POLITECNICO DI TORINO
Repository ISTITUZIONALE

Positioning and tracking enhancement through a spiking Legendre memory unit

Original
Positioning and tracking enhancement through a spiking Legendre memory unit / Tilocca, S., Fra, V., Pignata, A., Macii, E., Urgese, G.. - In: IFAC PAPERSONLINE. - ISSN 2405-8971. - ELETTRONICO. - 59 (26):(2025), pp. 235-240. (7th IFAC Conference on Intelligent Control and Automation Sciences (ICONS 2025) Padua (ITA) September 15-18, 2025) [10.1016/j.ifacol.2025.12.040].

Availability:
This version is available at: 11583/3003699 since: 2025-10-06T14:12:00Z

Publisher:
Elsevier

Published
DOI:10.1016/j.ifacol.2025.12.040

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

19 November 2026

Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 59-26 (2025) 235–240

Positioning and tracking enhancement through a spiking Legendre memory unit

Salvatore Tilocca * **Vittorio Fra** * **Andrea Pignata** *
Enrico Macii * **Gianvito Urgese** *

* *Politecnico di Torino, Turin, 10129, Italy*
(e-mail: {name}.{surname}@polito.it; andrea_pignata@polito.it)

Abstract: By coupling State Space Models (SSMs) with Spiking Neural Networks (SNNs), the potential of neuromorphic solutions for positioning and tracking can be successfully explored. A spiking SSM based on the Legendre Memory Unit (LMU) is shown to effectively predict the state of an Extended Kalman Filter (EKF) using Inertial Measurement Unit (IMU) and Global Positioning System (GPS) inputs, with memory reduction up to 96% compared to Deep Learning (DL) solutions. Additionally, such SNN-based architecture enables GPS error correction through data fusion of IMUs, wheel encoders, and gyroscopes, highlighting the efficiency of spike-based solutions over traditional models.

Copyright © 2025 The Authors. This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0/)

Keywords: Biologically plausible neural networks, Spiking Neural Network, Sensor Fusion, Dead Reckoning

1. INTRODUCTION

In the current era of Artificial Intelligence (AI), relying on Moore's Law for scaling and von Neumann architectures for computation is increasingly unsustainable, as these approaches fall short of growing demands for energy-efficient, high-performance processing (Peper, 2017). Neuromorphic computing, particularly through Spiking Neural Networks (SNNs) — the third generation of Artificial Neural Networks (ANNs) (Maass, 1997) — offers a promising alternative. SNNs use discrete spikes to process and transmit information, driving inspiration from biological neurons. Such bio-inspired principle allows SNNs to process information only when events, i.e. the spikes, occur, thus resulting in a more efficient energy use. This sparse and event-driven approach, along with local and parallel processing, significantly reduces power consumption, making SNNs well-suited for real-time, energy-efficient performance (Eshraghian et al., 2023).

In autonomous driving, accurate and energy-efficient positioning and tracking are crucial. For the former, while Global Positioning System (GPS) technologies like Real-Time Kinematics (RTK) perform well in open areas, their reliability drops significantly in urban environments due to signal degradation (Georgiadou and Kleusberg, 1988); and dead reckoning, though a viable alternative using inertial data, often depends on costly sensors (Akai et al., 2017). Position tracking, instead, typically involves sensor fusion through methods like Kalman Filter (KF), Extended Kalman Filters (EKFs), or hybrid models incorporating Deep Learning (DL) like Neural-KF (Du et al., 2023).

In this work, we explore the adoption of SNN-based neuromorphic solutions to address some of the limitations the standard solutions exhibit. Specifically, we evaluate standalone low-cost Inertial Measurement Unit (IMU) sensors and multi-sensor setups, applying SNNs for dead reckoning to improve accuracy and efficiency. We examine two use cases by relying on the LIF-based Legendre Memory Unit (L^2MU) (Fra et al., 2024), a natively neuromorphic redesign of the state space model defined by the Legendre Memory Unit (LMU) (Voelker et al., 2019). In a first use case, the L^2MU enhances EKF-based position estimation, and, in a second scenario, it is employed to predict vehicle position in complex, multi-sensor environments. In both cases, accuracy and efficiency improvements are reported: the L^2MU achieves around 90% model size reduction compared to Neural-KF in the first case, and it improves positioning accuracy by 50–70% over GPS alone in the second, depending on sensor configuration.

