A Novel Deep Learning Based Approach to Solve KRP for Autonomous Mobile Robots Relocalization Within Indoor Known Maps (2D Lidar)

by

Belkacem BEKKOUR

A thesis submitted to the
Department of Computer Science
in conformity with the requirements for
the degree of Master of Science

Bishop's University
Canada
June 2025

Abstract

Localization is a critical topic for Autonomous Mobile Robots (AMRs) operating within indoor environments. While laser based methods have shown a robust localization performance, they often face challenges such as the so called Kidnapped Robot Problem (KRP) and the relocalization problem or the initial localization, specifically in large scale environments. Monte Carlo Localization (MCL) has been proposed as a solution to these challenges; however, its computational inefficiency in expansive environments and its lack of robustness in ambiguous and highly dynamic settings, such as logistics and warehouses, limit its real time applicability. In this thesis, we propose a novel deep learning based dual framework approach to enhance the initial localization and relocalization capabilities of AMRs using 2D LiDAR point clouds, focusing on addressing the KRP. The proposed system integrates global and local localization strategies to achieve an accurate rough alignment of the robot's pose during initialization or after kidnapping events. For global zoning, advanced convolutional neural network (CNN) architectures such as GoogLeNet, VGG16, VGG19, and DenseNet are utilized, demonstrating exceptional accuracy and robust generalization by leveraging features from occupancy grid maps. For local localization, the YOLOv8 model is applied to Signed Distance Transform (SDT) maps visualized with a seismic colormap, enabling precise keypoint detection and achieving high precision recall performance metrics. To further enhance robustness, we have also introduced an automatic data augmentation tool, capable of simulating random occlusions, added objects, and rotations to enrich the training datasets and improve the system's resilience under real world conditions. The approach proposed has demonstrated significant advancements in AMR laser based initial localization within large environments, offering a reliable solution for initial localization and relocalization, with potential future applications in hierarchical feature extraction and enhanced pose estimation.

Acknowledgments

I would like to express my sincere gratitude to my supervisors, Dr. Mohammed Ayoub Alaoui Mhamdi and Dr. Madjid Allili , for their supervision and guidance throughout this research work.

I am especially thankful to Professor Sousso Kelouwani, from the Department of Mechanical Engineering at UQTR, for granting me the opportunity to conduct my research at the Hydrogen Research Institute, and for the trust he placed in me by offering a research scholarship. His teaching in the Advanced Optimization course also had a valuable impact on my academic background, Special thanks to Dr. Yasir Malik, Dr. Russell Butler and Dr. Stefan Bruda, Graduate Program Coordinator at Bishop's University, for his ongoing support and administrative guidance, and to Professor Layachi Bentabet, Head of the Department of Computer Science at Bishop's University, for providing the academic environment and resources that made this research possible. I also wish to thank Noovelia for their collaboration and trust, and to express my heartfelt appreciation to Professor Dalila Cherifi for her mentorship, encouragement, and insightful advice throughout my academic path. I am deeply grateful to my parents, Mr. Youcef Bekkour and Mrs. Zahia Zizi, for their unwavering support, sacrifices, and love, which have been the foundation of all my accomplishments. I gratefully acknowledge the Government of Quebec (Canada) for offering an exemption from international tuition fees for the 2024/2025 academic year, which significantly contributed to the success of this academic journey. Lastly, I extend my sincere thanks to all the professors who shaped my academic journey, particularly those at the Institute of Electrical and Electronic Engineering (IGEE, formerly INELEC) in Boumerdes, whose early guidance laid the groundwork for my career in engineering.

Contents

List of Tables

List of Figures

Chapter 1

Introduction & Problem Statement

1.1 Autonomous Mobile Robots and Indoor Localization

Autonomous Mobile Robots (AMRs) are computer-controlled systems designed to navigate and perform tasks independently in complex and dynamic environments without human intervention. These robots integrate advanced sensory technologies, computational systems, and intelligent algorithms, enabling adaptation to diverse scenarios and applications [30]. AMRs have become essential in multiple sectors, demonstrating significant versatility and the potential to revolutionize various industries. In industrial automation, AMRs replace human labor in repetitive, hazardous, or precision demanding tasks, particularly in assembly lines and warehouses [21, 40]. The healthcare sector employs AMRs to streamline hospital workflows, automate medical-supply delivery, assist patient monitoring, and perform critical tasks like disinfection [40, 21]. In agriculture, AMRs assist in planting, harvesting, and crop-health monitoring, while also supporting operations in hazardous environments, including mining, underwater exploration, and extraterrestrial missions [30].

The core functionality of AMRs encompasses perception, cognition, locomotion, and navigation. Perception relies on exteroceptive sensors principally LiDAR or cameras for obstacle detection and environmental mapping [30]. Cognition refers to decision-making, often implemented with advanced algorithms and machine-learning techniques. Locomotion concerns the robot's mechanical ability to traverse varied terrains. Navigation involves localization, path planning, and obstacle avoidance, all of which depend on robust mapping technologies [40, 40].

Indoor, GPS-denied environments impose stringent localization requirements. While 2-D LiDAR offers a practical balance of accuracy, real-time processing, and cost [43], its sparse point clouds limit environmental context, and sensor noise and occlusions degrade accuracy, especially when the robot starts without any pose estimate [48]. Hence, hybrid global–local strategies are needed [38]. Achieving full AMR autonomy further requires resilience to dynamic scenes and unforeseen scenarios [2].

Figure 1.1: Representative AMR applications in industry (left), last-mile delivery (centre), and healthcare (right).

1.2 Problem Statement: The Kidnapped Robot Problem

The Kidnapped Robot Problem (KRP) or the initial localization problem arises when a robot is physically displaced without its proprioceptive sensors detecting the motion, causing complete loss of pose [5, 42]. This failure mode jeopardizes safety-critical tasks and interrupts mission workflows [16]. Relocalization based on 2-D LiDAR reference maps [9, 24] is hampered by scan sparsity [8], dynamic occlusions, sensor noise [23], and feature-poor settings such as corridors [4]. Consequently, a computationally efficient, geometry-tolerant framework is required.

Key Challenges

- **Occlusions and dynamics**: Moving objects obscure salient map features.
- **Sparse LiDAR measurements**: Limited geometric information complicates feature extraction.
- **Sensor noise and misalignment**: Introduce false matches and large pose errors.
- **Real-time constraints**: Mobile platforms impose tight computational budgets.

1.3 Overview and Limitations of Existing localization Techniques

This section reviews principal localization paradigms and summarises their limitations with respect to KRP and dynamic indoor settings.

