Research Article

Improving the Accuracy of Location Data in UWB-Based RTLS Using Deep Learning Methods

Ramazan KAVAK[1,4], Fatih AYDEMİR[2], Serap CEKLİ[3]

[1] Graduate Education Institute, Istanbul University - Cerrahpasa, İstanbul, Orcid ID: https://orcid.org/0000-0002-5700-8439, E-mail: ramazankavak@ogr.iuc.edu.tr,
[2] Asis Automation and Fueling System Inc.,Istanbul ,Orcid ID: https://orcid.org/0009-0006-2735-2606, E-mail: f.aydemir@asis.com.tr,
[3] Electrical and Electronics Engineering, Istanbul University – Cerrahpasa, İstanbul, Orcid ID: https://orcid.org/ 0000-0002-8113-0514, E-mail: serap.cekli@iuc.edu.tr,
[4] Asis Automation and Fueling System Inc., E-mail: Istanbul., r.kavak@asis.com.tr

* Correspondence: r.kavak@asis.com.tr

Received: 17 May 2025
Revised: 12 June 2025
2nd Revised: 15 November 2025
Accepted: 26 November 2025
Published: 28 November 2025

Reference: Kavak, R., Aydemir, F., & Cekli, S. (2025). Improving the accuracy of location data in UWB-based RTLS using deep learning methods. *The European Journal of Research and Development, 5*(1), 246–262.

Abstract

In Real-Time Location Systems (RTLS) using Ultra-Wideband (UWB) technology, the Decawave DW1000 uses the Two-Way Ranging (TWR) method to obtain the location of a moving object. Multipath propagation occurring under NLOS conditions systematically negatively affects time-leads and distance measurements; this increases the bias (positive bias) and widens the variance, leading to instability in the location data. In this study, an autoencoder-based measurement improvement method proposed for the tag location data obtained using the TWR method. The raw TOF (time of flight) and range measurements obtained from the DW1000 are simultaneously integrated into a low-dimensional latent space with features such as RSSI and CIR-based quality metrics (e.g., first-path amplitude/index, channel energy, pulse width indicators). The denoising/regularized reconstruction process suppresses the jump and bias components in the location data caused by NLOS; thus, the improved measurements can increase the stability and repeatability of location data when used with classical Gauss-Newton location. This approach can trained with a highly dynamic setup (especially using clean LOS records), reducing the burden of relying on field geometry; its modular architecture allows for minimal integration into the existing TWR software chain. Experimental analysis and visualizations were performed on different indoor *scenarios (office, corridor, and semi-open space layouts) using MATLAB. This method has been shown to provide a consistent reduction in mean error metrics (MAE/RMSE), a significant improvement in axis bias errors (95th/97th percentile), and location path continuity, while also eliminating erroneous outliers originating from instantaneous NLOS.*

Keywords:UWB, RTLS, DW1000, TWR, NLOS, multipath, Autoencoder, LSTM, deep learning.

1. Introduction

The solution created with RTLS used in critical areas such as automation, warehouse management, and employee tracking. This enables positioning in enclosed spaces. UWB has emerged as an alternative to GPS/GNSS in indoor environments and offers high accuracy for indoor use. However, this results in increased positioning errors in indoor systems, referred to as NLOS (No Line of Sight). Therefore, with data-focused solutions and the improvements made, more stable results are obtained. Although error values are reduced with classical filtering, it may not perform the same due to the variability of NLOS. The aim of this study is to address the issue with a different approach using a CNN (Convolutional Neural Network)-based Autoencoder model. The developed model learns the normalized systematic pattern in the data and learns how to organize data in variable NLOS conditions. CNN does not require a label structure for the data. The system enters the learning process directly with the measured data. As a result, this approach aims to reduce NLOS effects in UWB-based positioning systems, enabling more reliable, stable, and real-time position estimation.

Recent research on Ultra-Wideband (UWB)-based Real-Time Location Systems (RTLS) has focused on mitigating range and position errors caused by multipath and Non-Line-of-Sight (NLOS) propagation through data-driven methods. Coene et al. [1] proposed a location-aware range-error correction (LARC) framework that improves the accuracy of Decawave DW1000 measurements by combining geometric and contextual information. Their regression-based approach demonstrated a significant reduction in mean ranging error, highlighting the potential of lightweight machine learning layers that can inserted before classical solvers.

Similarly, Pei et al. [2] introduced an attention-enhanced fully convolutional network (FCN-Attention) to classify LOS and NLOS signals using channel impulse response (CIR) features. The self-attention mechanism increased the robustness of classification, proving that deep learning models can capture temporal and spectral dependencies in UWB signal distortions.

Niu et al. [3] further extended this idea by proposing a deep learning-based CIR feature regression model that directly estimates and compensates the ranging bias. Their results confirmed that CIR-driven representations outperform handcrafted thresholds or statistical filters, particularly under severe multipath conditions.

Complementary to these static models, Poulose and Han [4] utilized a Long Short-Term Memory (LSTM) architecture for time-series-based UWB positioning, addressing

sequential NLOS effects and trajectory drift. This temporal modeling concept aligns closely with the hybrid Autoencoder-LSTM structure proposed in the present study, which aims to combine measurement denoising with dynamic correction.
Finally, Tran et al. [5] presented an experimental CIR dataset and analysis framework derived from DW1000/DWM3000 Two-Way Ranging (TWR) exchanges, revealing how CIR-based latent patterns correspond to NLOS-induced biases. Their findings underline the importance of feature selection and unsupervised learning for measurement enhancement layers similar to the one designed in this work.
Section 2 details the DW1000-TWR model, NLOS effects on bias/variance, and feature extraction (first-path amplitude, channel energy, pulse width, SNR). Section 3 covers system topology, anchor–tag geometry, TWR protocol, and the LOS/NLOS/static data collection setup. Section 4 reports RMSE, CEP95, bias reduction, trajectory consistency, and runtime, comparing our CNN-AE layer against EKF and standalone AE baselines. Section 5 discusses generalizability and dataset dependence, and points to next steps: hybrid CNN–LSTM models, online recalibration, domain adaptation, and tightly coupled UWB–IMU fusion.

