KEC Conference 2024
ISSN: 3021-940X (Print) DOI: https://doi.org/10.3126/injet.v1i2.66654

Particle Filter Enhanced by CNN-LSTM for Localization of Autonomous Robot in Indoor Environment

Rabin Giri[1,*], Anand Kumar Sah[2], Sanjivan Satyal[3]

[1], *Department of electronics and computer, Pulchowk Campus, Pulchowk, Lalitpur, Nepal, 075mscsk015.rabin@pcampus.edu.np*
[2]*Department of electronics and computer, Pulchowk Campus, Pulchowk, Lalitpur, Nepal , anand.sah@pcampus.edu.np*
[3]*Department of electronics and computer, Pulchowk Campus, Pulchowk, Lalitpur, Nepal, sanziwans@pcampus.edu.np*

Abstract

Particle filters play a crucial role in indoor localization tasks where GPS or similar sensors are unavailable, owing to their adaptability to dynamic environments and contribution to precise navigation. However, despite their effectiveness in non-Gaussian and non-linear settings, particle filters face challenges such as issues with particle initialization and recovery from scenarios like robot kidnapping. These limitations impede the complete autonomy of robots, a critical aspect of their functionality. This research presents a novel solution that integrates particle filters with a CNN-LSTM network to address these challenges. Leveraging the time-series image processing capabilities of CNN-LSTMs, this architecture aids in particle filter initialization and facilitates recovery from challenging situations. The integration enhances the overall performance and autonomy of robots, making them more efficient in indoor navigation tasks.

Keywords: AMCL, CNN, LSTM, Particle Filter, Kidnapped robot problem

1. Introduction

Localization refers to the process of determining the position and location of an object or entity within a specific environment. This involves estimating or establishing the object's coordinates or relative position in relation to a reference point or coordinate system.
Robot localization can be categorized into two main types: indoor and outdoor. Each presents unique challenges. This research focuses on addressing the challenges of indoor robot localization.
One of the primary challenges in indoor environments is the lack of access to direct absolute position information from sensors like GPS. While odometry and IMUs can provide relative position measurements, they suffer from drift, making them insufficient for accurate localization.
Furthermore, indoor environments are characterized by continuous environmental changes, such as moving objects, varying lighting conditions, and dynamic layouts. These factors introduce non-Gaussian noise and non-linear sensor models, making it difficult to accurately estimate the robot's pose. Additionally, real-world sensors are inherently prone to noise and errors, further impacting navigation accuracy. Therefore, localization algorithms for indoor navigation must be adaptable and robust to handle these complex scenarios.
Particle filters are the primary algorithm used for indoor localization due to their adaptability. They exhibit inherent robustness to uncertainties by maintaining multiple hypotheses about the robot's pose and dynamically adjusting them based on incoming sensor data, enabling effective navigation even in dynamic environments.

1.1 Problem Statement

Particle filters have certain limitations when used for robot localization.
Particle initial State: Particle filters are recursive Bayesian filters that estimate the robot's position recursively using motion and observation models. However, due to limitations in the number of particles that can be use proper initialization of these finite particles is crucial. Without proper initialization, the particle filter may fail to converge or take an excessively long time to reach the true state.

**Corresponding Author*

Robot Kidnapped Problem: Robot Kidnapped Problem occurs when the robot is unexpectedly relocated to a random position without its awareness.

Particle Degeneracy: It is a scenario where a few particles have high weights while the rest have negligible weights. This leads to insufficient exploration of the state space and reduced accuracy.

Recursive Bayesian estimation

(Thrun, 2002) Recursive Bayesian estimation (RBE) is a statistical and probabilistic approach commonly used in various fields, including robotics, to estimate the state of a system over time. This estimation method is rooted in the principles of Bayesian probability theory and is characterized by its recursive nature, where newly acquired information is continuously integrated into the estimation process. RBE is particularly useful in situations where the state of a system is subject to uncertainty and requires continuous updating

State Estimation

Estimate the state *x* of a system given observation *z* and control u

Goal: P (xt | z1:t , u1:t)

Let At time t belief of state x_t is Bel(x_t)

Input: observation information zt , control information ut ;

$Bel(x_t)$=$P(x_t|u_1, z_1, \ldots, u_t, z_t)$ (Equation 1)

Using Bayes

$\eta P(z_t|x_t, u_1, z_1, \ldots, u_t)P(x_t|u_1, z_1, \ldots, z_{t-1}, u_t)$ (Equation 2)

Using Markov

$\eta P(z_t|x_t)P(x_t|u_1, z_1, \ldots, z_{t-1}, u_t, x_{t-1})$ (Equation 3)

Using Total probability theorem

$\eta P(z_t|x_t) \int \quad P(x_t|u_1 z_1, .. u_t, x_{t-1})P(x_{t-1}|u_1, z_1, \ldots, u_t)dx_{t-1}$ (Equation 4)

Using Markov

$\eta P(z_t|x_t) \int \quad P(x_t|u_1, x_{t-1})P(x_{t-1}|u_1, z_1, \ldots, z_{t-1})dx_{t-1}$ (Equation 5)

$= \eta P(z_t|x_t) \int \quad P(x_t|u_t, x_{t-1})Bel(x_{t-1})dx_{t-1}$ (Equation 6)

.