2. BACKGROUND

2.1 Legendre Memory Unit (LMU)

The LMU (Voelker et al., 2019) is a recurrent architecture relying on the shifted Legendre polynomials to orthogonalize an input signal over a sliding window of length θ, which helps represent long-term dependencies and high-frequency components effectively. The core of the LMU is its memory cell, which stores the input signal $u(t)$ over a sliding window using Legendre polynomials. The dynamics of the memory are governed by the ordinary differential equation (ODE), as defined in (1):

$$\theta \dot{m}(t) = Am(t) + Bu(t) \tag{1}$$

where $m(t)$ represents the memory state vector, while A and B constitute the specific state space matrices (Voelker et al., 2019).
The LMU cell takes an input x_t and computes the hidden state h_t through a nonlinear function, which depends on the current input, the previous hidden state, and the memory m_t, as described in (2):

2405-8963 Copyright © 2025 The Authors. This is an open access article under the CC BY-NC-ND license.
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2025.12.040

$$h_t = f(W_x x_t + W_h h_{t-1} + W_m m_t) \quad (2)$$

where W_x, W_h, and W_m are learnable parameters. The input signal that writes to the memory is updated as shown in (3), where e_x, e_h and e_m are encoding vectors:

$$u_t = e_x^T x_t + e_h^T h_{t-1} + e_m^T m_{t-1} \quad (3)$$

The memory dynamics is discretized for computation through methods such as the Euler's method, and its update in time is described by (4):

$$\hat{m}_t = A\hat{m}_{t-1} + B\hat{u}_t \quad (4)$$

Increasing the memory dimensions d improves the LMU ability to store high-frequency input signals and efficiently handle complex temporal dependencies.

2.2 Leaky Integrate-and-Fire (LIF) neuron model

The fundamental units SNNs rely on are simplified models of biological neural dynamics (Gerstner and Kistler, 2002), and the Leaky Integrate-and-Fire (LIF) model is the one offering the most effective trade-off between biological plausibility and computational complexity (Izhikevich, 2006). It mimics the event-based biological neural communication through the evolution in time of the membrane potential, described by (5) as:

$$\tau_m \frac{dv}{dt} = -v(t) + Ri(t) \quad (5)$$

where τ_m is the decay time of the membrane potential $v(t)$, $i(t)$ is an input current and R models the electrical resistance. When the membrane potential overcomes a threshold V_{th}, a spike is emitted and $v(t)$ undergoes reset; the latter being realized by setting a new membrane potential value such that $v(t + dt) < V_{th}$ (Gerstner and Kistler, 2002).

2.3 LIF-based LMU (L^2MU)

The L^2MU design reconfigures the standard LMU by replacing each primary component (input representation, memory, and hidden state) with populations of LIF neurons, allowing for spike-based communication between the model's key components (Fra et al., 2024). Each LMU block is represented by a neuron population: u for the input, m for the memory state, and h for the hidden state. These populations use spiking activity to translate the traditional LMU equations into their event-based counterpart. Specifically, the input current $u_{t,\text{curr}}$ is calculated from a weighted sum of spiking activities from the input, memory, and previous hidden states, as defined by (6):

$$u_{t,\text{curr}} = \mathbf{e}_x^T \cdot x_{t,\text{spk}} + \mathbf{e}_y^T \cdot h_{t-1,\text{spk}} + \mathbf{e}_m^T \cdot m_{t-1,\text{spk}} \quad (6)$$

Similarly, the memory state $m_{t,\text{curr}}$ is updated based on the spiking activities of the previous memory and input states, as given in (7):

$$m_{t,\text{curr}} = A \cdot m_{t-1,\text{spk}} + B \cdot u_{t,\text{spk}} \quad (7)$$

where this update is contingent on spiking thresholds $m_{t,\text{mem}} > \Theta_m$ and $u_{t,\text{mem}} > \Theta_u$, ensuring that only significant spikes contribute to the memory update. The hidden state $h_{t,\text{curr}}$ is subsequently computed from the combined spiking activity of the input, previous hidden, and memory states, as shown in (8):

$$h_{t,\text{curr}} = \mathbf{W}_x \cdot x_{t,\text{spk}} + \mathbf{W}_h \cdot h_{t-1,\text{spk}} + \mathbf{W}_m \cdot m_{t,\text{spk}} \quad (8)$$

and its spiking is controlled by a threshold condition $h_{t,\text{mem}} > \Theta_h$.