2. Materials and Methods

2.1. UWB

The main operating principle of UWB is based on the transmission of low-power, short-duration pulses across a wide band. This structure provides high time resolution in distance measurements based on ToA/TDOA. Due to its centimeter-level resolution, low energy consumption, and interference resistance, UWB has become a preferred choice for indoor positioning.

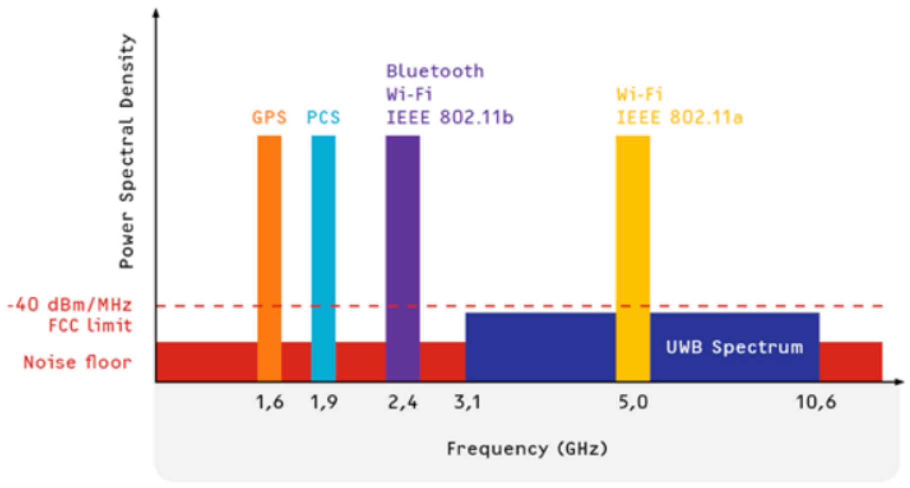


Figure 1: Spectrum of UWB and other tecnologies for RTLS [6].

2.2. Two Way Ranging (TWR)

Two-Way Ranging (TWR) is a positioning method that estimates distance by measuring the time of flight (ToF) between two wirelessly communicating devices [7]. This method is widely used, especially in UWB (Ultra-Wideband) systems. TWR is cost-effective and a more practical method than TDOA, as it works without requiring time synchronization. TWR is a two-way messaging method between a Tag (mobile device) and an Anchor (fixed reference point). The process generally consists of 3 stages [7]:
Poll (Request): Tag sends a message to the anchor.
Response: Anchor receives the message and sends a response after a certain processing time.
Final: Tag receives the response and calculates the distance based on all time measurements.

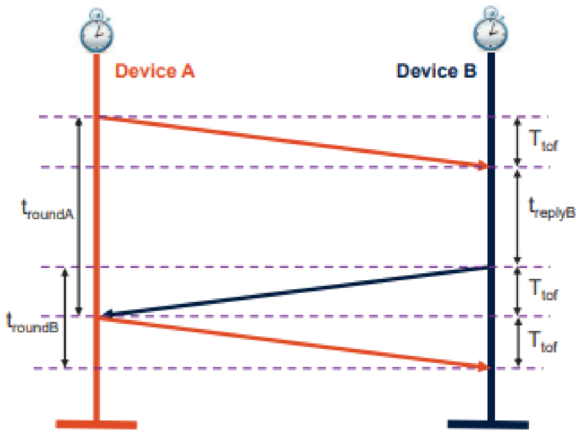


Figure 2: Times between 2 devices for TWR method [8].

$$T_{roundA} = 2 * T_{tof} + T_{replyB} \tag{1}$$
$$T_{roundB} = 2 * T_{tof} \tag{2}$$
$$T_{tof} = \frac{1}{4}\left(T_{roundA} + T_{roundB} - T_{replyB}\right) \tag{3}$$

Here [9];
T_{roundA} : Total time from the sending of the first message to the receipt of the Anchor response, measured by the Tag (mobile device).
T_{roundB} : The time from the reply sent to the final message received, measured by the Anchor (fixed device).
T_{replyB} : The processing time of the Anchor device from the time it receives the incoming message until it responds.
d : The calculated distance (in meters) :

$$d = c * T_{tof} \tag{4}$$

c : The speed of light in a space, approximately ≈ 3×10^8 m/s [9].

2.3. NLOS

NLOS refers to situations where the signal reaches the receiver indirectly via reflections/refractions, potentially obstructing communication within and around the wideband structure between the transmitter and receiver signals. As a result, the signal reaches the receiver from a longer distance than normal, causing the position data to be measured either too large or too small. Consequently, these are signal-distorting conditions that cause position data to drift incorrectly or systematically. Filtering/center-of-gravity methods have a certain degree of adaptability; deep learning offers solutions that are more flexible by learning from data with complex distributions.
This established system; In LOS conditions, the location adjustment of the Tag gives precise results on level of cm. However, in industrial or office environments, the placement of Anchors is possible. There are obstacles such as wall packages, columns, metal cabinets or raps between the Anchor and the Tag view. There may be structures between the Anchor and the Tag that may obstruct the view, as seen in Figure 3.

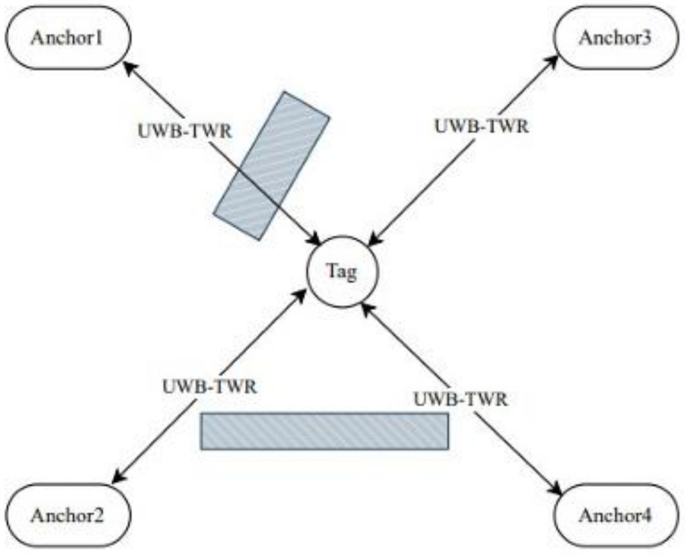


Figure 3: NLOS condition in UWB RTLS architecture.