Method	Strengths	Limitations
Odometry /Dead-Reckoning	Simple, low cost	Cumulative error growth
Kalman Filter	Real-time, noise-aware	Assumes linearity; needs good initial state
Monte Carlo localization (MCL)	Handles multi-modal beliefs	Computationally intensive in large spaces
Deep-Learning Approaches	Robust to appearance change	Data-hungry; limited interpretability

Table 1.1: Representative indoor localization techniques.

1.3.1 Particle-Filter Methods: MCL and AMCL

Particle filters are a class of probabilistic localization algorithms that estimate the state of a mobile robot based on a set of weighted hypotheses, commonly referred to as particles. These methods are well-suited for non-linear and multi-modal localization problems, however they struggle in large and highly changing environments as they are computationally expensive and they all rely on the initial pose. Monte Carlo Localization is a probabilistic approach that represents a robot's belief about its pose as a set of particles [27] as shown in Figure 1.2. Each particle corresponds to a possible pose, with a weight assigned based on its agreement with sensor measurements Figure 1.3. MCL operates through three primary steps: prediction, measurement update, and resampling which is demonstrated in the Figure 1.5 and Figure 1.4 respectively. The weight for particle m at time t is

$$w_t^{[m]} = p\big(z_t \mid x_t^{[m]}, m\big), \tag{1.1}$$

where z_t is the LiDAR observation and m the map. The posterior is approximated via

$$P(x_t \mid z_{1:t}, u_{1:t}) \approx \sum_{m=1}^{M} w_t^{[m]}\,\delta\big(x_t - x_t^{[m]}\big). \tag{1.2}$$

Adaptive MCL (AMCL) reduces computation by varying the particle count according to Kullback–Leibler divergence [11]. The required number of particles is

$$N = \frac{1}{2\epsilon}\,\chi^2_{1-\delta,\,k-1}, \tag{1.3}$$

with error tolerance ϵ, confidence δ, and k histogram bins.

It is also important to note that some recent studies have explored hybrid approaches combining deep learning models [22] with particle filtering to improve relocalization efficiency in large-scale environments. Future may focus on integrating learning based observation models and multi-modal sensor fusion to enhance robustness in complex indoor scenarios. However, these techniques rely on a specific distribution, and make it not convenient in highly changeable maps.

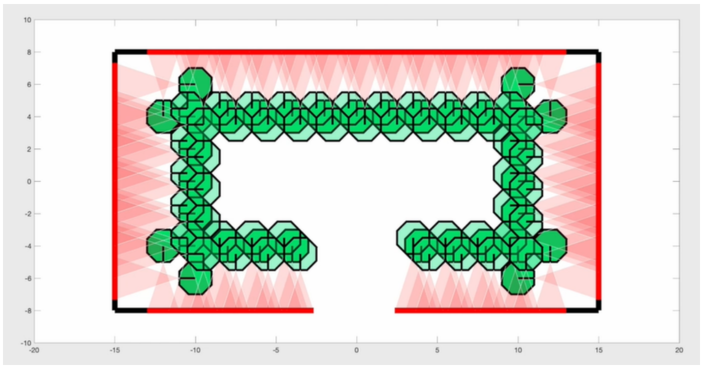


Figure 1.2: Initial uniformly sampled particles in MCL.

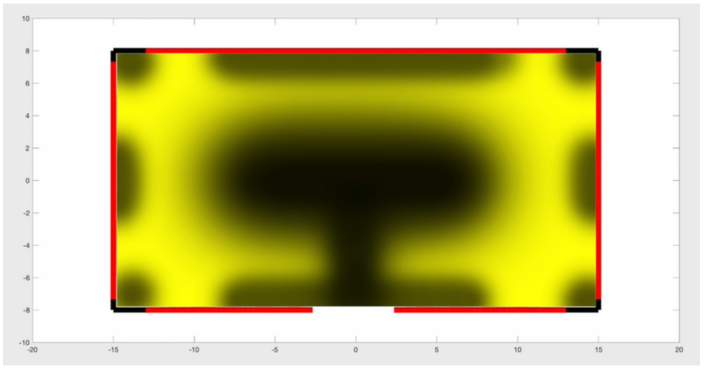


Figure 1.3: Probability density function of particle weights.

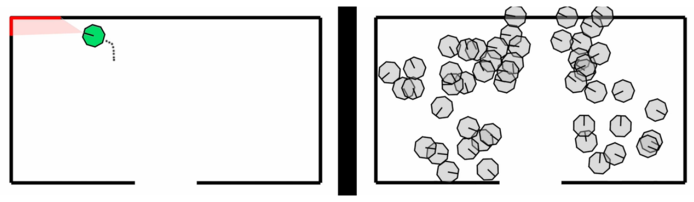


Figure 1.4: Particle convergence toward the true pose.

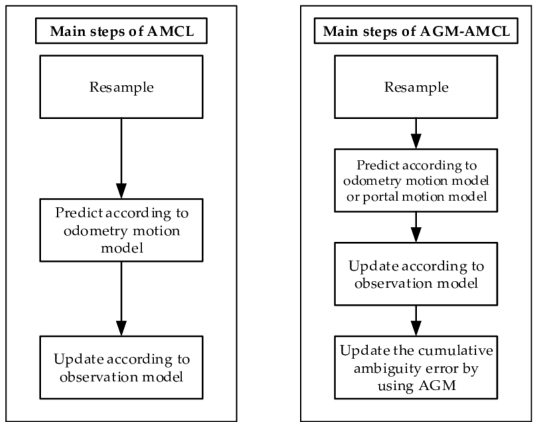

Main steps of AMCL
Resample
Predict according to odometry motion model
Update according to observation model
Main steps of AGM-AMCL
Resample
Predict according to odometry motion model or portal motion model
Update according to observation model
Update the cumulative ambiguity error by using AGM

Figure 1.5: Qualitative comparison of MCL and AMCL.

Feature	**MCL**	**AMCL**
Particle count	Fixed	Adaptive based on uncertainty
Computational efficiency	Can be high in large environments	More efficient by reducing unnecessary particles
Handling (KRP)	Struggles with sudden relocations	More robust due to adaptive particle redistribution
robustness to dynamic environments	Limited	Better but still challenged by rapid environmental changes

Table 1.2: Comparison of Monte Carlo Localization (MCL) and Adaptive Monte Carlo Localization (AMCL) [27]

1.3.2 State-Estimation Filters: KF and EKF

The linear Kalman filter (KF) [20] yields optimal estimates for Gaussian linear systems. Prediction and update obey

$$\hat{x}_k^- = A_k \hat{x}_{k-1} + B_k u_k, \tag{1.4}$$

$$P_k^- = A_k P_{k-1} A_k^T + Q_k, \tag{1.5}$$

$$K_k = P_k^- H_k^T \big(H_k P_k^- H_k^T + R_k\big)^{-1}, \tag{1.6}$$

$$\hat{x}_k = \hat{x}_k^- + K_k \big(z_k - H_k \hat{x}_k^-\big). \tag{1.7}$$

The Extended Kalman Filter (EKF) linearises around the current estimate with Jacobians F_k and H_k [19, 42]. Despite wide adoption, both filters depend on accurate initialisation and are brittle under severe non-linearities.