As final step, the spikes from the hidden layer are passed through a fully connected layer of LIF neurons which represents the output of the model.

2.4 Extended Kalman Filter

The Extended Kalman Filter (EKF) (Ribeiro, 2004) is an extension of the classic Kalman Filter (KF), designed to estimate the state of a non-linear dynamic system from noisy measurements, as commonly encountered in robotics and navigation. Unlike the standard KF, which assumes linearity, the EKF handles non-linearities by linearizing the state transition and observation models using Jacobian matrices computed around the current estimate.
The EKF operates in two main steps: prediction and update. During the prediction step, the system's next state is estimated using a non-linear function that depends on the current state and control input, incorporating process noise. This function is locally linearized using its Jacobian to approximate the system's evolution.
In the update step, the predicted state is mapped to the measurement space through a non-linear observation model, which is also linearized via its Jacobian. The difference between the actual and predicted measurements (innovation) is used to correct the estimate, with the Kalman Gain weighting the correction based on the respective uncertainties.
By adapting to non-linear system dynamics through local linearizations, the EKF enables accurate and efficient real-time state estimation, making it a fundamental tool in sensor fusion and control applications.

3. METHODOLOGY

The neuromorphic redesign of the LMU can be effectively adopted in encoding-free scenarios in which raw data are directly fed to SNN-based models (Fra et al., 2024). Additionally, by considering the membrane potential of the LIF neurons at the output layer as presented in the following, rather than their spiking activity, the L^2MU can be employed for regression tasks.

The hyperparameters of the model, such as decay rates and threshold values for each population of LIF neuron, as well as the dimensions of the second and third layers of the encoding module, are optimized using the Neural Network Intelligence (NNI) framework [1] (Fra et al., 2022; Fra, 2025).

The L^2MU is adopted for two standalone use cases in this work: the first on integrates the L^2MU with an EKF for position and velocity estimation, while the second on evaluates the performance of the L^2MU model operating independently of the EKF, focusing solely on position estimation.

[1] github.com/microsoft/nni

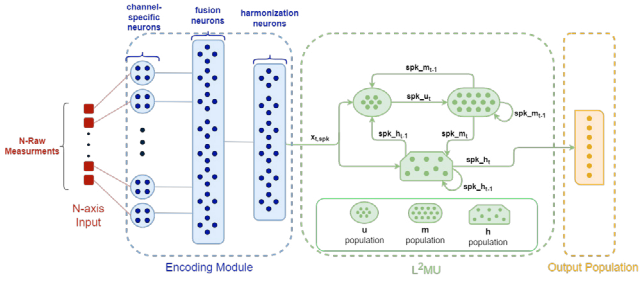


Fig. 1. Schematic of the model. The input consists of N measurements, varying with sensor configurations, fed into a 3-layer encoding module. The module transmits spikes to the L2MU, the output of the model is potential membrane of output population, computed from the hidden state (h).

3.1 Encoding module

As shown in Fig. 1, an encoding module can be adopted as an interface between the L^{2}MU and the sensory input, responsible of converting raw signals into spike trains. A fully connected architecture is considered, made of three layers of LIF neurons.The first layer generates spike patterns through neuron populations individually assigned to each input channel, increasing data dimensionality of channel-specific features. These patterns then propagate to the second layer of the encoding module, whose LIF neurons fuse information across channels creating an integrated representation. Finally, in the third layer, another population of LIF neurons harmonizes the information into a unified spiking pattern that captures a refined multidimensional encoding of the input.