2.4. Autoencoder

An autoencoder is a two-part (encoder–decoder) neural network that reconstructs the input data by transforming it into a low-dimensional latent representation (latent space). The encoder summarizes the data using the $x \to h$ transformation, and the decoder reproduces the input as close to the real one as possible using $h \to \hat{x}$. The undercomplete setup allows the model to learn discriminative patterns rather than "memorizing," and the reconstruction error is reduced, updating the weights for estimating the location data. This mechanism offers advantages over classical methods in terms of dimensionality reduction and noise suppression. Recent studies have confirmed that deep autoencoders can learn high-quality low-dimensional codes with effective weight initialization and layered pre-training [10].

The denoising autoencoder (DAE) type teaches the model "robust" features by adding artificial distortions to the input and demanding a smooth data output from the data network; this helps distinguish between random noise and systematic distortions in the signal. Theoretical interpretations of DAEs (e.g., correlation with score-matching) and their success in practice have demonstrated the usefulness of the reconstruction loss as a powerful feature extraction method with appropriate regularizations [11].

In the context of UWB-RTLS (DW1000, TWR), multipath propagation and signal reflections due to NLOS conditions make end-of-time (TOF) detection biased and scattered; measured distances are often positively biased. The autoencoder can be run in front of the location solver (Gauss-Newton) as a data-driven "measurement-enhancement layer" to correct these erroneous measurements. In a typical setup, the input vector contains features such as range/TOF obtained from the anchor, RSSI, first-path index/amplitude, CIR-based energy, and waveform indicators. Unsupervised training performed with "nominal" data collected under LOS; since the model learns the typical (LOS) distribution, it tends to suppress NLOS-induced anomalous patterns in the reproduction. In the loop, the AE output $\hat{x}$ (the improved measurement/feature) integrated with the location data, thus reducing both bias and narrowing the variance. It has been stated that CIR-based data-driven approaches provide significant improvements in NLOS conditions where LDE (leading-edge) bounds are insufficient [12].

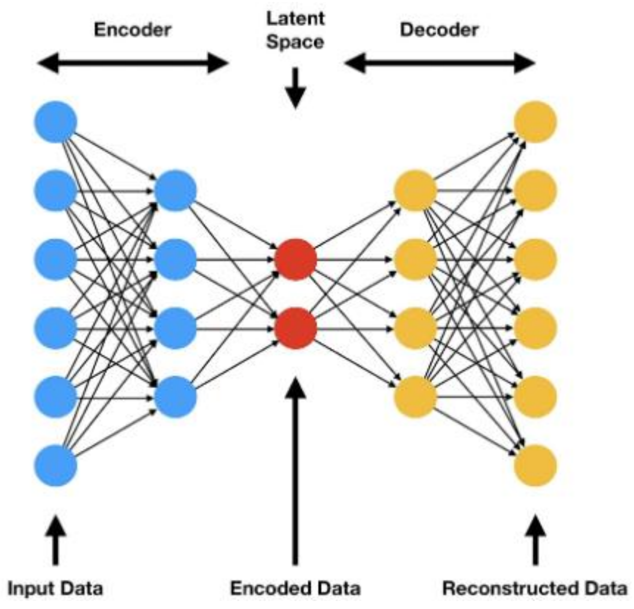


Figure 4: Data Processing Structure with Encoder–Decoder [12].

Recent studies using deep learning in UWB systems to address the deterioration of location data due to NLOS have shown that this approach is effective for both classification (LOS/NLOS) and direct error reduction tasks. For example, studies using transfer learning have improved generalization performance across different environments, while fully convolutional (FCN) networks with attention mechanisms have provided more accurate NLOS data. Such classifiers can be cascaded with AE (AE → classifier/regressor) to create a method for "cleaning the signal and/or estimating the measurement error" [13].

A method for this purpose follows the following steps: (1) data and feature design (TOF/distance, CIR statistics, quality flags); (2) pre-training standardization and/or windowing; (3) unsupervised training of a small and shallow AE (e.g., 2–3 hidden layers, 32–128 neurons) on LOS data; (4) inference in the field: $x \rightarrow \text{AE} \rightarrow \hat{x} \rightarrow$ location solver. The computational complexity is low; since the feedforward is single-pass, the latency can kept in the millisecond range. Real-time execution in RTLS applications is possible with small parameter budgets. This data-driven correction provides improvement in the measurement layer without making any changes to the classical estimation stage (GN/UKF). Moreover, "bias estimation and compensation" can done directly by adding a lightweight regressor after the latent code of the AE; this design, which learns the non-linear relationship between CIR and measurement error, has been shown to provide significant error reductions in the field [14].

As a result, the classical encoder–decoder autoencoder provides a low-latency, modular measurement-enhancement layer that can suppress NLOS-induced drift in UWB-RTLS in an unsupervised manner. Training based on LOS data reduces the cost of large-scale tags, and the AE output converted into "cleaned" measurements, improving the stability and accuracy of the location solver. Findings from application studies show that CIR/feature-based deep learning pipelines reduce systematic bias in TWR measurements and significantly reduce the total location error [15].

3. Installation and Measurement Results

Our system consists of three main devices: Anchor, Tag, and Gateway. At least three anchors placed at fixed points in the area. A tag used to locate a moving object or personnel. Tag and Anchor designed using an STM series microcontroller. Location data from the anchor sent to a Gateway device via RS485. The Gateway then sends this data to the server via a TCP/IP connection. The Gateway design was also implemented using a Raspberry Pi.