Particle Filter

The particle filter algorithm type of Bayesian filter, is a probabilistic method used for state estimation and tracking in dynamic systems. It is particularly useful in situations where the system is nonlinear, and the state distribution is non-Gaussian or multimodal.

The particle filter algorithm represents the state of the system using a set of particles, where each particle represents a possible state hypothesis. These particles are propagated through time by using a dynamic model that describes the system's behavior. At each time step, the particles are weighted based on their likelihood of generating the observed measurements.

(Thrun, 2002) In particle filters, the particles represent samples from the posterior distribution. In the case of robot localization, these samples represent possible robot positions within the map.

$X_t: = x_t{}^{[1]}, x_t{}^{[2]}, x_t{}^{[3]}, \ldots, x_t{}^{[M]}$ (Equation 7)

Each particle $x_t{}^{[m]}$ with $1 \leq m \geq M$ is a concrete instantiation of the state at time t, that is, a hypothesis as to what the true world state may be at time t. Here *M* denotes the number of particles in the particle set X_t, $\omega_t{}^{[m]}$ denote weight of m particle at time t.

Algorithm Particle filter(X_{t-1}, u_t, z_t):

1: $\underline{X_t} = X_t = \phi$

2: for $m = 1$ to M do

3: sample $x_t{}^{[M]} \sim p(x_t|u_t, x_{t-1}^{[M]})$

4: $\omega_t^{[m]} = p(z_t|x_t^{[M]})$ (Equation 8)

5: $\underline{x_t} = \underline{X_t} + < x_t^{[M]}, \omega_t^{[m]} >$ (Equation 9)

6: endfor

7: for m=1 to M do
8: draw i with probability $\alpha\ \omega_t^{[m]}$
9: add $x_t^{[i]}$ to X_t
10: endfor
11: return X_t

2. Literature Review

A multitude of approaches exist for robot localization, each tailored to specific factors like sensor types, navigation complexity, and environmental characteristics. Extensive research has explored diverse approaches, including CNN based methods using camera images for landmark detection (Nilwong, et al., 2019), laser-based algorithms, wheel
encoder and IMU fusion techniques, and even combinations of these. However, the optimal technique depends heavily on the specific robot task at hand. To deal with the broad challenge of general indoor robot localization, algorithms must be prepared to handle a wide range of conditions
(Guo, et al., 2020) utilized Spherical Camera Images with ground truth grid positions to train a CNN-LSTM network for localization, framing the problem as a classification task. Integrating camera image data with IMU sensor data can significantly improve the accuracy of robot localization compared to relying solely on camera images. While cameras alone can lead to inaccurate orientation estimates, including IMU data in the training dataset allows the model to compensate for these limitations and achieve more precise results (Yusefi, et al., 2021).
The particle filter emerges as a consistent solution for robotics problems, particularly when modeling the uncertainty of the environment proves challenging and the fusion of multiple sensor data is required. In the context of robot localization based on laser scans and wheel odometry, the particle filter integrates odometry information with laser scan data. Initially, the particle filter represents the environment with multiple position hypotheses. It then predicts the new position of particles using odometry information, assigns proper weights based on laser scan matching, and finally, resamples the particles according to their weights. This iterative process transforms the particle filter's output into an accurate ground truth position, effectively addressing the localization problem. However, real environments are inherently complex and uncertain. Human navigation relies on vision, which provides extensive information, resulting in more accurate and robust localization compared to robots. This observation has prompted researchers to integrate vision into robot localization. Vision also proves valuable in robot path planning within navigation systems (Lu, et al., 2020), (Hoshino & Yoshida, 2022). Vision can address a wide range of challenges in robotics (Han, 2023), (Jiao, et al., 2019).
CNN-LSTM (Convolutional Neural Network - Long Short-Term Memory) stands out as a method to leverage visual information for solving robot localization problems (Patel, et al., 2018). This model processes time-series image data using CNNs and LSTMs to predict the robot's position in the environment. However, CNN-LSTM also has several limitations, such as an insufficient amount of proper data, high computation cost, required complex neural mode, etc., leading to accuracy problems. One way to solve this issue is to let CNN-LSTM predict the approximate robot position and utilize a particle filter-based approach to enhance accuracy with CNN-LSTM predictions (Xu, et al., 2019). In this way, we can remove the burden of the particle filter to predict the robot position in every situation. The job of the particle filter is only to maintain accuracy, and finding the approximate robot location is the job of the CNN-LSTM network.

3. Methodology

3.1 Environment Setup

The environment used in this research is both real and simulated. The simulated environment is designed using Gazebo, a popular physics engine that allows us to create diverse and sophisticated environments at no cost. Simulated environment encompasses nine interconnected rooms, featuring various obstacles and designated areas for robotic navigation.

The robot used in this research is a differential drive robot designed for simulation, visualization within a ROS2 environment, and testing in real-world settings. Its design is defined by a URDF file. Featuring two powered wheels for movement and one free caster wheel for stability, the robot is equipped with various sensors for
navigation and perception, including a lidar for ranging, two wheel encoders for odometry, and a camera for visual information.