3.2 Integration of L^{2}MU for EKF state prediction

The first use case is inspired by the Neural-Kalman framework proposed in (Du et al., 2023), which integrates Global Navigation Satellite System (GNSS) and IMU sensor data to enhance position estimation for robotic systems. Our approach combines the real-time processing capabilities of the biologically-inspired L^{2}MU model with the long-term accuracy of GPS, replacing the Temporal Convolutional Network (TCN) in the original model with L^{2}MU. In this scenario, the L^{2}MU model does not rely on an encoding module, and instead directly processes raw IMU data as input to the u and h populations.

An EKF is employed to estimate the state of a moving object—comprising its position and velocity—by fusing the velocity predictions generated by the L^{2}MU with noisy GPS measurements. The L^{2}MU model plays a key role in predicting the velocity of the object, which is then used to estimate its future state. Specifically, the state prediction is based on a simple kinematic model that updates the current position using the predicted velocity, as defined in (9):

$$\hat{x}(t+1) = \begin{bmatrix} x(t) + v_x(t)\Delta t \\ y(t) + v_y(t)\Delta t \\ v_x(t) \\ v_y(t) \end{bmatrix} \tag{9}$$

where $x(t)$, $y(t)$ represent the position of the object at time t, and $v_x(t)$, $v_y(t)$ are the predicted velocities along the x- and y-axes, respectively. Δt is the time interval between consecutive steps. Once the future state is predicted, the EKF performs a correction step using GPS measurements, which may be affected by noise and biases. The Kalman gain refines the final estimate by minimizing the residual between the predicted state and the observed data.
For this use case, we employed the Agrobot dataset (Du et al., 2023). In this dataset, the robot, equipped with a Sparkfun Razor IMU sensor, operated indoors over a distance of approximately 2.5 km during a 3-hour session. IMU data was sampled at 100 Hz, and GPS data was synthetically generated. Ground-truth positioning was obtained with an accuracy of ±5 cm using a video toolbox that identified the center of the robot's bounding box. The dataset thus provides controlled and ideal conditions for testing state prediction performance.

3.3 L^{2}MU standalone position prediction

For the second use case, the L^{2}MU is used to predict the displacement of a moving object from sensor measurements alone and to enhance the accuracy of GPS-based positioning by correcting errors over time. For this use case L^{2}MU model utilizes the encoding module that transforms raw sensor data into spike patterns, which are processed to capture temporal details and enable accurate position estimation.
The model training utilized different multi-sensor configurations, such as IMU combined with Wheel Encoder (WE), to assess the impact of sensor fusion on performance.
Each configuration was applied to one of the model's two primary tasks. In the first task, the model aimed to improve GPS-based position estimates. Starting with the GPS position at $t-1$, the model corrected minor inaccuracies in the GPS data by leveraging additional sensor inputs. This configuration is valuable when GPS data is available but may contain minor errors.
The second task involved dead reckoning, where the model relied on sensor data alone to predict displacement without GPS. Starting from Ground Truth (GT) position, the model used the temporal sensor data to calculate cumulative displacement, allowing for accurate position tracking in scenarios where GPS may be unavailable or unreliable.
For both tasks, the training data structure included the initial position is either GPS or GT at $t-1$, depending on the task. The GPS task uses the GPS initial position, while the dead reckoning task uses GT. Multi-sensor measurements consist of a 1-second sequence sampled at 50 Hz, providing 50 measurements between $t-1$ and t. The final position is the GT position at time t, which serves as the prediction target.
During training, the GT is used as the loss function to optimize performance across both tasks. At the inference stage, as can be seen in Fig. 2, the model employs the GPS-refinement configuration when GPS data is available and reliable, using the GPS coordinates as the initial position. Conversely, in the absence or unreliability of GPS signals, the dead-reckoning configuration is applied, using the last predicted position as the initial position. The reliability of the GPS signal is determined based on the operating environment: GPS is considered unreliable in indoor settings, and reliable in outdoor scenarios.