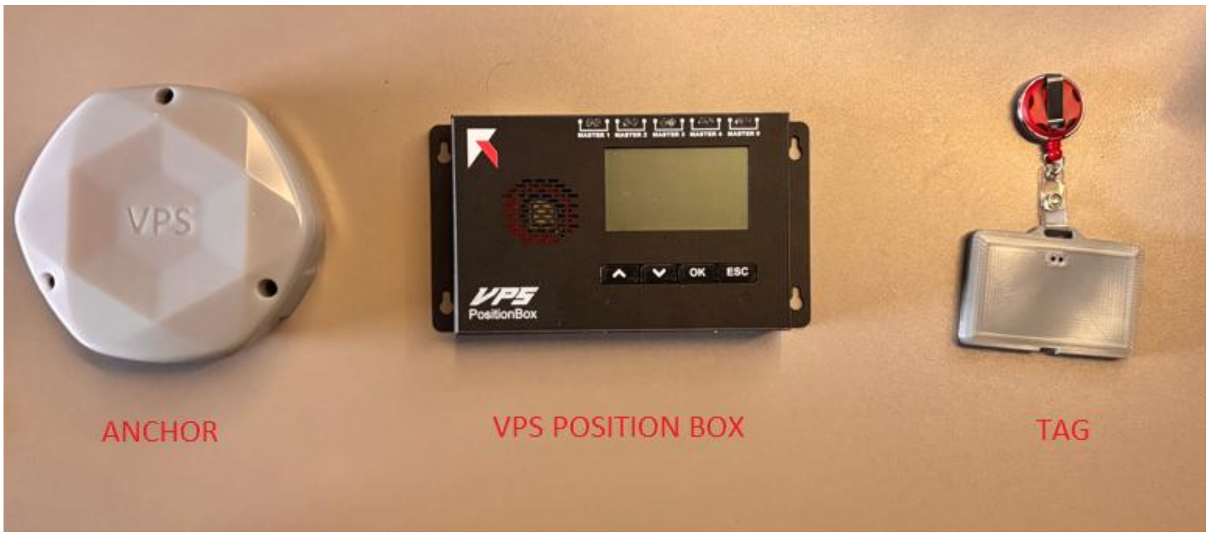


Figure 5: Devices designed in our RTLS system.

In the RTLS architecture, we designed; 4 Anchors are fixedly placed in the corners as a 4-diagonal layout in a 15x15 m^2 area. The communication between the Tag attached to the personnel and the Anchors made with UWB. The TWR method used in our system. After the Tag receives responses from the Anchors to the Poll and Final messages in order with the TWR method, the location data is obtained by taking the Trilateration method in 3 dimensions.

In our test environment, location data obtained with UWB-TWR is inaccurate and garbled due to NLOS-induced multipath conditions. The block diagram of the Autoencoder algorithm we propose to improve this performance loss in location data, along with the designed code and algorithm functions, provided below.

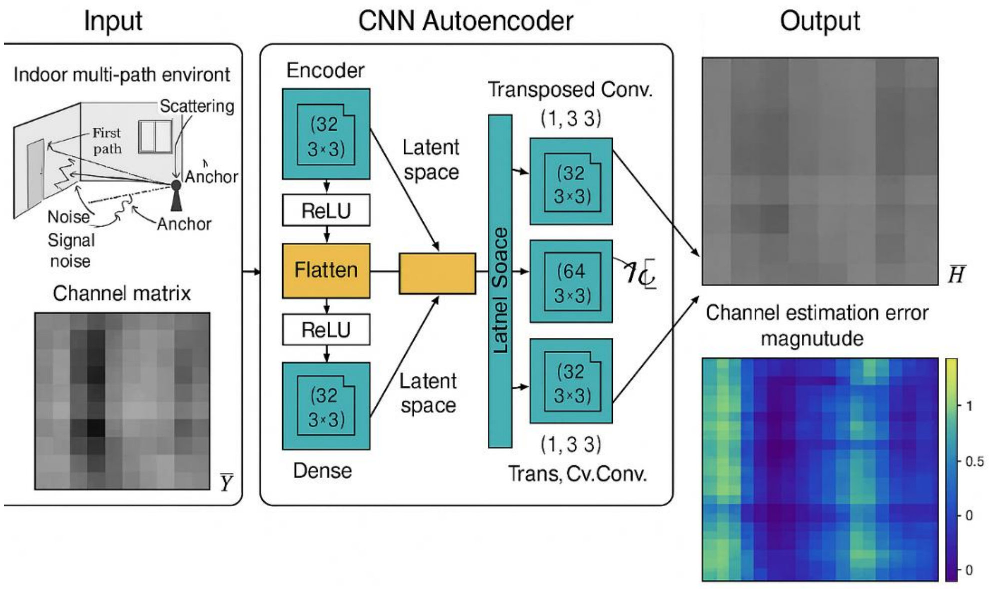


Figure 6: A Convolution Neural Network for CNN autoencoder for channel estimation visualization.