Figure 1. Simulated and Real Environment

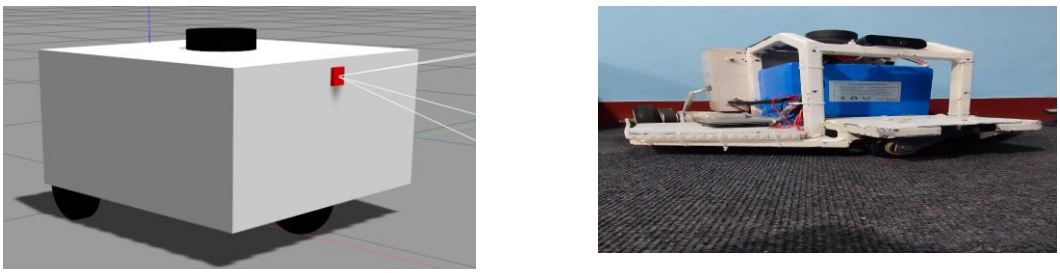

Figure 2. Robot Used in simulated and real environment

3.2. Localization and Data generation

Figure 3: Obstacle Map of simulated and real environment

For robot localization in an indoor environment, a 2D obstacle map is required. Robot environment mapping is conducted using the SLAM toolbox in ROS2. The process initiates with the acquisition of LiDAR data from the robot, offering insights into its surroundings. Time series data is generated by navigating a map.

AMCL helps track the robot's position. Both automatic and manual control mode is used. Auto-navigation takes the shortest path, missing some areas. To explore fully, joystick is used for manual control, capturing data from all possible coordinates

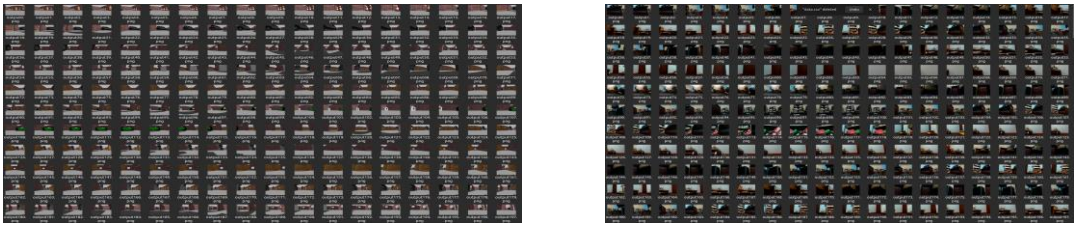

Figure 4. Time series images captured from the simulated and real environment

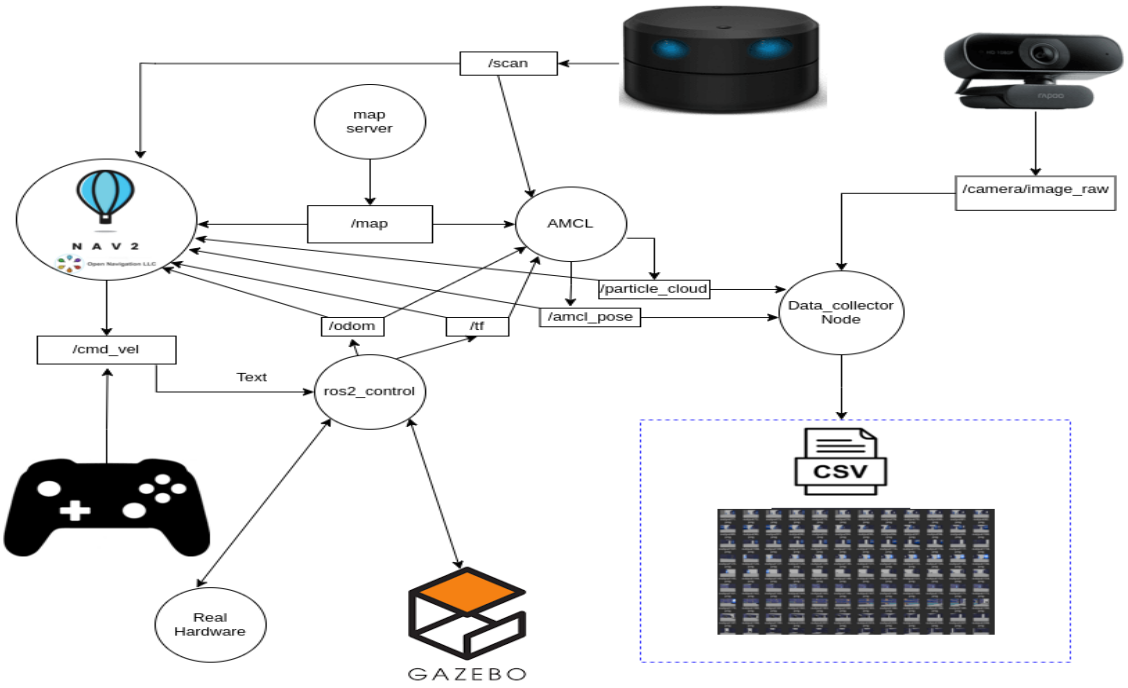

Figure 5. ROS2 Data generation block diagram

Data is recorded only when AMCL's position prediction is accurate, confirmed by particle cloud values. Time series images, labeled with coordinates, are saved to train a CNN-LSTM network. These images (224x224, color) are captured by a camera on the robot while moving. The coordinates include robot position (x, y) and yaw. This sequence of images will help the network predict the robot's location.