Feature	KF	EKF
Model assumption	Linear	Linearised nonlinear
Computational load	Low	Moderate (Jacobians)
Robustness to dynamics	Limited	Improved but approximation errors

Table 1.3: Comparison of KF and EKF.

1.3.3 Pose-Graph SLAM

Pose-graph approaches formulate SLAM as non-linear least squares optimisation over robot poses (nodes) and spatial constraints (edges). They excel at loop closure but require a reasonably aligned initial trajectory and grow computationally expensive in large maps [15]. The Figure 1.6 and Figure 1.7 show the misalignment and the correction using the pose graph, respectively. However, they are not used for initial localization but rather for unknown environment Mapping and localization simultaneously.

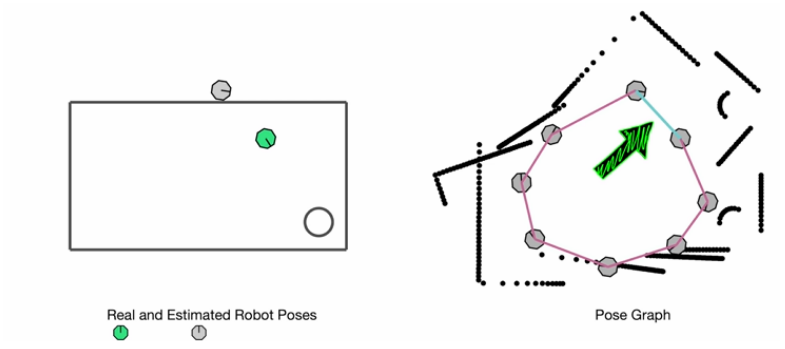


Figure 1.6: Misaligned pose graph prior to optimisation.

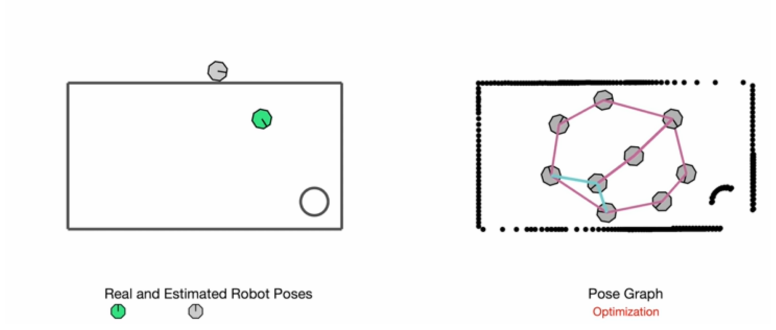


Figure 1.7: Pose-graph correction after optimisation.

1.3.4 Feature- and Scan-Matching Methods

1. **Grid & ICP:** Scan matching aligns current LiDAR scans with a pre-built map using ICP [3] or [35]. These demand good initial alignment and degrade in repetitive geometry [29].

2. **Keypoint Matching**: Algorithms such as SIFT [26] and ORB [34] extract geometric keypoints for relocalization but falter under occlusion and sensor noise [36, 46]. Figure 1.8 illustrates a typical sucess, they act as initial guess for ICP.

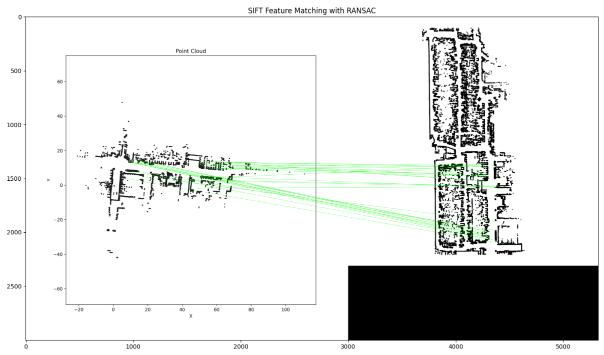


Figure 1.8: Attempted initial localization using SIFT keypoints; mismatches arise in dynamic scenes.

3. **Correlation-Based**: Correlation-based techniques compute the likelihood of a pose by correlating LiDAR data with a map. Techniques like Monte Carlo correlation improve localization robustness by sampling possible poses and comparing likelihood [29, 31]

1.3.5 Global Localization Using Other Sensors

Global localization refers to the process by which a mobile robot determines its position within a known map without any prior knowledge of its initial location (Global Area X,Y and θ). Beyond 2D LiDAR, other sensors have been explored for global localization, either standalone or in combination with LiDAR :

Camera-Based Methods

Cameras provide rich environmental data, enabling visual odometry and scene understanding. CNNs have been widely used for RGB image-based global or local localization as shown in the Figure 1.9, extracting features to match against a visual map [7]. These methods are sensitive to lighting changes and occlusions, limiting their reliability in dynamic environments [47],

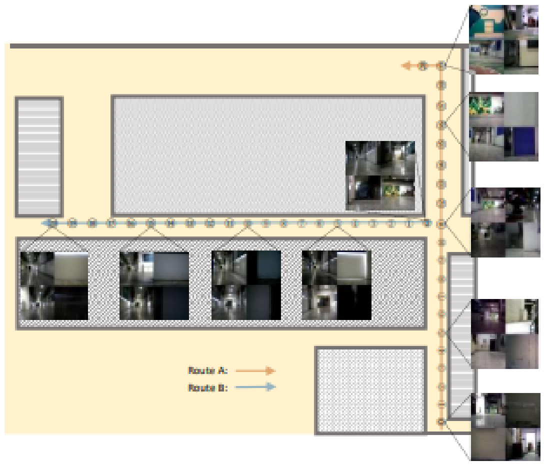


Figure 1.9: visual landmarks based localization

GPS-Based Localization

GPS is a common choice for outdoor localization due to its global availability, high accuracy in open environments, and ease of integration with various devices. However, it is impractical for indoor environments due to signal loss and multipath effects. High-accuracy GPS, such as Real-Time Kinematic, can offer precise localization outdoors but requires additional infrastructure [4].

