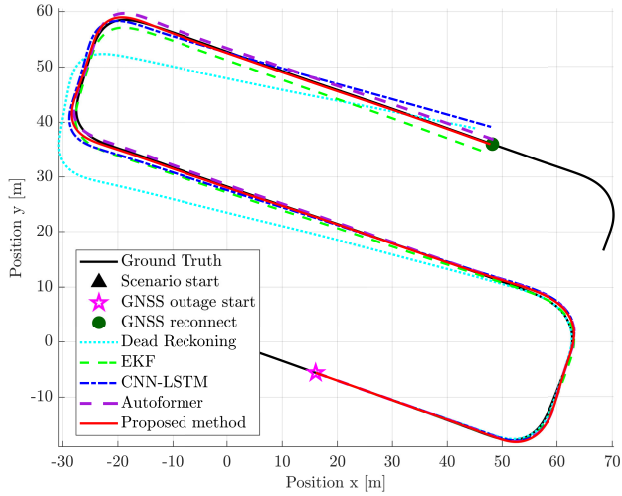


FIGURE 5. Estimated trajectories in Scenario 1 under GNSS outage conditions. The trajectories obtained by dead reckoning, the EKF-based method, the CNN-LSTM model, the Autoformer model, and the proposed method are compared with the ground-truth path.

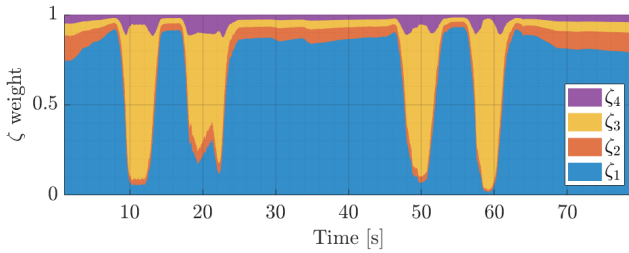


FIGURE 6. Expert mixture weights (Scenario 1).

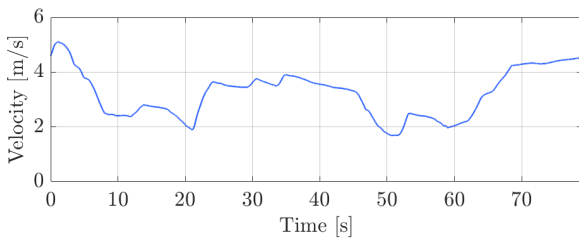


FIGURE 7. Input vehicle velocity (Scenario 1).

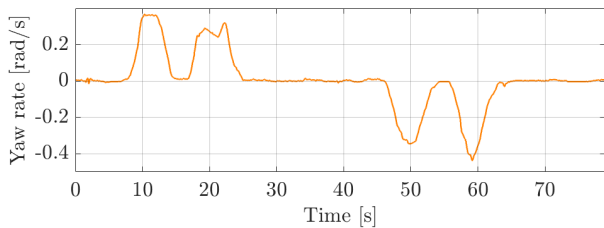


FIGURE 8. Input vehicle yaw rate (Scenario 1).

method in which yaw-rate bias is included in the state and recursively estimated, while position measurements are

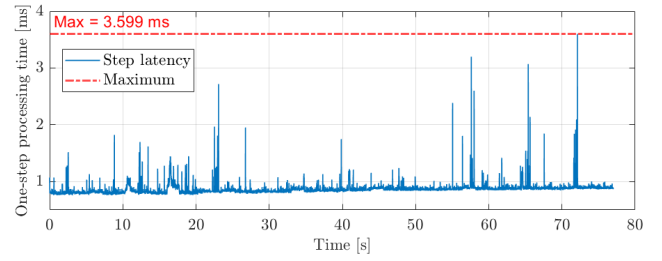


FIGURE 9. Per-step end-to-end processing time (Scenario 1).