Algorithm1:UWB Localization With $\text{CNN}_{\text{Encoder}}$

```
Input: $D_{train}$={($D_k$, $y_k$)}{k=1.. $N_{train}$}$D_{valid}$={($d_v$, $y_v$)}{v=1.. $N_{valid}$}$D_{test}$={$D_t$}_{t=1.. $N_{test}$}
Hyperparameters: $E_{max}$, B, $\eta_0$, $\alpha$, $\beta$, patience, $Ir_{decay}$, $MIN_{IR}$, $\lambda_{reg}$, K
Output: $\hat{H}$={ ($\hat{x}_t$, $\hat{y}_t$) } for t in $D_{test}$ and learned $\theta_{enca}$, $\theta_{dec}$, $\theta_{pred}$
1:Initialize $\theta_{enca}$, $\theta_{dec}$, $\theta_{pred}$
2: $best_{\text{valid loss}} \leftarrow +\infty$
3: $No_{\text{improve epoch}} \leftarrow 0$
4:$\eta \leftarrow \eta_0$
5:for epoch=1 to $E_{Max}$do
6:   Shuffle $D_{train}$
7:   Partition $D_{train}$into minibatches size B
8:   for each minibatch ($B_{data}$, $B_{labels}$) in $D_{train}$do
9:        $z_{batch}$= $\text{CNN}_{\text{Encoder}}$ ($B_{data}$; $\theta_{enca}$)
10:       $\hat{y}_{\text{reconBatch}}$= $\text{CNN}_{\text{Decoder}}$ ($z_{batch}$; $\theta_{dec}$)
11:       ( $\hat{x}_{batch}$, $\hat{y}_{batch}$) = $\text{Pred}_{\text{Head}}$ ($z_{batch}$; $\theta_{pred}$)
12:       $L_{\text{Recon Batch}}$= $(1/|B_{data}|)\ \Sigma_i \| B_{data}[i] - \hat{y}_{\text{reconBatch}}[i] \|^2$
13:       $L_{\text{Pos Batch}}$= $(1/|B_{data}|)\ \Sigma_i \| B_{labels}[i] - (\hat{x}_{batch}[i], \hat{y}_{batch}[i]) \|^2$
14:       $L_{\text{Reg}}$= $\lambda_{\text{Reg}} (\|\theta_{enca}\|^2 + \|\theta_{dec}\|^2 + \|\theta_{pred}\|^2)$
15:       $L_{\text{Total}}$= $\alpha \cdot L_{\text{Recon Batch}} + \beta \cdot L_{\text{Pos Batch}} + L_{\text{Reg}}$
16:       ($\nabla \theta_{enca}$, $\nabla\theta_{dec}$, $\nabla\theta_{pred}$)=Backpropagate( $L_{\text{Total}}$)
17:       $\theta_{enca}$= $\text{optimizer}_{\text{update}}$ ($\theta_{enca}$, $\nabla\theta_{enca}$, $\eta$)
18:       $\theta_{dec}$= $\text{optimizer}_{\text{update}}$ ($\theta_{dec}$, $\nabla\theta_{dec}$, $\eta$)
19:       $\theta_{pred}$= $\text{optimizer}_{\text{update}}$ ($\theta_{pred}$, $\nabla\theta_{pred}$, $\eta$)
20:    end for
21:    $Valid_{loss}$= EvaluateValidLoss($D_{Valid}$, $\theta_{enca}$, $\theta_{dec}$, $\theta_{pred}$)
22:    if $Valid_{loss} < Best_{Valid\ Loss}$ then
23:       $Best_{Valid\ Loss}$= $Valid_{loss}$
24:       SaveModelCheckpoint($\theta_{enca}$, $\theta_{dec}$, $\theta_{pred}$)
25:       $No_{Improve\ epochs}$ = 0
26:    else
27:       $No_{Improve\ epochs}$ +=1
28:    end if
```

```
29:   if $No_{Improve\ epochs} \geq$ patience then
30:       break
31:   end if
32:   if epoch mod K = 0 then
33:       $\eta = \max(\eta \cdot Ir_{Decay}, Min_{Ir})$
34:   end if
35:end for
36: $\theta_{enca}, \theta_{dec}, \theta_{pred}$= LoadBestModel()
37:$\hat{H}$ = empty list
38:for each $d_t$ in $D_{Test}$ do
39:   $Z_t = CNN_{Encoder}\ (d_t; \theta_{enca})$
40:   $(\hat{x}_t, \hat{y}_t) = Pred_{Head}\ (Z_t; \theta_{pred})$
41:   Append $(\hat{x}_t, \hat{y}_t)$ to $\hat{H}$
42:end for
43:return $\hat{H}$
Subroutine EvaluateValidLoss($D_{Valid}, \theta_{enca}, \theta_{dec}, \theta_{pred}$) Inputs: $D_{Valid}$={($D_V, Y_V$)}, $\theta_{enca}, \theta_{dec}, \theta_{pred}$
Output: $validation_{loss}$
$A_{Sum}$= 0
$B_{Sum}$= 0
for each ($D_V, Y_V$) in $D_{Valid}$ do
    $Z_V = CNN_{Encoder}\ (D_V; \theta_{enca})$
    $\hat{y}_{recon\ v} = CNN_{Decoder}\ (z_V; \theta_{dec})$
    $(\hat{x}_v, \hat{y}_v) = Pred_{Head}\ (z_V; \theta_{pred})$
    $A_{Sum}$+= $D_V \|- \hat{y}_{Recon\ v} \|^2$
    $B_{Sum}$+= $\| Y_V - (\hat{x}_v, \hat{y}_v) \|^2$
end for
$L_{Recon\ Val} = A_{Sum} / | D_{Valid} |$
$L_{Pas\ Val} = B_{Sum} / | D_{Valid} |$
return $\alpha \cdot L_{Recon\ Val} + \beta \cdot L_{Pas\ Val}$
```

This algorithm generates position estimates ($\hat{x}, \hat{y}$) from UWB signals using a CNN-based autoencoder structure. During the training process, the model is optimized with both reconstruction ($\hat{y}_{recon}$) and position loss ($L_{Pos\ Val}$). The validation loss

(EvaluateValidLoss) monitored at the end of each epoch; the parameters with the lowest loss kept and early stopping is applied. The result is a highly accurate position estimation model that is robust to NLOS effects.

FunctionBlock1: Feature Extraction and Prediction (CNN Encoder / CNN Decoder / Pred Head)

```
Input : $B_{data}, \theta_{enca}, \theta_{dec}, \theta_{pred}$
Output : $z_{batch}, \hat{y}_{\text{reconBatch}}, (\hat{x}_{batch}, \hat{y}_{batch})$
1: $z_{batch} = CNN_{Encoder}\ (B_{data}, \theta_{enca})$
2: $\hat{y}_{\text{reconBatch}} = CNN_{Decoder}\ (z_{batch}; \theta_{dec})$
3:( $\hat{x}_{batch}, \hat{y}_{batch}) = Pred_{\ Head}\ (z_{batch}; \theta_{pred})$
```

Explanation: The Encoder extracts features from the input data, the Decoder produces reconstructions from these features, and the Prediction Head generates (x̂, ŷ) position predictions.