Sensor Fusion Approaches

Sensor fusion combines multiple sensors (e.g., LiDAR, camera, IMU, SONAR) to enhance localization accuracy [44]. Techniques like EKF and graph-based optimization are used to fuse data, leveraging complementary strengths of sensors. However, these methods increase system complexity and computational requirements.

Role of Rough Alignment in Localization

Nearly all localization techniques depend on reliable **initial localization** or **rough alignment** [44]. This step provides a coarse pose estimate, enabling other methods (e.g., particle filters, graph-based optimization) to converge effectively. The proposed CNN-based global zoning framework addresses this critical requirement by leveraging 2D LiDAR data to classify zones accurately, serving as a robust foundation for further localization tasks.

Motivation and Proposed Deep-Learning Framework

Classical localization pipelines whether filter based, particle ased, or scan matching suffer from four persistent limitations: reliance on a coarse initial pose to ensure convergence; computational cost that scales unfavourably with map size or particle count; vulnerability to occlusion, scene dynamics, and sensor noise; and restrictive assumptions about sensor modalities or motion models [27, 44]. These constraints motivate the exploration of data-driven methods that can learn robust geometric signatures directly from sparse 2-D LiDAR data.

Research Objectives

1. **Develop a global, learning-based zoning strategy** that provides a reliable initial pose estimate without handcrafted feature engineering.

2. **Design a fine, keypoint-based local refinement stage** that improves pose accuracy within the selected zone and remains resilient to occlusions and dynamic obstacles.

3. **Validate the complete pipeline in simulation** across diverse, photorealistic scenarios generated via an automatic `.pgm` map-modification tool, demonstrating robustness under partial map mismatch and sensor noise.

Proposed Solution

1. **Global Zoning with CNNs:** This approach employs CNN-based architectures (GoogLeNet, DenseNet, VGG) to analyze features from occupancy grid maps generated using 2D LiDAR data. The framework provides robust initial zone-level localization, serving as a deep learning based alternative to traditional methods and enhancing Monte Carlo Localization (MCL) with precise initial pose estimates.

2. **Fine Localization via YOLOv8 Pose Estimation:** Utilizing Signed Distance Transform (SDT) maps with seismic-style visualization, YOLOv8 accurately detects local keypoints. This stage refines the robot's pose within the identified zone, significantly improving localization precision.

3. **Validation through Simulation:** The framework's effectiveness and robustness are demonstrated through simulation tests, confirming its performance in dynamic conditions and scenarios involving partial occlusions. These results underline the solution's potential applicability for practical deployment in autonomous mobile robot navigation within indoor, two-zone environments. Scenarios were generated using an automatic `.pgm` modification tool.

To summarize, MCL, AMCL, KF and EKF all excel at continuous localization in known maps, but not for initial localization [12], by fusing a motion model with 2D LiDAR measurements, but they remain vulnerable to failures in the motion model e.g., the Kidnapped Robot Problem if odometry drifts or the map is suddenly mismatched. Pose-graph methods extend this to SLAM in unknown environments, still combining motion and LiDAR data to optimize pose graphs. For global initialization, feature matching (SIFT or ORB) can provide an initial guess to ICP so that scan-to-map alignment converges more reliably, they are manly used in mapping or registration in general but not specifically for initial localization [1] or in visual camera features [14], though LiDAR scan errors often caused by magnetic disturbances can still break convergence. While camera-based methods, GPS, or multi-sensor fusion offer alternative initial-pose solutions, this work focuses exclusively on 2D LiDAR approaches.

Chapter 2

Background on Deep Learning Architectures

2.1 Fundamentals of Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a class of deep learning models designed to automatically and adaptively learn and encode dominant spatial hierarchies of features from inputs, such as images or occupancy grids or its SDT version. By stacking convolutional, activation, and pooling layers, CNNs capture both local and global patterns, making them particularly effective for tasks like feature extraction in 2D LiDAR–based global zoning.

2.1.1 Convolution, Activation & Pooling Layers

Convolution Operation

The convolution operation applies a kernel K to an input feature map I to produce an output feature map y:

$$y_{m,n} = \sum_{i,j} I_{m+i,n+j} \cdot K_{i,j} \tag{2.1}$$

As illustrated in Figure 2.1, each output pixel is the weighted sum of a neighborhood in the input feature map [33].

Activation Functions

Activation functions introduce non-linearities that enable CNNs to learn complex patterns. Common choices include ReLU, Mish, and Leaky ReLU. For example, ReLU is defined as:

$$f_{\text{ReLU}}(x) = \max(0, x) \tag{2.2}$$

Figure 2.2 shows the shapes of these functions [33].

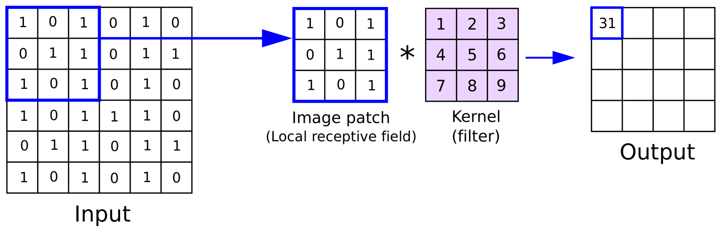


Figure 2.1: Illustration of the convolution operation where a kernel is applied to an input matrix to produce a feature map. Adapted from [33].

Pooling Layers

Pooling layers downsample feature maps to reduce spatial dimensions and control overfitting. Max-pooling selects the maximum value in each window:

$$P_{k,l} = \max_{i,j \in W_{k,l}} (x_{i,j}) \tag{2.3}$$

As shown in Figure 2.3, this operation significantly reduces the size of feature maps while retaining the most salient activations.

2.1.2 Learning Rate & Optimizers

Model weights are updated via gradient descent according to:

$$w_{\text{new}} = w_{\text{old}} - \eta \nabla L \tag{2.4}$$

where η is the learning rate. Two widely used optimizers are:

- **SGD** (Stochastic Gradient Descent), optionally with momentum.
- **Adam** (Adaptive Moment Estimation), which adaptively scales per-parameter learning rates using first and second moment estimates.