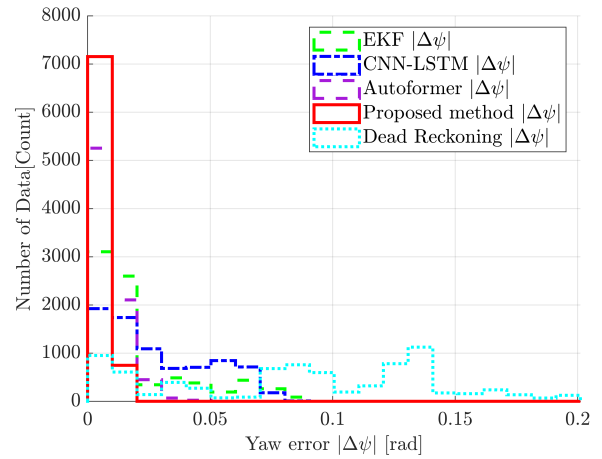


FIGURE 10. Yaw error histogram (Scenario 1).

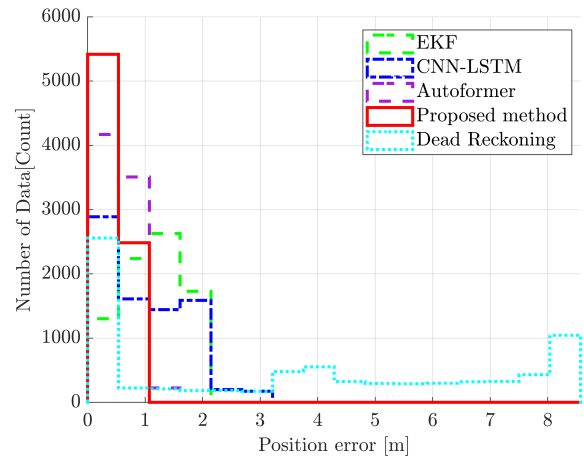


FIGURE 11. Position error histogram (Scenario 1).

used for correction during GNSS-available intervals. The trajectories obtained by these methods, as well as by the proposed model, were compared against the ground-truth trajectory reconstructed from RTK-GNSS measurements, and quantitative performance evaluation was conducted. To examine the operating-state dependence of yaw error, the uncorrected dead-reckoning yaw error was evaluated over non-overlapping 1-s windows, with the dead-reckoning yaw

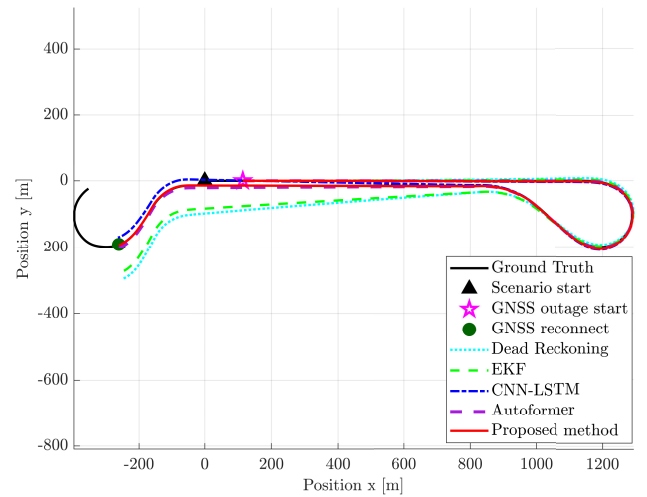
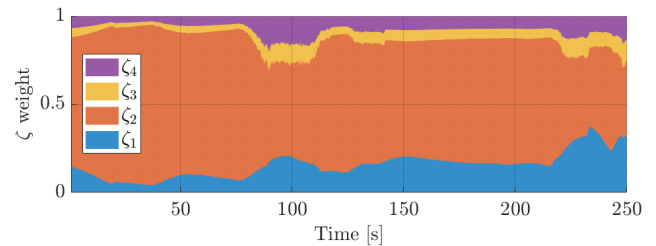
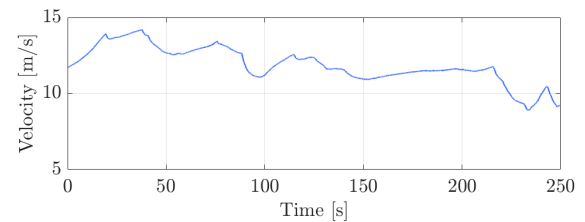
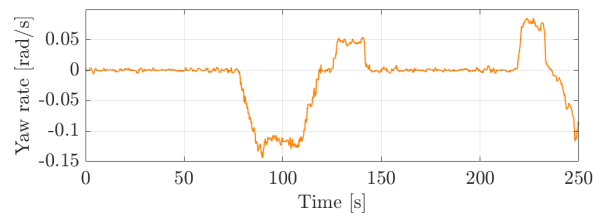
TABLE 1. Results and Error Comparison Under GNSS Outage Driving Condition (Scenario 1).