3.3. CNN-LSTM Network Training and Evaluation

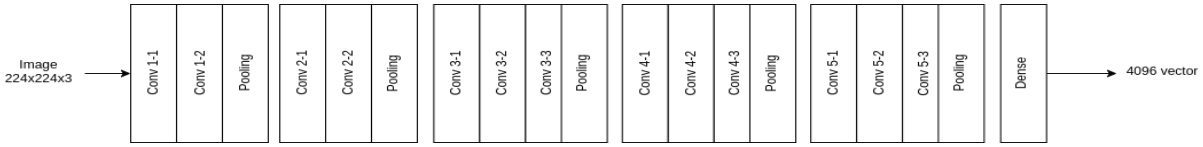

Figure 6. VGG16 layer for image embedding

The training phase of the network begins by encoding images with VGG16. All training and testing image sets are encoded using the fully connected layer of fc1 in VGG16, resulting in a 4096-dimensional output encoding vector.

The network is then trained with multiple LSTM layers. It starts with a single LSTM layer with 64 units, followed by two stacked LSTM layers with 128 and 64 units, respectively. The final output layer consists of 3 neurons.

The network uses linear activation functions, the Adam optimizer, and mean squared error (MSE) as the loss function. To monitor training progress, a checkpoint is implemented that tracks the validation loss. If the validation loss decreases compared to the previous epoch, the current network state is saved as the best model.

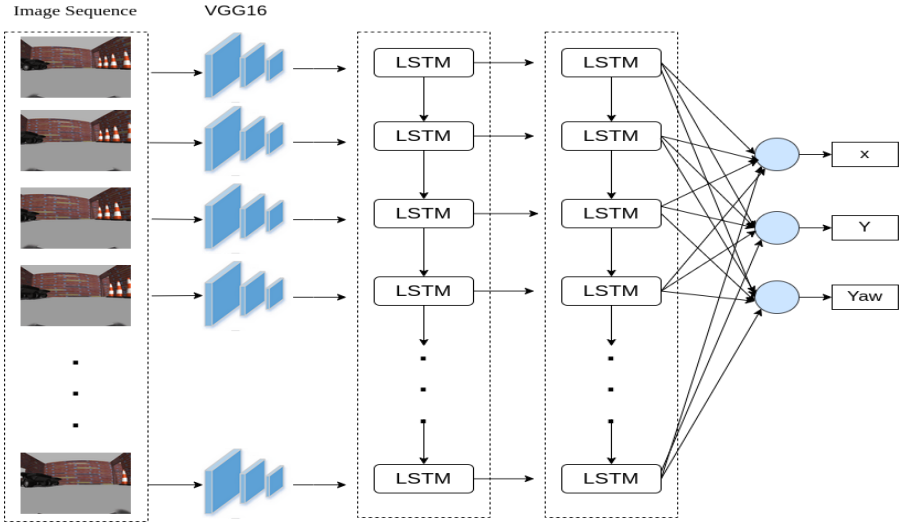

Figure 7. CNN-LSTM network block diagram

3.4. Particle Filter with CNN-LSTM network

In this research, the employed Particle Filter is the Adaptive Monte Carlo Localization (AMCL). For the CNN-LSTM component, a model trained with time series images labeled with robot positions is utilized to construct the CNN-LSTM node within ROS2. This node is designed to continually monitor the ROS2 AMCL node state through the /particle_cloud topic.

During the initial phase of robot localization, the CNN-LSTM node first observes the environment by sending velocity commands /cmdvel to the robot hardware, thereby predicting the current environmental location. Subsequently, it publishes the initial position of the robot using the /initialpose topic. The AMCL node initializes its particles with the information from the /initialpose message, and then recursively estimates the robot's position using Recursive Bayesian Estimation.

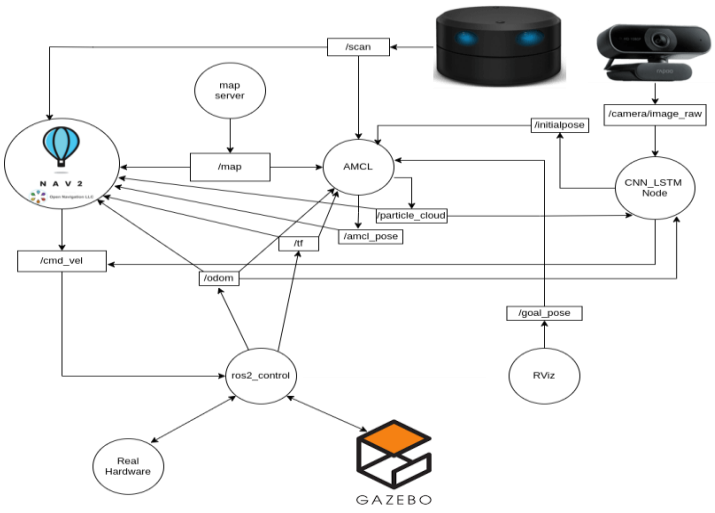

Figure 8. Proposed ROS2 echo system for improved localization system

In situations where the AMCL node encounters challenges in predicting the robot's location, it dynamically increases the number of particles to their maximum value. Simultaneously, the variance between particles escalates, a parameter monitored by the CNN-LSTM node. Additionally, the CNN-LSTM node continuously evaluates the robot's location through a sequence of images captured from the robot camera. In cases where the AMCL fails to predict the location accurately, the CNN-LSTM node takes charge, re-initializing the particles within the AMCL. This collaborative approach allows the AMCL and CNN-LSTM to enhance the localization process.