2.1.3 Regularization Techniques

To prevent overfitting, several regularization methods are standard:

Weight Decay Adds an L_2 penalty on weights:

$$L' = L + \lambda \tfrac{1}{2} \|w\|^2 \tag{2.5}$$

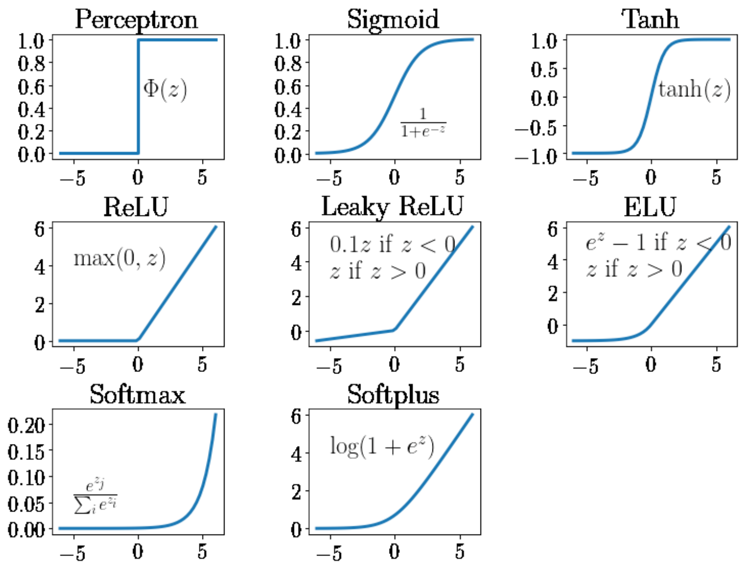


Figure 2.2: Various activation functions used in neural networks, highlighting differences and applications. Adapted from [18].

Batch Normalization Normalizes layer activations to reduce internal covariate shift:

$$B(x) = \gamma\left(\frac{x-\mu_B}{\sigma_B}\right) + \beta \tag{2.6}$$

where μ_B and σ_B are computed over each mini-batch.

Dropout Randomly zeroes activations during training:

$$y = x \cdot D \tag{2.7}$$

with D a Bernoulli mask (keep probability p).

2.2 Architectural Features of Selected CNNs

This section reviews three CNN architectures chosen for global zoning based on their structural innovations and performance characteristics.

2.2.1 GoogLeNet (Inception Modules)

GoogLeNet, also known as Inception v1 [41], is built upon the idea of network-in-network modules, termed Inception modules that perform parallel convolutions at multiple scales (1×1, 3×3, two stacked 3×3 to approximate 5×5) alongside pooling layers, then concatenate all outputs. This design captures both fine and coarse

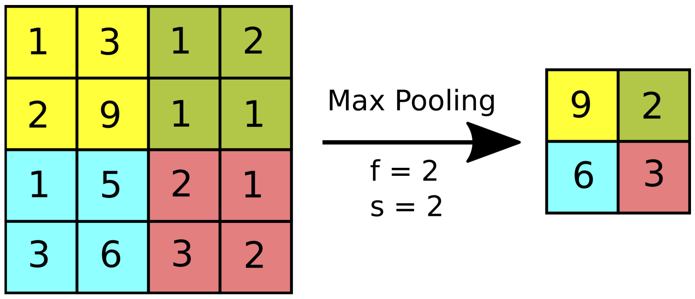


Figure 2.3: Illustration of max pooling operation reducing the spatial dimensions of feature maps [33].

features efficiently and with relatively few parameters [32] the global architecture of inception modules is illustrated in Figure 2.4.

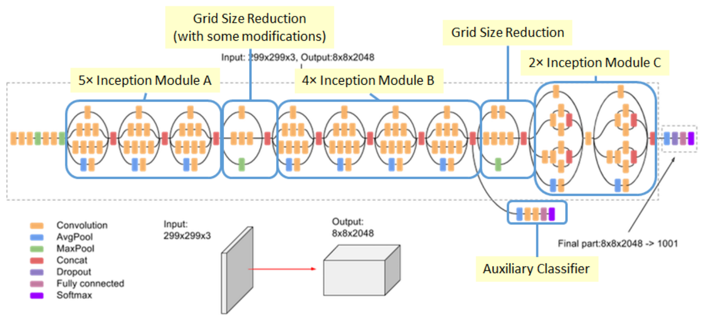


Figure 2.4: Architecture of an Inception module [32].

2.2.2 DenseNet (Dense Connections)

DenseNet connects each layer to every other layer in a feed-forward fashion, ensuring maximum feature reuse and strong gradient flow. Transition layers between dense blocks compress feature maps and reduce dimensionality, keeping the network compact and mitigating the vanishing gradient problem [17]. A simple dense connection block is illustrated in Figure 2.5. Specifically, the output feature maps of each layer are concatenated with those of all preceding layers and passed as input to every subsequent layer within the block. This design distributes contextual information throughout the network, so that each layer can directly leverage the combined representations learned by all of its predecessors.

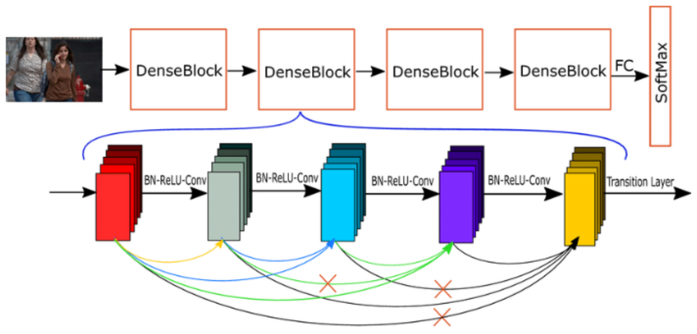


Figure 2.5: Illustration of dense connections in DenseNet [10].

2.2.3 VGG16/VGG19 (Deep Stacks of 3×3 Filters)

VGG networks use only 3×3 convolutions stacked deeply 13 convolutional layers plus 3 fully connected layers in VGG16, extended to 16 convolutional layers in VGG19. As shown in Figure 2.6, this uniform design makes them easy to implement and effective at capturing hierarchical spatial patterns, at the cost of higher parameter counts [37].

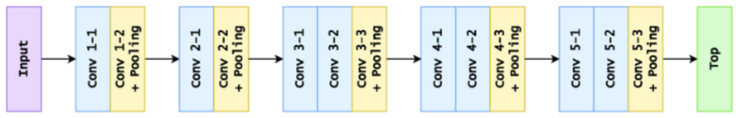


Figure 2.6: VGG16 architecture illustrating its depth and convolutional layers [6].

To summarize the strengths, weaknesses, and core ideas of each CNN architecture, the Table 2.1 below provides an overview.

2.3 Enhanced Keypoint Detection with YOLOv8 for Local Localization

Architecture and Feature Extraction

The architecture of YOLOv8 is designed to efficiently process high dimensional data and extract features that are essential for localizing keypoints. It employs advanced convolutional layers that extract features at multiple scales, ensuring that the detection process is not only fast but also extremely reliable.