	Dead Reckoning	EKF	CNN-LSTM	Autoformer	Proposed Method
Yaw MAE [deg]	4.993	1.821	1.636	0.490	0.273
Yaw RMSE [deg]	5.852	2.229	2.048	0.631	0.339
ATE-RMSE [m]	4.829	1.838	1.233	0.571	0.484
Final position error [m]	4.116	1.373	3.141	0.856	0.301

initialized from the ground-truth yaw at the beginning of each window. The samples from both evaluation scenarios were combined in the velocity–absolute-yaw-rate space. As shown in Fig. 4, relatively large local yaw errors are mainly observed in low-speed states, particularly at higher absolute yaw rates, whereas the errors are generally smaller in higher-speed and lower-yaw-rate states. These observations indicate that the local yaw-error characteristics vary across vehicle operating states [17].

1) SCENARIO 1

The first scenario corresponds to a low-speed driving environment that includes multiple steering segments, and a GNSS outage duration of 80 seconds was assumed for evaluation. Fig. 5 presents the estimated trajectories during the outage period in Scenario 1, comparing the paths obtained from pure dead reckoning, the EKF-based method, the CNN-LSTM model, the Autoformer model, and the proposed method. The proposed method effectively suppresses accumulated position errors and yields a trajectory that remains closest to the ground-truth path. Fig. 6 illustrates the temporal evolution of the expert weights ζ_q , providing insight into how the contributions of the four experts shift according to the driving state. In particular, ζ_3 becomes relatively dominant during sharp turns and low-speed segments, confirming that the state-based gating mechanism selectively emphasizes the corresponding expert based on the state coordinates. Figs. 7 and 8 show the input velocity v and yaw rate $\dot{\psi}$ profiles used for the evaluation, highlighting the driving conditions and steering patterns present in this scenario. Fig. 9 shows the temporal variation of the end-to-end one-step processing time in Scenario 1. The average processing time was 0.862 ms, while the maximum observed processing time reached 3.599 ms. Fig. 10 depicts the histogram of the absolute yaw errors between the ground-truth yaw ψ and the yaw estimates corrected by each method during the outage period, while Fig. 11 compares the distribution of the corresponding position errors. In both histograms, the proposed method exhibits a higher concentration in the low error region compared to the other approaches. The quantitative evaluation results for Scenario 1 are summarized in Table 1. In the low-speed driving environment, the proposed method achieves significant reductions in the yaw RMSE, MAE, absolute trajectory error-RMSE (ATE-RMSE), and final position error compared to the conventional dead reckoning approach.

**FIGURE 12. Estimated trajectories in Scenario 2 under GNSS outage conditions. The trajectories obtained by dead reckoning, the EKF-based method, the CNN-LSTM model, the Autoformer model, and the proposed method are compared with the ground-truth path.****FIGURE 13. Expert mixture weights (Scenario 2).****FIGURE 14. Input vehicle velocity (Scenario 2).****FIGURE 15. Input vehicle yaw rate (Scenario 2).**

2) SCENARIO 2

The second scenario utilizes a high-speed driving dataset, where a GNSS outage duration of 250 seconds was assumed. Fig. 12 illustrates the estimated trajectories during the outage

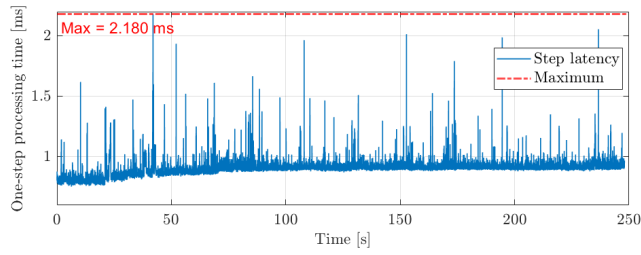


FIGURE 16. Per-step end-to-end processing time (Scenario 2).