FunctionBlock2: Optimization (Backpropagate + optimizer_update)

```
input$L_{Total}$ , η, $\theta_{enca}, \theta_{dec}, \theta_{pred}$
Output: $\theta_{enc\ new}, \theta_{dec\ new}, \theta_{pred\ new}$
1:($\nabla\ \theta_{enca}, \nabla\theta_{dec}, \nabla\theta_{pred}$) = Backpropagate($L_{Total}$ )
2: $\theta_{enc\ new}$= $\text{optimizer}_{\text{update}}$ ($\theta_{enca}, \nabla\theta_{enca}$, η)
3: $\theta_{dec\ new}$= $\text{optimizer}_{\text{update}}$ ($\theta_{dec}, \nabla\theta_{dec}$, η)
4: $\theta_{pred\ new}$= $\text{optimizer}_{\text{update}}$ ($\theta_{pred}, \nabla\theta_{pred}$, η)
5:return $\theta_{enc\ new}, \theta_{dec\ new}, \theta_{pred\ new}$
```

This block defines the network's learning mechanism. Gradients calculated from the loss using backpropagation, and each parameter optimized using the learning rate η.

FunctionBlock3: EvaluateValidLoss ($D_{Valid}, \theta_{enca}, \theta_{dec}, \theta_{pred}$)

```
Inputs: $D_{Valid}$={($D_V$ , $Y_V$ )}, $\theta_{enca}, \theta_{dec}, \theta_{pred}$
Output: $Validation_{loss}$
1: $A_{sum}$ =0 s
2: $B_{sum}$ =0
3: for each ($D_V$ , $Y_V$ ) in $D_{Valid}$ do
```

```
4:    $Z_V = CNN_{Encoder}\ (D_V; \theta_{enca})$
5:    $\hat{y}_{recon\ v} = CNN_{Decoder}\ (z_v\ ; \theta_{dec})$
6:    $(\hat{x}_v, \hat{y}_v) = Pred_{\ Head}\ (z_v\ ; \theta_{pred})$
7:    $A_{sum} += \| D_V - \hat{y}_{Recon\ v} \|^2$
8:    $B_{sum} += \| Y_V - (\hat{x}_v, \hat{y}_v) \|^2$
9: end for
10 $L_{Recon\ val} = A_{sum} / |D_{Valid}|$
11: $L_{Pos\ val} = B_{sum} / |D_{Valid}|$
12: return $\alpha \cdot L_{Recon\ val} + \beta \cdot L_{Pos\ val}$
```

This function measures the validation performance of the model. For each example, the reconstruction error and position error calculated separately. Then, these two components weighted by the coefficients α and β to produce the total validation loss. This metric is used to determine decisions regarding early stopping and saving the best model (model checkpoint).

3.1. System Installation

In the system design, four anchors used to obtain location data in one environment. Location data obtained using a single tag. The resulting location data could sent to the server via the Gateway. Log records of the resulting location data kept, and graphical outputs of the results obtained using MATLAB. A test scenario and our proposed EKF and Gauss-Newton iterations were also included. Tests conducted on the obtained location data for the resulting situation. Specifically, under NLOS conditions, the Autoencoder algorithm applied to our system, and system outputs monitored.

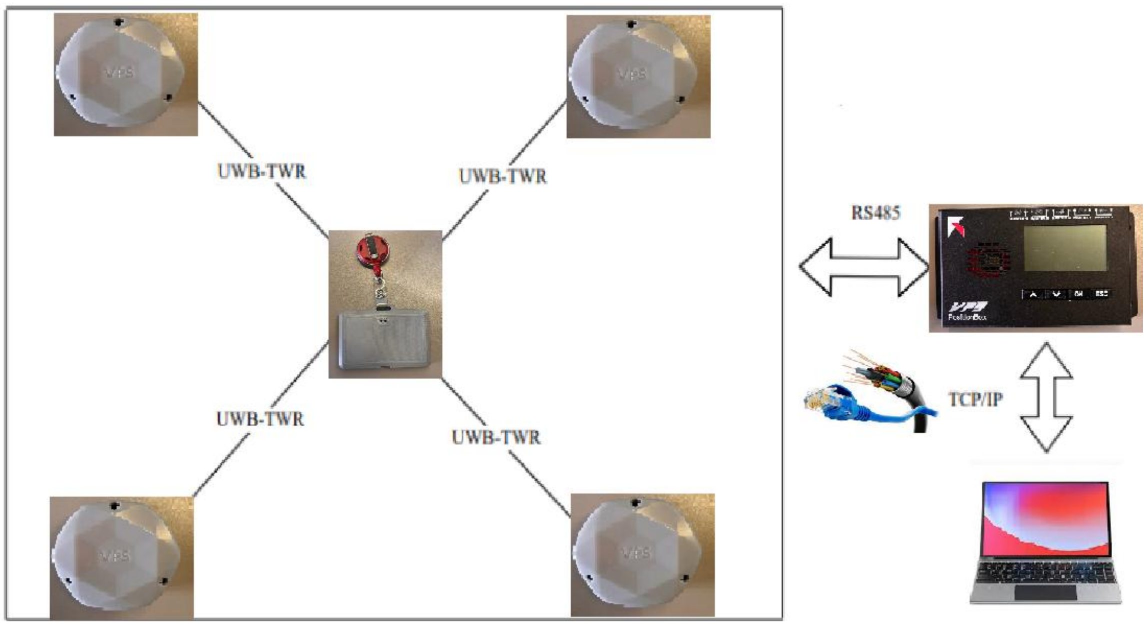
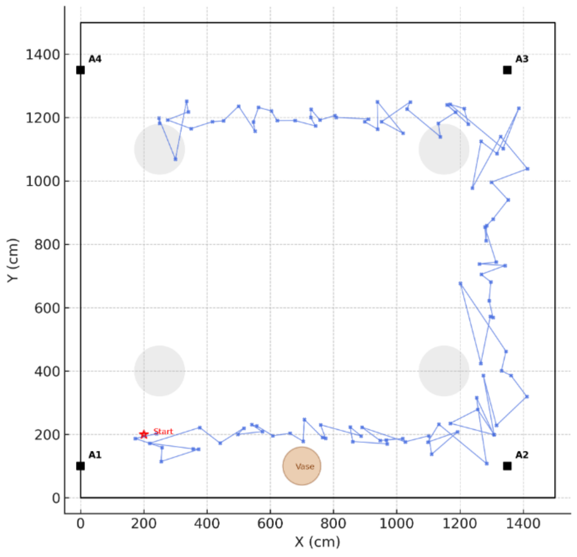