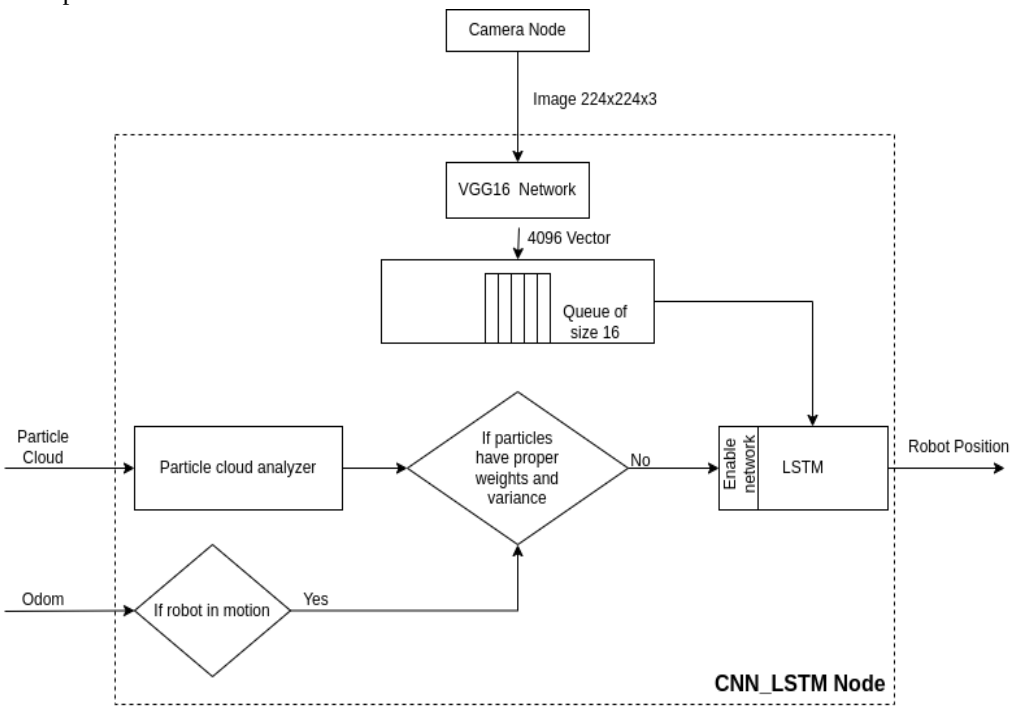

Figure 9. CNN-LSTM Node

3.5. Evaluation of proposed system

3.5.1 Average particles count

The number of particles in an Adaptive Monte Carlo Localization (AMCL) algorithm varies depending on the complexity of the localization task. Specifically, the"average particle count" refers to the average number of particles used throughout the localization process, from the starting point to the end point.

3.5.2 Localization error(LE)

Localization error is the distance between the estimated and true positions of the robot.

$$LE = \sqrt{(x_{true} - x_{estimated})^2 + (y_{true} - y_{estimated})^2} \quad \text{(Equation 10)}$$

4. Result and Analysis

4.1. CNN-LSTM network training and evaluation

Initially, multiple CNN-LSTM networks were trained and evaluated on simulated environment data. This involved testing various network architectures and tuning hyper parameters to identify the best-performing model for the specific needs of this research.

Through this process, we discovered that reusing a pre-trained CNN network, namely VGG16's fc1 layer with its 4096-dimensional output, as a feature extractor significantly outperformed training CNN from scratch.

Table 1. Training Parameter of 2 layer CNN-LSTM network

Hyperparameter	Value
Number of CNN Layer	2
Number of LSTM Layer	2
Input size	(15,224,224,3)
Output size	3
Input window size	15
Learning rate	0.001
Batch size	32
Epochs	10
Optimizer	Adam
Loss	mean squared error

Table 2. Result of 2 layer CNN-LSTM network

Output	Value
Training Accuracy	0.9735
Training Loss	0.029
Validation Accuracy	0.7278
Validation Loss	2.2214

Table 3. Training Parameter of LSTM with VGG16-embedded image input

Hyperparameter	Value
Number of LSTM layer	2
Input size	(15,4096)
Output size	3
Input window size	15
Learning rate	0.001
Batch size	32
Epochs	10
Optimizer	Adam
Loss	mean squared error

Table 4. Results of LSTM with VGG16-embedded image in simulated environment

Output	Value
Training Accuracy	0.9144
Training Loss	0.037
Test Accuracy	0.9117
Test Loss	0.0362