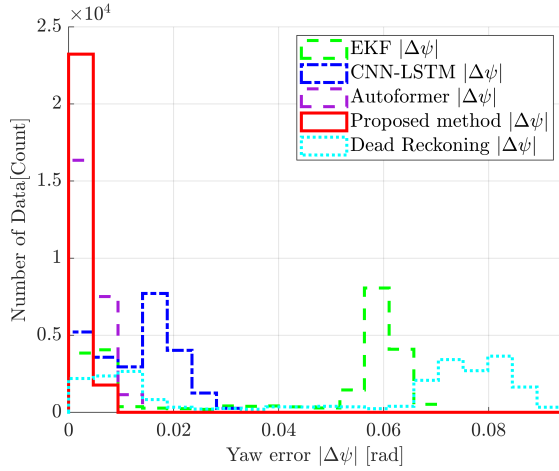


FIGURE 17. Yaw error histogram (Scenario 2).

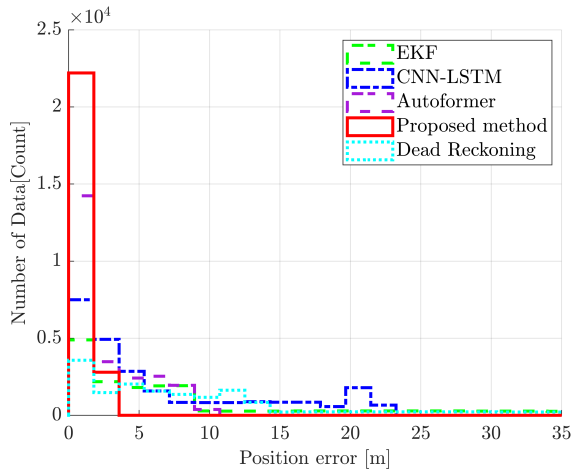


FIGURE 18. Position error histogram (Scenario 2).

interval in Scenario 2, demonstrating that the proposed method effectively suppresses long-term error even under high-speed conditions. Despite the substantially longer outage period compared to Scenario 1, the trajectory reconstructed by the proposed method remains closely aligned with the ground-truth path. Fig. 13 shows the temporal variation of the expert weights ζ_q for the high-speed scenario, where ζ_2 tends to take relatively larger values than the other weights.

TABLE 2. Results and Error Comparison Under GNSS Outage Driving Condition (Scenario 2).

	Dead Reckoning	EKF	CNN-LSTM	Autoformer	Proposed Method
Yaw MAE [deg]	2.923	2.235	0.739	0.202	0.116
Yaw RMSE [deg]	3.458	2.679	0.856	0.267	0.148
ATE-RMSE [m]	45.774	37.889	9.564	3.709	1.171
Final position error [m]	105.45	84.87	22.23	9.256	0.607

This indicates that the corresponding expert contributes more significantly under high-speed, near-straight driving conditions. Figs. 14 and 15 present the velocity v and yaw rate $\dot{\psi}$ profiles used in Scenario 2, confirming the characteristic high-speed behavior with higher average velocities and smaller or more gradual yaw-rate variations compared to Scenario 1. Fig. 16 shows the temporal variation of the end-to-end one-step processing time in Scenario 2. The average processing time was 0.901 ms, while the maximum observed processing time was 2.180 ms. The histogram comparisons in Figs. 17 and 18 further show that the proposed method yields a greater concentration of samples within low error regions than the other approaches. The quantitative results for Scenario 2 are summarized in Table 2. Even with the extended GNSS outage duration, the proposed method consistently outperformed dead reckoning, the EKF, the CNN-LSTM, and the Autoformer model. In particular, it achieved an ATE-RMSE of 1.171 m and a final position error of 0.607 m, demonstrating effective suppression of accumulated localization error during the prolonged GNSS outage.