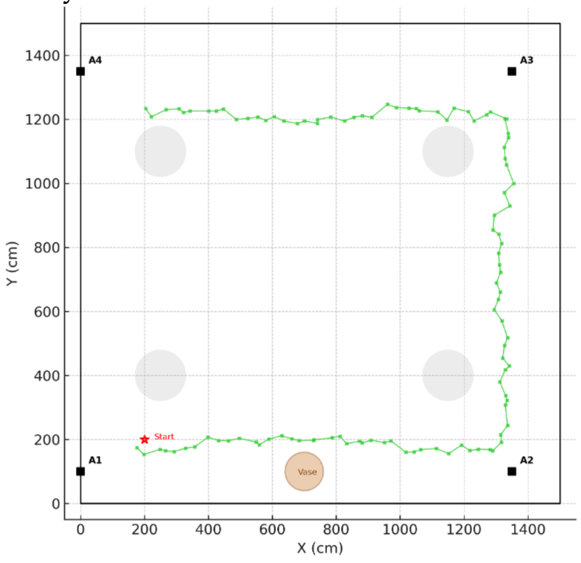
Figure 7: Designed RTLS architecture.

3.2. Test Scenarios and Analysis of Generated Data

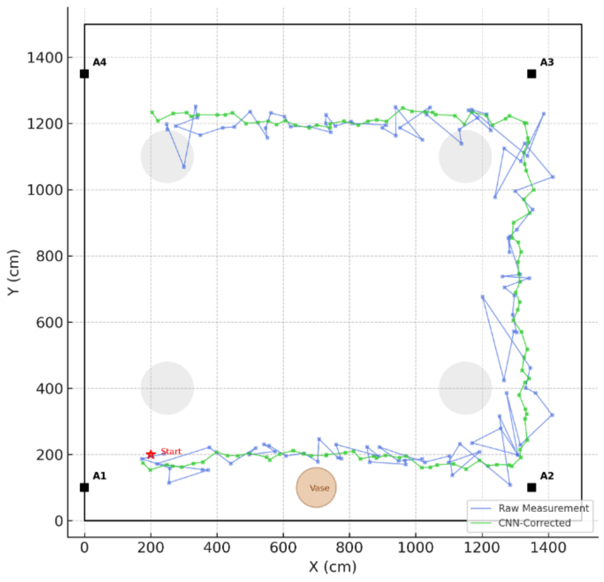
A Personnel Tag has activated in a corner of the office environment. There are four large columns in the environment, and there are metal ventilation pipes and wooden cabinets on the ceiling. All of this equipment causes NLOS conditions. Accordingly, the raw data first handled. Subsequently, the CNN/Autoencoder algorithm we proposed applied to the data, aiming to increase the stability of the system.



a) Raw noisy UWB position data.



b) UWB location data enhanced with autoencoder.



c) Comparison of the two measurements taken.

Figure 8: Test results obtained under NLOS conditions.

As seen in Figure 8 a), positional uncertainties occur, particularly during turns around columns, and significant performance losses are experienced. Figure 8 b) shows that the dispersion of position data occurring in angles and rotations that create the NLOS situation has improved. There has been an upward orientation on the Y-axis in the route caused by the pot located on the movement path. Our proposed autoencoder algorithm has distinguished this as an orientation on an axis in the route, rather than a distortion of position data caused by NLOS. Figure 8 c) clearly shows the improvement in position data by superimposing the data from the two situations.

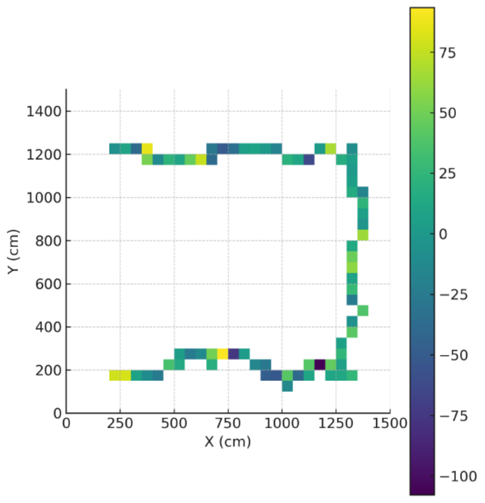


Figure 9: Heat map showing improvement in data obtained by applying the CNN-AutoEncoder algorithm.

As shown in Figure 9, the improvement rates in position data displayed using a heat map based on centimeters.