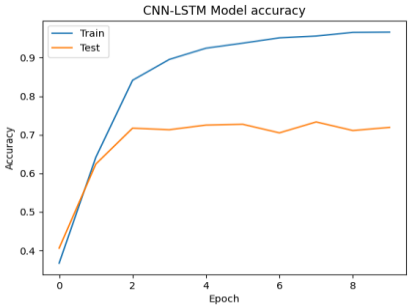

Figure 10. Accuracy vs epoch using simple CNN-LSTM

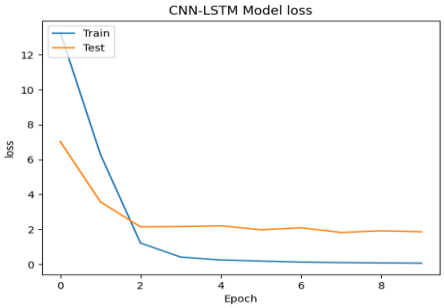

Figure 11. Loss vs epoch using simple CNN-LSTM

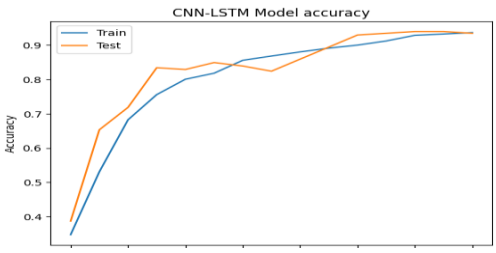

Figure 12. Accuracy vs epoch with embedded images in a Simulated environment

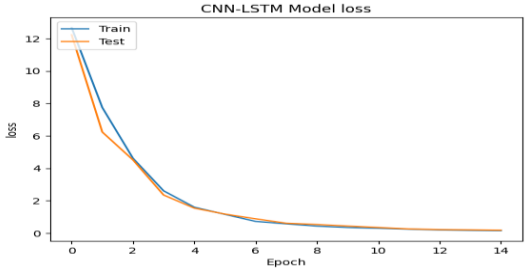

Figure 13. Loss vs epoch with embedded images in a Simulated environment

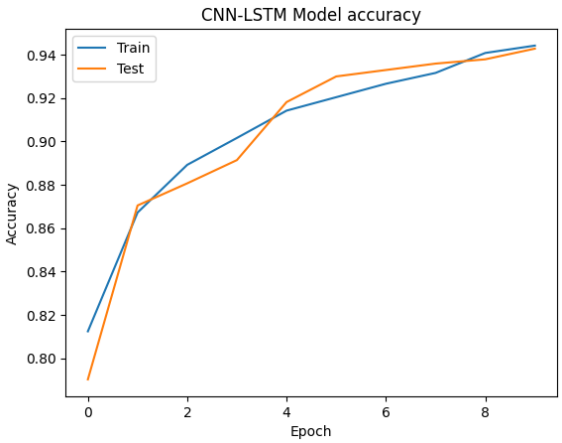

Figure 14. Accuracy vs epoch with embedded images in a Real environment

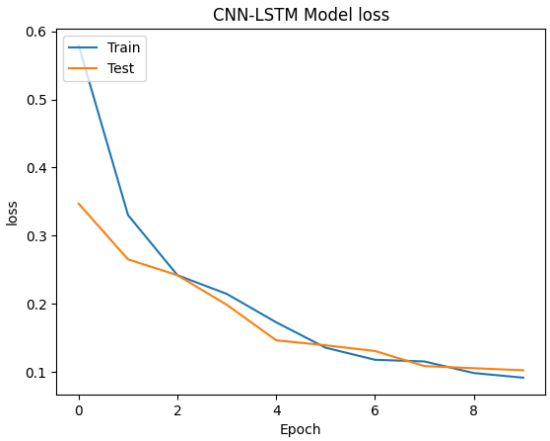

Figure 15. Loss vs epoch with embedded images in a Real environment

As seen from figure 4.1.1, the Test accuracy of the model is only 72.78%, and as from figure 4.1.2 the loss is 2.2214. This result was unsatisfactory for the robot localization task. Subsequently, all the images were encoded with the VGG16 network and trained exclusively on the LSTM network. The performance of the modified network is significantly better. From figure 4.1.3 and figure 4.1.4 the improved network has 91% test accuracy and loss is only 0.0362. The near-identical values for training and test accuracy, as well as loss, can be attributed to the nature of our dataset. Time series data is collected by repeatedly observing the environment from different perspectives to capture all possible scenarios. This led to the presence of overlapping sequences within both the training and test sets, resulting in similar behavior and hence, comparable performance metrics. Finally, the performance of a model trained with real environment is shown on figure 4.1.5 and figure 4.1.6, test accuracy of real environment is 94.45% and loss is 0.0113. The real-world environment delivers even better performance than the simulated environment, likely due to the increased feature richness found in real images. Observing Figure 4.1.5 and Figure 4.1.6, we can see that the real environment test accuracy reaches 94.45%, with a loss of only 0.0113. This surpasses the simulated environment's performance shown in Figure 4.1.3 and 4.1.4, where the test accuracy is 91% and the loss is 0.0362. The training and test data size of the real environment are 5000 and 1014, respectively.