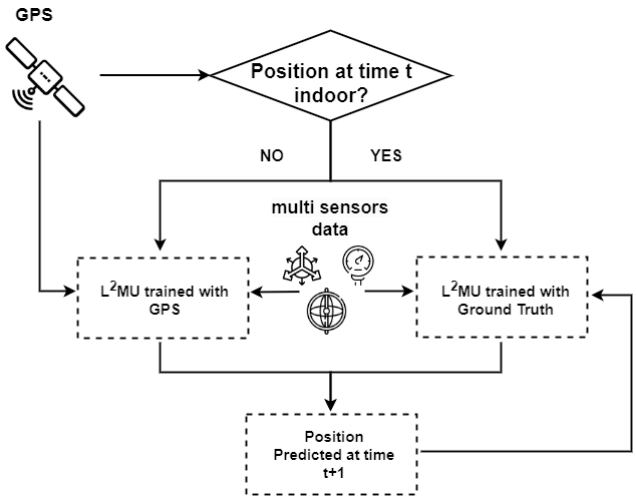


Fig. 2. The figure illustrates the model architecture used for the NCLT Vision and Lidar dataset testing phase.

For the second use case, the North Campus Long-Term (NCLT) dataset (Carlevaris-Bianco et al., 2016) was employed to train and evaluate the L^{2}MU model, which is tasked with predicting the position without relying on an EKF. This dataset, recorded on the University of Michigan's North Campus, offers diverse real-world sensor data captured during 90-minute sessions. Using a Segway robot equipped with IMUs, GPS, cameras, Light Detection And Ranging (LiDAR), and WE, it encompasses various scenarios, including urban streets and indoor spaces, under different weather and lighting conditions. GT positioning was derived through a combination of LiDAR-based SLAM and RTK GPS, ensuring high accuracy positioning even in challenging environments. These features make the NCLT dataset well-suited for evaluating model performance in realistic environments where GPS signals may be unreliable or interrupted.

4. RESULTS AND DISCUSSIONS

The results are examined within two distinct yet closely related use cases, aligned with the models, each of them trained and tested on a different dataset.

4.1 Use Case I: integration of L^{2}MU for EKF state prediction

In the first use case, we replaced the TCN in the Neural-Kalman model with a neuromorphic approach using L^{2}MU, optimizing real-time position and speed prediction in agricultural environments. Neural-Kalman was originally designed to process accelerometer, magnetometer, and gyroscope data using a TCN, which predicts the state inputs for an EKF. The EKF then integrates these state predictions with measurements derived from GPS data to estimate position and velocity. The capacity of L^{2}MU to process IMU data efficiently, omitting the encoding module, results in a reduction in memory usage without a corresponding loss of accuracy. Furthermore, its neuromorphic design suggests the potential for enhanced energy efficiency, making it a promising choice for real-time applications with limited memory and computational resources.

Looking at the comparison in Table 1, we can further appreciate the advantages of the L^{2}MU model. With an Absolute Trajectory Error (ATE) of 1.14 m, a metric detailed in (Du et al., 2023). The comparison of L^{2}MU with other models, such as Neural-KF (Agrobot model), EKF (Qi and Moore, 2002), and Unscented Kalman Filter (UKF) (Brossard et al., 2018), reveals that L^{2}MU demonstrates superior accuracy and memory efficiency. The model achieves a reduction of ATE compared to the results obtained by the EKF (2.22 m) and UKF (4.06 m) models. Furthermore, the model has a significantly smaller size than these other models. While it does not surpass the performance of Neural-KF, it offers comparable accuracy with the declared results (1.07 m), but outperforms it in terms of accuracy with the results obtained by re-training the model under the same configurations (2.45 m).

Table 1. Comparison of tracking methods

Method	ATE (m)	Model Size (MB)
UKF-MINS+GPS	4.06	8.09
EKF INS+GPS	2.22	5.48
Neural-KF (Du et al., 2023)	1.07 (2.45)	2.17
L^{2}MU *[Ours]*	1.14	0.05

4.2 Use Case II: L^{2}MU standalone position prediction

In the second use case, the model is evaluated on a version trained with the NCLT dataset, using a L^{2}MU model that operates without an EKF. Training employed Mean Squared Error (MSE) loss in two variants: one for GPS refinement—predicting the final position from an initial GPS reading—and one for distance estimation—starting from the ground truth. A combined model leveraging both approaches is analyzed on the full dataset, as illustrated in Fig. 2. In this case, "full dataset" refers to the inclusion of data sequences that were previously discarded due to synchronization issues between sensor, as described in Section 3.3.