4. Discussion and Conclusion

In this study, an autoencoder-based enhancement layer proposed to suppress signal reflection and scattering caused by NLOS conditions in UWB-RTLS (DW1000, TWR) at the measurement layer. Tests conducted in a testbed demonstrate the practical and efficient implementation of the approach. The main objective is to compress feature vectors generated from the DW1000's side outputs (CIR-based indicators, first-path index/amplitude, RSSI, protocol flags, etc.) using the TWR methodology into a low-dimensional representation using an autoencoder to generate the reconstructed data. By learning "nominal" measurements collected under LOS conditions in an unsupervised manner, the autoencoder naturally eliminates NLOS-induced drift during reproduction in a real-time stream, resulting in more stable and less biased measurements reaching the location solver. Our experimental data demonstrates significant reductions in mean error metrics, significant improvements, particularly in high-percentage tails, and reduced jumps in and out of NLOS transitions, preserving track continuity. A key finding is that the method yields more stable results when CIR enhancement is applied compared to using range-based input alone. A key feature of the proposed approach is its modular and low-cost integration. The autoencoder inserted before the location solver as a "measurement-optimization" block; the DW1000 hardware or TWR protocol flow remains unchanged. Training data requirements are manageable thanks to the unsupervised setup, which largely based on LOS data; this is a significant advantage for NLOS scenarios where label generation in the field is difficult and expensive. The computational burden is light; single-pass extraction can performed in milliseconds with a small, shallow architecture. This allows the model to be trained with Python code and then applied to the system. In practice, it is also possible to record the reconstruction error as a confidence score and dynamically adjust the measurement weights in the location solver layer; this allows for more stable behavior than existing least squares, Gauss-Newton, or Kalman solvers without requiring radical architectural changes. Field-based inferences also reveal the method's significant advantages in terms of maintenance and operation. Fine-tuning can achieved with small LOS windows collected at short intervals. Parallel processing of data from multiple anchors is also possible; training the model with common features that are not anchor-specific simplifies maintenance and version management. Furthermore, the method adds accuracy to existing infrastructure without requiring additional hardware or complex stage calibrations, making it a cost-effective solution in NLOS-intensive areas such as production sites and logistics.

In order to develop a better system with future studies adaptive filtering schemes that adjust measurement covariance values online using confidence scores generated by the autoencoder could evaluated. Transfer learning and domain adaptation approaches for transitions between environments can enable rapid calibration with small amounts of target environment data. Explicitly modeling uncertainty (e.g., with ensemble approaches or dropout-based methods) contributes to risk-aware localization. Loosely or tightly coupled fusion with the IMU can further improve track continuity accuracy, particularly in moving tag scenarios. Finally, systematic validation on the DW3000 and different physical layer settings, across different architectures and scales will strengthen the method's production-grade generalizability. In summary, the autoencoder-based measurement-enhancement layer offers a tangible value proposition to DW1000-TWR-based UWB-RTLS links by combining gains in accuracy, stability, and track continuity with low integration cost and ease of real-time operation. With the development of hybrid time series setups, field adaptation, and uncertainty-focused strategies, it appears possible to further reduce location data errors under NLOS conditions and safely deploy to different sites.

5. Acknowledgments

The authors would like to thank R&D Center of Asis Automation and Fueling Systems Inc. for the test environment, technical support they provided during the realization of this study. Thanks for their support, for having opportunity to carry out this research.

References

[1] Coene S, Li G, Plets D, Tanghe E, Joseph W, Poorter E. Location-Aware Range-Error Correction for Improved UWB Localization. Sensors. 2024; 24(10): 3203. doi:10.3390/s24103203.

[2] Pei Y, Chen L, Zhang J, Chen Q. FCN-Attention: A Deep Learning UWB NLOS/LOS Classification Algorithm Using Fully Convolution Neural Network With Self-Attention Mechanism. Geo-spatial Information Science. 2024; 27(2): 251–266. doi:10.1080/10095020.2023.2178334.

[3] Niu Z, Li H, Zhao Y, Wang X. Deep Learning-Based Ranging Error Mitigation Method for UWB Using CIR Features. Computers and Electronics in Agriculture. 2023; 204: 107555. doi:10.1016/j.compag.2022.107555.

[4] Poulose A, Han D-S. UWB Indoor Localization Using Deep Learning: LSTM-Based Approach. Applied Sciences. 2020; 10(18): 6290. doi:10.3390/app10186290.

[5] Tran V, et al. Insights Into CIR-Based Data-Driven UWB Error Mitigation. Technical Report, University of Oxford; 2022. URL: https://ora.ox.ac.uk/objects/uuid:2ff9d847-f95c-4bd8-8dc5-8c004cdd87f0.

[6] https://www.everythingrf.com/community/what-is-ultra-wide-band-uwb-technology.

[7] Beauvisage, A., Ahiska, K., & Aouf, N. (2022). Robust multispectral visual-inertial navigation with visual odometry failure recovery. IEEE Transactions on Intelligent Transportation Systems, 23(7), 9089–9101.

[8] Hashim, H. A., Abouheaf, M., & Abido, M. A. (2021). Geometric stochastic filter with guaranteed performance for autonomous navigation based on IMU and feature sensor fusion. Control Engineering Practice, 116, 104926.

[9] Symmetry Electronics. (2024, July 15). An overview of DecaWave's DW1000 UWB wireless transceiver for precise indoor positioning. https://www.symmetryelectronics.com/blog/an-overview-of-decawave-s-dw1000-uwb-wireless-transceiver/.

[10] Bank, D., Koenigstein, N., & Giryes, R. (2020). Autoencoders. arXiv preprint arXiv:2003.05991.

[11] Bourlard, H., & Kabil, S. H. (2022). Autoencoders reloaded. Biological Cybernetics, 116(4), 389–406.

[12] Angarano, S., Mazzia, V., Salvetti, F., Fantin, G., & Chiaberge, M. (2021). Robust ultra-wideband range error mitigation with deep learning at the edge. Engineering Applications of Artificial Intelligence, 102, 104278.

[13] Park, J., Nam, S., Choi, H., Ko, Y.-E., & Ko, Y.-B. (2020). Improving deep learning-based UWB LOS/NLOS identification with transfer learning: An empirical approach. Electronics, 9(10), 1714.

[14] Fontaine, J., Ridolfi, M., Van Herbruggen, B., Shahid, A., & De Poorter, E. (2020). Edge inference for UWB ranging error correction using autoencoders. IEEE Access, 8, 139143–139155.

[15] Liu, W., Cheng, Q., Deng, Z., Chen, H., Fu, X., Zheng, X., Zheng, S., Chen, C., & Wang, S. (2019). Survey on CSI-based indoor positioning systems and recent advances. In 2019 International Conference on Indoor Positioning and Indoor Navigation (IPIN) (pp. 1–8). IEEE.