Multi-Scale Feature Integration YOLOv8 incorporates a multi-scale feature integration strategy that allows it to detect objects of various sizes more accurately

Table 2.1: Comparison of CNN Architectures for Global Localization

Aspect	**GoogLeNet**	**DenseNet**	**VGG16/VGG19**
Main Idea	Inception Modules with multi-scale convolutions	Dense connections for maximal feature reuse	Deep stacks of 3×3 filters capturing fine details
Architecture	Parallel multi-scale convolutions	Feed-forward dense blocks with transition layers	Sequential 3×3 conv layers followed by fully connected layers
Strengths	Parameter-efficient multi-scale feature extraction	Mitigates vanishing gradients; feature reuse	Simplicity; strong transfer learning performance
Weaknesses	Complex branching; tuning difficulty	Higher memory consumption	Large parameter count; risk of overfitting

through backbone, model neck and model head as shown in Figure 2.7. This is particularly important for keypoint detection where keypoints can vary in scale due to perspective changes. The architecture utilizes a combination of skip connections and up-sampling techniques to merge features from different layers, ensuring rich feature maps at multiple resolutions.

Advanced Activation Functions The use of advanced activation functions such as Mish and Leaky ReLU in YOLOv8 enhances the non-linearity in the model's learning, which is crucial for distinguishing complex patterns in keypoint structures. These functions help in maintaining the flow of gradients during training, thus improving the model's ability to learn robust features.

Keypoint Detection Mechanism

YOLOv8's keypoint detection mechanism is tailored to identify at least three distinct keypoints that correspond to fixed objects in the environment. This capability is crucial for performing the necessary alignment between the robot's submap and the global reference map. The model's robustness to occlusions and its ability to handle scale variations make it uniquely suited for environments where the layout may change or be partially obscured.

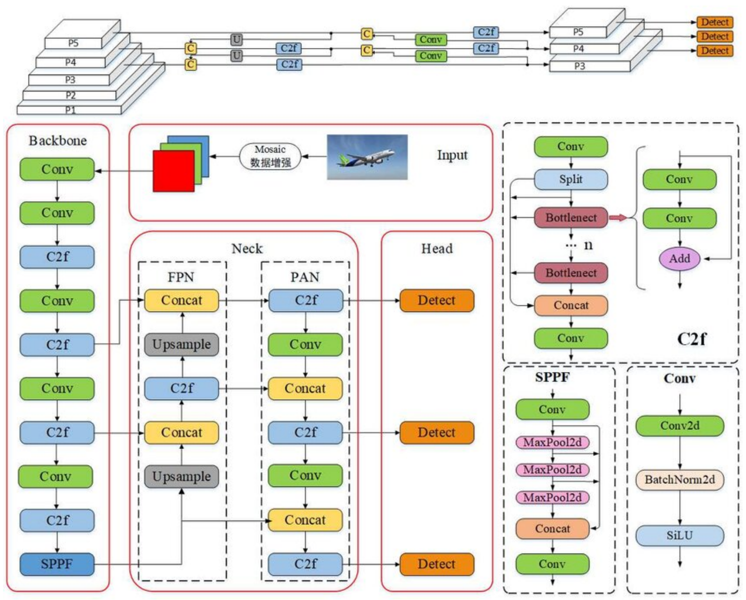


Figure 2.7: Detailed architecture of YOLOv8, showcasing its convolutional layers and feature extraction mechanisms specifically optimized for keypoint detection [13, 45]

2.3.1 Architecture Stages and Components

YOLOv8 Pose follows a single stage detection framework, meaning it predicts bounding boxes and keypoints in one forward pass. This approach is faster than traditional two-stage methods (e.g., Faster R-CNN) while maintaining high accuracy. Below is a detailed breakdown of its architecture and the stages involved explaining what is presented in the Figure 2.7 and Table 2.2:

Backbone (Feature Extraction)

The backbone is responsible for extracting features from the input image. YOLOv8 uses a CSPDarknet53 variant, which is optimized for speed and accuracy. It employs Cross-Stage Partial (CSP) connections to reduce computational complexity while maintaining feature richness, see Figure 2.7.

Neck (Feature Aggregation)

The neck module aggregates features from different scales to improve detection accuracy. As shown in the Figure 2.7 YOLOv8 uses a PANet (Path Aggregation Network) for this purpose. It merges high-level semantic features with low-level spatial details via lateral connections, ensuring both context and fine structure are preserved. PANet's bidirectional pathways enhance localization of objects and keypoints across a wide range of scales.

Head (Keypoint Prediction)

The head module is responsible for predicting keypoints and bounding boxes. YOLOv8 Pose introduces a keypoint head alongside the traditional detection head. The keypoint head outputs per-joint heatmaps and offset maps, supervised with a combination of focal and L1 losses to balance detection and pose accuracy. This dual-head design enables simultaneous object detection and precise pose estimation with minimal added complexity.

Table 2.2: Architectural Components in Each Stage

Stage	**Component**	**Functionality**
Backbone	CSPDarknet53	Extracts multi-scale features using CSP blocks and convolutional layers.
	Focus Layer	Reduces computational cost by focusing on important regions.
Neck	PANet	Aggregates features from different scales to improve detection accuracy.
	FPN	Builds a feature pyramid to handle objects of varying sizes.
Head	Detection Head	Predicts bounding boxes, objectness scores, and class probabilities.
	Keypoint Head	Predicts coordinates and confidence scores for keypoints.

In summary, this chapter has detailed the core principles and distinctive features of key CNN and detection architectures, laying the theoretical groundwork for their tailored application in 2D LiDAR–based localization addressed in the following chapters.

Chapter 3

Proposed Methodology

3.1 Framework Overview

The proposed relocalization framework is designed to address the unique challenges posed by the Kidnapped Robot Problem (KRP) in dynamic and complex indoor environments. This framework integrates both global and local localization strategies, enhancing the robustness and accuracy of autonomous mobile robots (AMRs) in establishing their position within a known map.

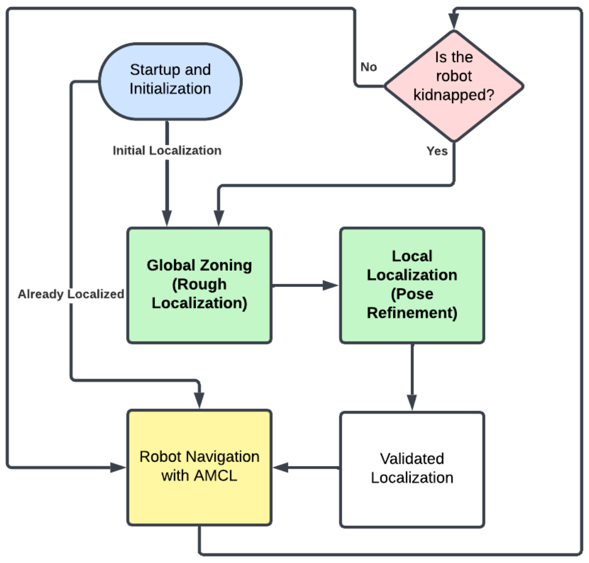

Startup and Initialization
Is the robot kidnapped?
No
Yes
Initial Localization
Global Zoning (Rough Localization)
Local Localization (Pose Refinement)
Already Localized
Robot Navigation with AMCL
Validated Localization

Figure 3.1: Proposed dual-framework workflow for relocalization.