These tasks were tested under different sensor configurations, including IMU alone, IMU with WE, and IMU with both WE and Optical Gyroscope (OG), over 1-second windows. The dataset was preprocessed to ensure proper temporal alignment, discarding samples with excessive desynchronization (>10 ms), unrealistic GPS shifts (>20 m), or temporal mismatches between GPS and sensor data.

In the GPS refinement task, the baseline GPS Root Mean Square Error (RMSE) ranged from 5.2 m to 6.2 m. The use of IMU alone slightly reduced the error. Adding the WE brought the RMSE down to approximately 5.0–5.5 m, while the inclusion of the OG had minimal further impact. This indicates that while the model can compensate for some GPS noise, issues such as signal loss or multipath effects are more difficult to mitigate.

In the distance estimation task, starting from the ground truth position, the IMU-only model exhibited the highest errors (RMSE between 0.40 m and 0.51 m). Adding the WE significantly improved accuracy, reducing the RMSE to 0.11–0.18 m. Including the OG offered only slight variations, showing limited additional benefit.

Subsequently, a testing phase was conducted in which the model operated at a frequency of 1 Hz, processing all

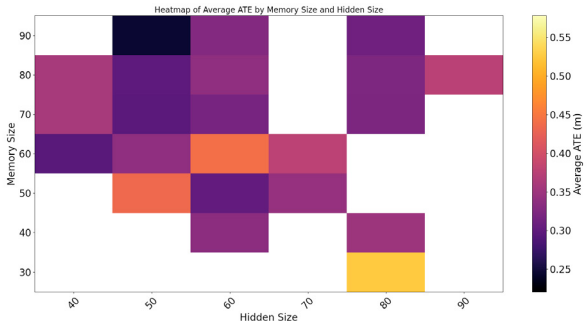


Fig. 3. Heatmap of L^{2}MU performance as a function of hidden size and memory size. Performance is measured using the ATE, averaged over different settings of the remaining hyperparameters. Results refer to the distance estimation task using the configuration with WE data.

data—including those with synchronization issues. The initial position is obtained from GPS when the signal is reliable and the object is outside a building; otherwise, it falls back to the last predicted position. Switching between the GPS-trained and non-GPS-trained models occurs depending on whether the object is indoors or outdoors.

The results using the three sensor configurations are reported in Table 2. Notably, the GPS exhibited significantly higher RMSE in the 2013-02-23 and 2013-04-05 test sets, indicating degraded performance in challenging environments. In contrast, the L^{2}MU model, particularly when using only IMU data, showed greater robustness, with lower and more stable error values. Interestingly, configurations with additional sensors (WE, OG) performed worse in this setup due to temporal misalignments, reaffirming challenges in multi-sensor fusion discussed in (Pignata et al., 2024).

Table 2. RMSE comparison between GPS and L^{2}MU models across different test sets in NCLT dataset (Chen et al., 2018).

Configuration	2013-01-10	2013-02-23	2013-04-05
RMSE GPS	13.04 m	435.10 m	636.03 m
RMSE L^{2}MU (IMU)	6.13 m	10.56 m	21.09 m
RMSE L^{2}MU (IMU+WE)	22.23 m	53.83 m	77.96 m
RMSE L^{2}MU (IMU+WE+OG)	13.30 m	38.50 m	43.24 m

A complementary analysis of model performance is shown in Fig. 3, where a heatmap illustrates how the ATE varies with hidden and memory size in the distance estimation task. Higher memory sizes generally yield better results, while the impact of hidden size is less consistent. Only a subset of hyperparameters is explored, given the large size of the overall search space.

4.3 NeuroBench metrics

This section presents the evaluation results obtained using NeuroBench (Yik et al., 2025), a specialized benchmarking framework for assessing computational metrics in the neuromorphic domain. The evaluation focuses on five key metrics: Footprint, Activation Sparsity, Synaptic Operations, Learnable Parameters, and Membrane Updates. These metrics offer a comprehensive overview of the performance, efficiency, and resource utilization of the examined neuromorphic systems and algorithms. The metrics are described as follows:

- Footprint: it measures the memory required to represent a model, expressed in Megabytes.
- Activation Sparsity: it represents the relative amount of inactive neurons during the execution of the model. A value of 0 denotes no sparsity, with all the neurons active at all times. Conversely, a value of 1 indicates maximum sparsity, with all neurons being inactive.
- Synaptic Operations: it quantifies the total computational workload during model execution, encompassing neuron activations and their synaptic connections. It includes three subcategories: Dense Operations, which reflects the load on hardware without sparsity support; effective Multiply-Accumulate Operationss (MACs), which measure operations with nonzero activations and connections on sparsity-aware hardware; and effective Accumulate Operationss (ACs), analogous to effective MACs but specific to binary activations involving additions rather than multiplications.
- Membrane Updates: it corresponds to the number of times the membrane potential of neurons changes during SNN execution.

The results for these metrics are presented in Table 3 and Table 4, which provide a detailed comparison of the models in terms of memory usage, neuron activation behavior, computational workload, and spiking activity. These tables highlight the performance of the evaluated models across various metrics, providing insights into their computational characteristics and resource requirements.

Table 3. Evaluation of different configurations for displacement prediction based on key performance metrics, calculated using NeuroBench.

Configuration	IMU	IMU + WE	IMU + WE + OG
Footprint (MB)	0.41	0.27	0.25
Activation Sparsity	0.70	0.82	0.81
Membrane Updates	42.46×10^3	37.26×10^3	33.26×10^3
Effective MACs	18.00×10^3	21.97×10^3	17.97×10^3
Effective ACs	433×10^3	393.03×10^3	306.72×10^3
Dense Operations	2.50×10^6	1.86×10^6	1.56×10^6
Learnable Parameter	102×10^3	69×10^3	46×10^3

Table 4. Evaluation of different configurations for GPS refinement based on key performance metrics, calculated using NeuroBench.

Configuration	IMU	IMU + WE	IMU + WE + OG
Footprint (MB)	0.12	0.24	0.25
Activation Sparsity	0.79	0.74	0.84
Membrane Updates	18.32×10^3	32.36×10^3	37.09×10^3
Effective MACs	4.5×10^3	16.48×10^3	17.97×10^3
Effective ACs	100×10^3	328.93×10^3	331.28×10^3
Dense Operations	0.48×10^6	1.56×10^6	1.58×10^6
Learnable Parameter	29×10^3	59×10^3	98×10^3

5. CONCLUSION

In this work, we have reported on the potential of the L^{2}MU as a spiking alternative to traditional DL architectures for dead reckoning and GPS refinement. Employed

for two different use cases, the L^{2}MU demonstrated competitive accuracy and significant reductions in memory requirements, making it suitable for real-world applications with limited computational resources. This positions the L^{2}MU as a solution for scenarios with strict hardware constraints. Future research will focus on deploying L^{2}MU on neuromorphic hardware using the Neuromorphic Intermediate Representation (NIR) tool (Pedersen et al., 2024) to fully exploit the low power consumption and event-driven nature of SNN. Without this hardware level deployment, such advantages are significantly reduced or lost, representing a key limitation of the approach. By enabling hardware-level implementation, L^{2}MU can fully leverage these strengths, expanding its applicability in domains like AI and Internet of Things (IoT), where resource-efficient and low-latency solutions are critical. As the demand for smarter, more efficient computing grows, models like L^{2}MU offer promising alternatives to conventional architectures, driving the next generation of intelligent systems optimized for resource-constrained environments.

ACKNOWLEDGMENT

This Research is funded by the European Union - NextGenerationEU Project 3A-ITALY MICS (PE0000004, CUP E13C22001900001, Spoke 6) and the Fluently project with Grant Agreement No. 101058680. We acknowledge a contribution from the Italian National Recovery and Resilience Plan (NRRP), M4C2, funded by the European Union – NextGenerationEU (Project IR0000011, CUP B51E22000150006, "EBRAINS-Italy").

REFERENCES

Akai, N., Morales, L.Y., Yamaguchi, T., Takeuchi, E., Yoshihara, Y., Okuda, H., Suzuki, T., and Ninomiya, Y. (2017). Autonomous driving based on accurate localization using multilayer lidar and dead reckoning. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 1–6. IEEE.