4.2. Evaluation of Improved Localization

The enhanced localization technique employed in this research is a combination of the particle filter algorithm with the CNN-LSTM network. As mentioned earlier, the particle filter algorithm used in this research is AMCL. ROS2 framework is chosen because it fulfills all the requirements of the research, eliminating the need to rebuild the entire navigation ecosystem from the very beginning.

Initially, the CNN-LSTM network explores the environment by rotating the differential-drive robot around its origin and capturing time-series images of the environment in an attempt to estimate its initial coordinates. After that, the CNN-LSTM node initializes the particles in the AMCL node. Figure 4.2.1 and Figure 4.2.2 shows an obstacle map and the path where the robot is localized in a simulated and real environment. The red line shown in Figure 5.16 is the global path that the robot needs to follow. The black spot in the map represents the robot, and the small red dots are the particles of the AMCL node. The large green arrow in the figure indicates the target.

Proper initialization of particles is crucial for the AMCL node to estimate the robot's position. In the absence of external support at the start, an infinite number of particles is needed for position estimation. The CNN-LSTM detects the kidnapping of the robot when the majority of particles have very small weights, indicating that none of the particles accurately predict the robot's location. This situation is analogous to the challenge faced when the robot wakes up, and CNN-LSTM initializes the robot.

Figures 4.2.3 and 4.3.4 depict the cumulative error over time, with a focus on inducing errors in AMCL through dynamic changes in obstacle positions during the robot's runtime. In the absence of CNN-LSTM, the cumulative error is greater compared to its use. This discrepancy arises from the fact that, when the error of AMCL becomes significant, CNN-LSTM reinitializes its particles with estimated values, thereby mitigating the error. Consequently, the overall error of the combination of AMCL with CNN-LSTM consistently remains lower than that of standalone AMCL. This outcome also highlights that CNN-LSTM effectively addresses the particle degeneracy problem.

Figure 16: Global planner path in the simulated environment

Figure 17: Global planner path in the real environment

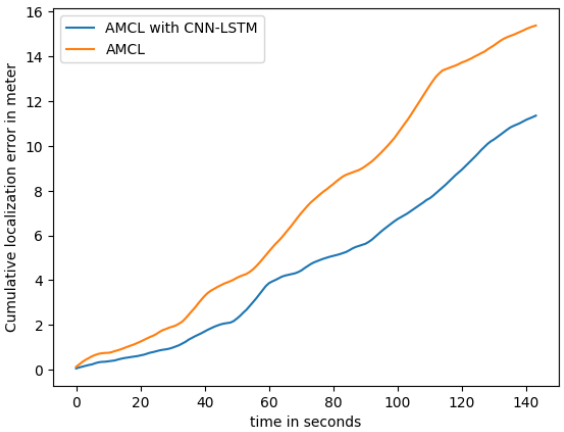

Figure 18:Cumulative localization error over time in the simulated environment test

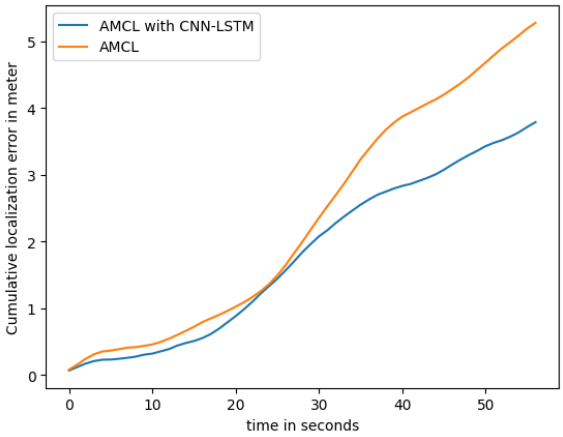

Figure 19: Cumulative localization error over time in the real environment test

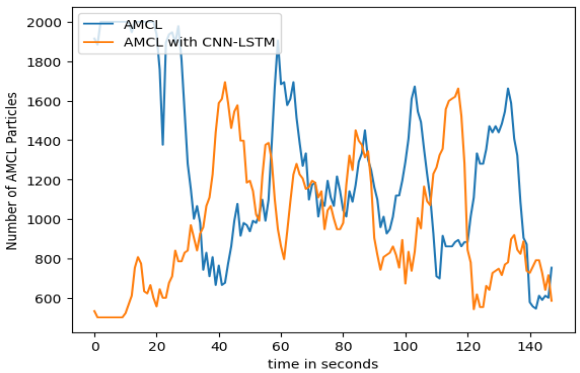

Figure 20. Number of particle with respect to time tested in simulated environment

Figure 4.2.5 depicts the dynamic adjustment of particle numbers by the AMCL node for calculating the robot's location. The illustration clearly indicates that when utilizing the CNN-LSTM node, a high initial particle count is unnecessary for AMCL during localization. It can properly initialize the particles during the starting phase. In contrast, in the absence of the CNN-LSTM node and with external assistance, AMCL commences localization with the maximum particle count. The average number of particles utilized by the AMCL node, with and without LSTM, is 966 and 1279, respectively. Without CNN-LSTM, the AMCL node initiates localization with an initial particle count of 2000, whereas with CNN-LSTM, it only requires 533 particles.