V. CONCLUSION

This paper presented DRIFT-Net, a learning-based dead-reckoning correction framework for vehicle localization in localization-degraded environments. By combining a CNN-LSTM feature extractor with a deterministic state-dependent mixture-of-experts architecture, the proposed framework estimates yaw error using only onboard vehicle signals, namely velocity and yaw rate. The predicted yaw error is used to correct the dead-reckoning heading, while a Kalman filtering stage further improves temporal consistency by suppressing high-frequency noise. Experimental results obtained from real-world driving scenarios involving various speed profiles and driving maneuvers demonstrated consistent improvements in both yaw estimation accuracy and trajectory tracking performance compared with conventional dead-reckoning approaches. The proposed framework achieved yaw RMSE values of 0.339 deg and 0.148 deg under low-speed and high-speed driving conditions, respectively. These results indicate that DRIFT-Net can effectively mitigate error accumulation during prolonged GNSS outages while maintaining computational efficiency and model interpretability through its deterministic state-dependent expert-weighting strategy.

REFERENCES

- [1] I. Skog and P. Handel, "In-car positioning and navigation technologies—A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 1, pp. 4–21, Mar. 2009.
- [2] S. Thrun et al., "Stanley: The robot that won the DARPA grand challenge," *J. Field Robot.*, vol. 23, no. 9, pp. 661–692, 2006.
- [3] P. D. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*, 2nd ed., Norwood, MA, USA: Artech House, 2013.
- [4] M. S. Grewal, L. R. Weill, and A. P. Andrews, *Global Positioning Systems, Inertial Navigation, and Integration*, 2nd ed., Hoboken, NJ, USA: Wiley, 2007.
- [5] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 3565–3572.
- [6] J. S. Kim, W. Y. Choi, Y. W. Jeong, and C. C. Chung, "Vehicle localization using radar calibration with disconnected GPS," in *Proc. 21st Int. Conf. Control, Autom. Syst. (ICCAS)*, Oct. 2021, pp. 1371–1376.
- [7] M. Elkholy, M. Elsheikh, and N. El-Sheimy, "Radar/INS integration and map matching for land vehicle navigation in urban environments," *Sensors*, vol. 23, no. 11, p. 5119, May 2023.
- [8] J. H. Yang, W. Y. Choi, and C. C. Chung, "StrikeNet: Deep convolutional LSTM-based road lane reconstruction with spatiotemporal inference for lane keeping control," *IEEE Access*, vol. 12, pp. 97500–97514, 2024.
- [9] V. Rausch, A. Hansen, E. Solowjow, C. Liu, E. Kreuzer, and J. K. Hedrick, "Learning a deep neural net policy for end-to-end control of autonomous vehicles," in *Proc. Amer. Control Conf. (ACC)*, May 2017, pp. 4914–4919.
- [10] H.-F. Dai, H.-W. Bian, R.-Y. Wang, and H. Ma, "An INS/GNSS integrated navigation in GNSS denied environment using recurrent neural network," *Defence Technol.*, vol. 16, no. 2, pp. 334–340, Apr. 2020.
- [11] R. Clark, S. Wang, H. Wen, A. Markham, and N. Trigoni, "VINet: Visual-inertial odometry as a sequence-to-sequence learning problem," in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, 2017, pp. 3995–4001.
- [12] K. He, H. Ding, N. Xu, and K. Guo, "Wheel odometry with deep learning-based error prediction model for vehicle localization," *Appl. Sci.*, vol. 13, no. 9, p. 5588, Apr. 2023.
- [13] M. Brossard, A. Barrau, and S. Bonnabel, "AI-IMU dead-reckoning," *IEEE Trans. Intell. Vehicles*, vol. 5, no. 4, pp. 585–595, Dec. 2020.
- [14] X. Meng, H. Tan, P. Yan, Q. Zheng, G. Chen, and J. Jiang, "A GNSS/INS integrated navigation compensation method based on CNN-GRU + IRAKF hybrid model during GNSS outages," *IEEE Trans. Instrum. Meas.*, vol. 73, pp. 1–15, 2024.
- [15] Z. Wang, H. Liu, G. Guo, and M. Han, "An enhanced vehicle self-positioning method during GNSS outages using factor graph optimization and CNN-LSTM-attention," *IEEE Trans. Instrum. Meas.*, vol. 74, pp. 1–11, 2025.
- [16] R. Xiong, C. Chen, J. Xu, X. Dong, and J. Pu, "Multi-AAV-assisted on-demand charging in dense wireless rechargeable sensor networks," *IEEE Internet Things J.*, vol. 12, no. 7, pp. 8821–8834, Apr. 2025.
- [17] R. Rajamani, *Vehicle Dynamics and Control*, 2nd ed., Cham, Switzerland: Springer, 2012.
- [18] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
- [19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.
- [20] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Comput.*, vol. 3, no. 1, pp. 79–87, Mar. 1991.
- [21] M. I. Jordan and R. A. Jacobs, "Hierarchical mixtures of experts and the EM algorithm," *Neural Comput.*, vol. 6, no. 2, pp. 181–214, Mar. 1994.
- [22] W. Y. Choi and C. C. Chung, "Optimizing look-ahead distance for vehicle lateral control: A controller-adaptive approach," *IEEE Trans. Intell. Vehicles*, vol. 10, no. 9, pp. 4583–4594, Sep. 2025.
- [23] W. Y. Choi, D. J. Kim, C. M. Kang, S.-H. Lee, and C. C. Chung, "Autonomous vehicle lateral maneuvering by approximate explicit predictive control," in *Proc. Annu. Amer. Control Conf. (ACC)*, Jun. 2018, pp. 4739–4744.
- [24] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, Mar. 1960.
- [25] W. Y. Choi, S.-H. Lee, and C. C. Chung, "On-road object collision point estimation by radar sensor data fusion," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 14753–14763, Sep. 2022.
- [26] S. H. Lee and W. Y. Choi, "Enhancing object estimation by camera-LiDAR sensor fusion using IMM-KF with error characteristics in autonomous robot systems," *IEEE Access*, vol. 12, pp. 197247–197258, 2024.
- [27] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–8.
- [28] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 34, 2021, pp. 22419–22430.