Brossard, M., Bonnabel, S., and Barrau, A. (2018). Unscented kalman filter on lie groups for visual inertial odometry. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 649–655. IEEE.

Carlevaris-Bianco, N., Ushani, A.K., and Eustice, R.M. (2016). University of michigan north campus long-term vision and lidar dataset. *The International Journal of Robotics Research*, 35(9), 1023–1035.

Chen, C., Lu, X., Markham, A., and Trigoni, N. (2018). Ionet: Learning to cure the curse of drift in inertial odometry. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Du, Y., Saha, S.S., Sandha, S.S., Lovekin, A., Wu, J., Siddharth, S., Chowdhary, M., Jawed, M.K., and Srivastava, M. (2023). Neural-kalman gnss/ins navigation for precision agriculture. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 9622–9629. IEEE.

Eshraghian, J.K., Ward, M., Neftci, E., Wang, X., Lenz, G., Dwivedi, G., Bennamoun, M., Jeong, D.S., and Lu, W.D. (2023). Training spiking neural networks using lessons from deep learning. *Proceedings of the IEEE*, 111(9), 1016–1054.

Fra, V. (2025). Application-oriented automatic hyperparameter optimization for spiking neural network prototyping. doi:10.48550/arXiv.2502.12172. ArXiv:2502.12172 [cs].

Fra, V., Forno, E., Pignari, R., Stewart, T.C., Macii, E., and Urgese, G. (2022). Human activity recognition: suitability of a neuromorphic approach for on-edge AIoT applications. *Neuromorphic Computing and Engineering*, 2(1), 014006. doi:10.1088/2634-4386/ac4c38. Publisher: IOP Publishing.

Fra, V., Leto, B., Pignata, A., Macii, E., and Urgese, G. (2024). Natively Neuromorphic LMU Architecture for Encoding-Free SNN-Based HAR on Commercial Edge Devices. In M. Wand, K. Malinovská, J. Schmidhuber, and I.V. Tetko (eds.), *Artificial Neural Networks and Machine Learning – ICANN 2024*, volume 15025, 377–391. Springer Nature Switzerland, Cham. doi:10.1007/978-3-031-72359-9_28. Series Title: Lecture Notes in Computer Science.

Georgiadou, P. and Kleusberg, A. (1988). On carrier signal multipath effects in relative gps positioning. *Manuscripta geodaetica*, 13(3), 172–179.

Gerstner, W. and Kistler, W.M. (2002). *Spiking neuron models: single neurons, populations, plasticity*. Cambridge University Press, Cambridge, U.K. ; New York.

Izhikevich, E.M. (2006). *Dynamical Systems in Neuroscience*. The MIT Press. doi:10.7551/mitpress/2526.001.0001.

Maass, W. (1997). Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, 10(9), 1659–1671. doi:10.1016/S0893-6080(97)00011-7.

Pedersen, J.E., Abreu, S., Jobst, M., Lenz, G., Fra, V., et al. (2024). Neuromorphic intermediate representation: a unified instruction set for interoperable brain-inspired computing. *Nature Communications*, 15(1), 8122. doi:10.1038/s41467-024-52259-9.

Peper, F. (2017). The end of moore's law: opportunities for natural computing? *New Generation Computing*, 35(3), 253–269.

Pignata, A., Fra, V., Macii, E., and Urgese, G. (2024). A time-synchronized framework for bluetooth low energy wireless sensor networks. In *2024 IEEE 18th International Conference on Application of Information and Communication Technologies (AICT)*, 1–6. doi:10.1109/AICT61888.2024.10740423.

Qi, H. and Moore, J.B. (2002). Direct kalman filtering approach for gps/ins integration. *IEEE Transactions on Aerospace and Electronic Systems*, 38(2), 687–693.

Ribeiro, M.I. (2004). Kalman and extended kalman filters: Concept, derivation and properties. *Institute for Systems and Robotics*, 43(46), 3736–3741.

Voelker, A., Kajić, I., and Eliasmith, C. (2019). Legendre memory units: Continuous-time representation in recurrent neural networks. *Advances in neural information processing systems*, 32.

Yik, J. et al. (2025). The neurobench framework for benchmarking neuromorphic computing algorithms and systems. *Nature Communications*, 16(1), 1545.