4.3 Comparison with existing system

In robotics, the ultimate goal is to provide practical solutions to human problems. Therefore, a technique's effectiveness in addressing these problems hinges not only on accuracy but also on its ease of implementation. This research endeavors to develop a technique that seamlessly integrates a CNN-LSTM model with a particle filter for robot localization, eliminating the need for human intervention. This research strategically implemented CNN-LSTM as a regression model, enabling the same deep learning model to operate universally across diverse indoor environments. This approach diverges from classification models, which require environment specific grid adjustments, rendering them unsuitable for universal deployment. Existing research methodologies present limitations in directly utilizing AMCL output for model training, necessitating additional equipment and human resources during system deployment. To circumvent these constraints, this research leveraged the regression model framework and developed a novel data generation node. This node autonomously constructs a suitable CNN-LSTM network training set by harnessing camera images and AMCL robot position data, a feature absent in current systems. This research proposed model demonstrates a reduced computational footprint compared to existing approaches. The CNN-LSTM node engages only when the AMCL node falters in accurately predicting the robot's position. In such instances, CNN-LSTM intervenes to reinitialize AMCL particle positions and seamlessly hands over control back to AMCL for continued localization tasks.

5. Conclusion

From the above experiment, it can be concluded that in the field of indoor robot localization, both AMCL and CNN-LSTM address the localization problem, but each has its limitations. The particle filter is quite accurate due to its adaptive nature in dynamic environments. However, it requires proper initialization at the starting phase and cannot predict the robot's location simply by observing the environment initially, as CNN-LSTM does. On the other hand, CNN-LSTM simply estimates the robot's position by observing the environment, but its estimate is not precise. The combination of CNN-LSTM's initial estimate with AMCL's recursive calculation solves all the problems in robot localization. There is no issue of degeneracy and robot kidnapping with CNN-LSTM, so when AMCL faces such problems, CNN-LSTM helps AMCL to overcome them. CNN-LSTM needs proper training datasets of images and robot positions so that it can predict the robot's location by observing time series images, but preparing the dataset is a very challenging task. This problem can be resolved by navigating the robot on the map, capturing images, and labeling the time series images with AMCL-estimated robot values, which means AMCL also helps CNN-LSTM obtain proper training sets. In conclusion, the combination of the particle filter and CNN-LSTM is beautiful; by combining these two algorithms localization problems can be solved.

The combination of particle filters and CNN-LSTM networks holds immense potential for building robust and cost-effective autonomous navigation systems. This has significant implications for various critical scenarios, including those where human intervention is limited, such as disaster response or remote exploration. Recent experiences like the COVID-19 pandemic highlight the need for such autonomous systems that can operate reliably and efficiently even in challenging environments.

Acknowledgement

The authors express their gratitude and extend their deepest appreciation to all the faculty members of the Department of Electronics and Computer, Pulchowk Campus for providing opportunities as well as valuable suggestions and feedback during the course of this research.

References

Guo, L., Shan, Y. & Li, S., 2020. *CNN-LSTM-Network Based Robot Localization Using Spherical Images of Neighboring Places.* s.l., 2020 IEEE 9th Global Conference on Consumer Electronics (GCCE).

Han, Z., 2023. Multimodal intelligent logistics robot combining 3D CNN, LSTM, and visual SLAM for path planning and control. *Front. Neurorobot.,* Volume 17 , p. 1285673.

Hoshino, S. & Yoshida, Y., 2022. *Motion Planner based on CNN with LSTM through Mediated Perception.* Kumamoto, Japan, IEEE.

Jiao, J. et al., 2019. MagicVO: An End-to-End Hybrid CNN and Bi-LSTM Method for Monocular Visual Odometry,. *IEEE Access,* Volume 7, pp. 94118-94127.

Lu, Y., Wang, W. & Xue, L., 2020. *A Hybrid CNN-LSTM Architecture for Path Planning of Mobile Robots in Unknow Environments.* s.l., Chinese Control and Decision Conference.

Nilwong, S., Hossain, D., Kaneko, S.-i. & Capi, G., 2019. Deep Learning-Based Landmark Detection for Mobile Robot Outdoor Localization. *Machines* , 7(2).

Patel, M., Emery, B. & Chen, Y.-Y., 2018 . *Contextualnet: Exploiting contextual information using lstms to improve image-based localization.* s.l., IEEE International Conference on Robotics and Automation (ICRA).

Thrun, S., 2002. Probabilistic robotics. *Communications of the ACM,* 45(3), pp. 52-57.

Xu, S., Chou, W. & Dong, H., 2019. A Robust Indoor Localization System Integrating Visual Localization Aided by CNN-Based Image Retrieval with Monte Carlo Localization. *Sensors* , 9(2), p. 249.

Yusefi, A., Durdu, A., Aslan, M. F. & Sungur, C., 2021. LSTM and Filter Based Comparison Analysis for Indoor Global Localization in UAVs. *IEEE Access,* Volume 9, pp. 10054-10069.