The relocalization framework operates on a dual-layer approach:

- **Global zoning:** As the Figure 3.1 illustrates, the global zoning utilizes the advanced CNNs architectures to process 2D LiDAR data, extracting broad spatial features from occupancy grid maps. This layer aims to establish a rough alignment of the robot's position within the environment, providing a coarse estimate that narrows down the search area for more detailed analysis.

- **Local Localization:** Employs the YOLOv8 model for high-precision keypoint detection on Signed Distance Transform (SDT) maps. This stage refines the pose estimation by pinpointing specific landmarks and features that are crucial for precise navigation and task execution see Figure 3.1.

The integration of global and local localization strategies is critical for achieving seamless and efficient navigation. The global localization layer provides a macroscopic view of the environment, which is essential for initial pose estimation and rapid recovery from disorientation. In contrast, the local localization layer offers a pose refinement which is essential for detailed navigation and interaction with the environment.

Role of CNNs in Global Localization

Convolutional Neural Networks (CNNs) are pivotal in the global localization of Autonomous Mobile Robots (AMRs) using 2D Lidar data. The primary function of CNNs in this context is to extract meaningful spatial features from occupancy grid maps, which are binary or grayscale images where each pixel value represents the presence or absence of an obstacle.

Feature Extraction with CNNs

CNNs excel in identifying patterns in visual data (in our case the submaps or what we call zones), which makes them ideally suited for interpreting the complex layouts of indoor environments depicted in occupancy grid maps. Through a series of convolutional layers, each employing a set of filters, CNNs are able to capture various aspects of the spatial layout, such as edges, corners, and other geometric patterns, which are crucial for distinguishing different zones or areas within a map.

As shown in Figure 3.2, the basic architecture of a CNN consists of multiple layers designed to process and extract features from input data [32], and then passed through a dead forward neural network to make a prediction based on the falattned features extracted by CNN.

Selection of CNN Architectures

The choice of CNN architectures in this framework and experiments was driven by their ability to process large-scale spatial data efficiently. Architectures such as

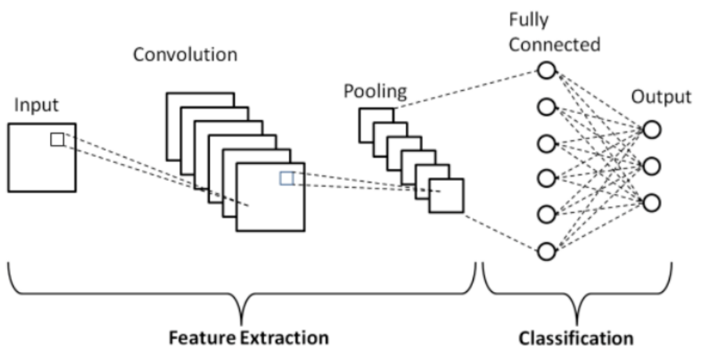


Figure 3.2: Schematic diagram of a basic convolutional neural network (CNN) architecture [32].

GoogLeNet, VGG16, VGG19, and DenseNet were selected based on their depth and complexity, which allow for a comprehensive extraction of hierarchical features. These models vary in terms of layer depth and feature abstraction capabilities:

GoogLeNet introduces inception modules that process data at various scales, capturing fine to coarse spatial features. While VGG16 and VGG19 offer deep architectures with repeated convolutional layers, making them robust at capturing more detailed spatial patterns. In other hand DenseNet [17] leverages feature reuse through densely connected layers, enhancing feature propagation and reducing the risk of overfitting. Finally, each architecture contributes uniquely to the robustness and accuracy of global localization, facilitating reliable initial pose estimation in diverse indoor settings.

Global localization was approached as an image-based classification problem: given a single submap image, the task is to predict the zone or location of the robot. We evaluated four CNN architectures as the localization model: GoogLeNet, VGG16 [37], and DenseNet. The standard architectures were largely unmodified; we added only a dropout layer in the final classification stage and applied L2 regularization (weight decay) of 1×10^{-3}. Each network was trained to classify input images into one of the predefined zones. Training was conducted using Stochastic Gradient Descent (SGD) with carefully tuned hyperparameters (learning rate, momentum, weight decay) to ensure stable convergence, as the models were at risk of overfitting. Hence, regularization techniques such as dropout, L2 norm, and data augmentation were employed via rotation on the training set only, especially for the larger VGG networks to reduce overfitting; in addition, batch normalization was used to mitigate internal covariance shift.

Role of YOLOv8 in Local Localization

The YOLOv8 model plays a crucial role in the local localization phase, where precision is paramount. Local localization involves refining the robot's pose estimation to a high degree of accuracy, crucial for detailed navigation and interaction within the environment.

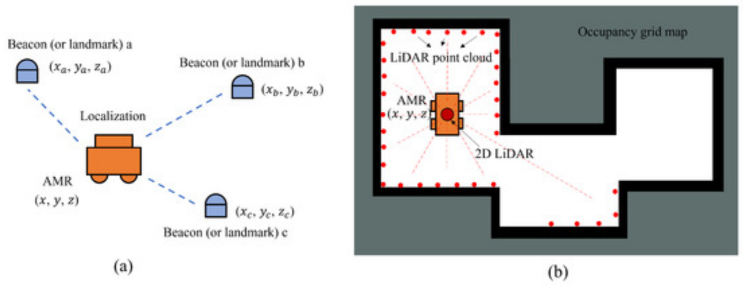


Figure 3.3: Role of keypoints detection for pose alignment [25].

Precision in Local Localization

Precision in local localization is critical for tasks requiring high fidelity spatial awareness, such as obstacle avoidance, path planning, and targeted operations within tight spaces. YOLOv8, with its advanced object detection capabilities, is adept at identifying and precisely locating keypoints from Signed Distance Transform (SDT) maps. These keypoints represent specific features within the environment, such as doorways, junctions, or designated areas of interest as shown in Figure 3.3. These enhancements ensure that YOLOv8 provides reliable and precise local localization, critical for the effective navigation and operation of AMRs in complex indoor settings.