JAEE WON JEONG (Graduate Student Member, IEEE) received the B.S. degree in control and instrumentation engineering from Pukyong National University, Busan, South Korea, in 2025, where he is currently pursuing the M.S. degree in intelligent robot engineering.

His research interests include autonomous vehicle, machine learning, localization, and error compensator. He is a member of Korean Society of Automotive Engineers (KSAE) and the Institute of Control, Robotics, and Systems (ICROS).



YING SHUAI QUAN (Member, IEEE) received the B.S. degree in control science and engineering from Harbin Institute of Technology, Harbin, China, in 2017, and the Ph.D. degree in electrical engineering from Hanyang University, Seoul, South Korea, in 2024.

She is currently a Postdoctoral Researcher with the Department of Electrical Engineering, Chalmers University of Technology, Göteborg, Sweden. Her main research interests include optimal control, learning-based control, and autonomous driving. She is a member of the IEEE Intelligent Transportation Systems Society (ITSS), the IEEE Control System Society (CSS), Korean Society of Automotive Engineers (KSAE), and the Institute of Control, Robotics, and Systems (ICROS).



WOO YOUNG CHOI (Member, IEEE) received the B.S. degree in control and instrumentation engineering from Pukyong National University, Busan, South Korea, in 2015, and the Ph.D. degree in electrical engineering from Hanyang University, Seoul, South Korea, in 2021.

From 2021 to 2022, he was with Korea Institute of Science and Technology, Seoul. From 2022 to 2023, he was an Assistant Professor with the Department of Automotive System Engineering, Keimyung University, Daegu, South Korea. In 2023, he joined the Department of Control and Instrumentation Engineering, Pukyong National University, as a Faculty Member. His research interests include control theory, model predictive control, optimal control, estimation, artificial intelligence, autonomous vehicles, and mobile robots. He is a member of the IEEE Industrial Electronics Society, the IEEE Intelligent Transportation Systems Society, the IEEE Control Systems Society, the Society of Automotive Engineers, Korean Society of Automotive Engineers, and the Institute of Control, Robotics, and Systems. He was awarded the Best Dissertation Award for his doctoral thesis at Hanyang University, in 2021, and received the Outstanding Young Researcher Award at the Institute of Control, Robotics, and Systems, in 2023.

• • •