3.2 Dataset Preparation

The effectiveness of the proposed localization framework heavily relies on the quality and preparation of the dataset used for training the convolutional neural networks and YOLOv8 model. This section describes the specific characteristics of the 2D LiDAR data used, the protocols followed for data collection, and the preprocessing and augmentation techniques implemented to ensure robustness and accuracy in localization.

Characteristics of Indoor 2D LiDAR Data

Two-dimensional (2D) Light Detection and Ranging (LiDAR) sensors form the core of our indoor localization dataset by emitting laser beams and timing their reflections to produce high-resolution measurements, full 360° coverage, and immunity to ambient lighting variations. However, they also present inherent challenges: since data are captured in a single horizontal plane, objects above or below that plane may be missed; shiny or specular surfaces can distort range readings; and open areas with few nearby obstacles yield sparse scan points that complicate interpretation. Additionally, unpredictable magnetic disturbances can introduce scan errors that no current simulator can faithfully reproduce.

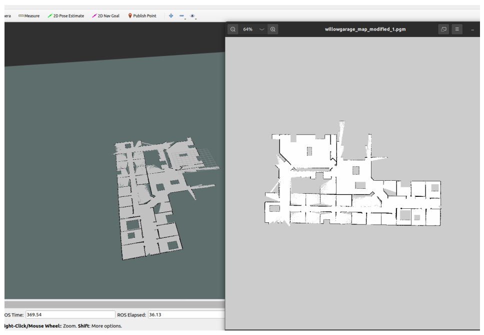
Data Collection and Preprocessing

Data for the experiments was collected in the Gazebo simulator within the Willow Garage environment using ROS1 Noetic. A TurtleBot3 Burger equipped with a 2D LiDAR sensor navigated through various simulated indoor scenes, recording raw distance measurements and odometry data to build discrete, separated submaps an approach that mitigates continuous odometry drift. Preprocessing of the LiDAR scans involved applying median and Gaussian filters to smooth out sensor noise, and converting the filtered scans into occupancy grid maps (occupied, free, or unknown cells) via ROS1 Noetic to support accurate model training. Where the inputs occupancy grid maps have been resized to 224×224 and normalized according to standard ImageNet RGB means and variances, with random jittering [39] added applied online during training. In addition to raw accuracy, we examined the models' confusion behavior between zones.

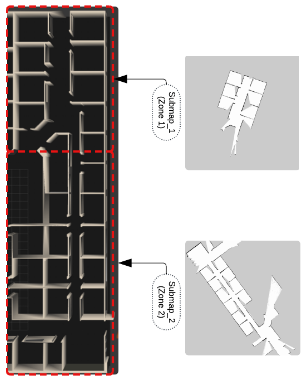
A dataset of 1,200 labeled images was used and collected from the simulation, covering two zones under various viewpoints. The dataset was randomly split ensuring no leakage between training, validation, and test sets into 60% for training, 25% for validation, and 15% for testing. We began with a set of base zone samples and manually added distortions and occlusions (random rotations, translations, and occlusions) to generate diverse scenarios. To further improve the generalization of the CNN models, we applied random jittering and rotation augmentations to the training images only. Each image in the dataset is labeled with its ground-truth zone (for global zoning) and annotated with keypoint locations of interest (for local localization refinement) after transforming it to SDT version.

3.2.1 Scenario Generation Techniques

Scenario generation and rotation based augmentation play a crucial role in improving the generalization capability of the proposed dual framework, by extending the dataset to cover more real world scenarios.



(a) Global occupancy grid map visualized in RViz.



(b) Simulated indoor test environment used for evaluation from Gazebo

Figure 3.4: Simulated indoor test environment used for evaluating the localization system. The environment is divided into distinct zones (e.g., rooms or areas) for the global localization task.

1. **Geometric Transformations:** Including rotations, and translations to simulate different perspectives and scales that the robot might encounter.
2. **Simulated Occlusions:** Artificially adding random sized obstacles in the occupancy grid maps to teach the models to handle blocked sensors or unexpected objects,
3. **Noise Injection:** Introducing random noise to the data mimics real-world environmental noise and sensor errors, further robustifying the models.

3.3 Problem Formulation

State Vector Representation

The state vector encapsulates the robot's position, orientation, and extracted features from 2D LiDAR scans, defined as:

$$s = [x, y, \theta, f_1, f_2, \ldots, f_n] \tag{3.1}$$

where: x, y, θ represent the robot's 2D position and orientation, and $f_1, f_2, \ldots, f_n$ are the features extracted from 2D LiDAR-based submaps.

Optimization Objective: Cost Functions

The goal is to minimize localization errors while ensuring reliable relocalization. The optimization is guided by the following cost functions:

Submap Classification Cost Function

This cost function classifies the current submap into a known zone using a CNN and categorical cross-entropy loss:

$$L_{\text{submap}} = -\sum_{i=1}^{C} y_i \log(\hat{y}_i) \tag{3.2}$$

where C is the number of submaps (zones) classes, y_i is the one-hot ground truth label, and $\hat{y}_i$ is the softmax probability for class i.

Keypoint Detection Cost Function

This cost helps refine pose by regressing key landmark coordinates from the SDT map:

$$L_{\text{KP}} = \frac{1}{N} \sum_{i=1}^{N} \left\| \hat{\mathbf{k}}_i - \mathbf{k}_i \right\|^2 \tag{3.3}$$

where: N is the number of keypoints, $\hat{\mathbf{k}}_i$ is the predicted keypoint, and $\mathbf{k}_i$ is the ground truth keypoint.

YOLOv8-Pose Objective Function

YOLO (You Only Look Once) is a family of real-time object detection models. YOLOv8 is the latest and most advanced version, integrating improvements in speed and accuracy. YOLOv8-Pose extends this architecture to also estimate the object keypoints using a single-stage detector that jointly predicts bounding boxes, class labels, and keypoint coordinates in a single forward pass [45].

$$L_{\text{YOLOv8}} = \lambda_{\text{box}} L_{\text{box}} + \lambda_{\text{cls}} L_{\text{cls}} + \lambda_{\text{pose}} L_{\text{pose}} \tag{3.4}$$

where: L_{box} is the Complete Intersection over Union (CIoU) loss for bounding box regression, L_{cls} is the Binary Cross-Entropy (BCE) loss for classification, L_{pose} is the Mean Squared Error (MSE) loss over keypoints, and $\lambda_{\text{box}}, \lambda_{\text{cls}}, \lambda_{\text{pose}}$ are the respective loss weights.

.

This combined objective allows YOLOv8-Pose to simultaneously detect objects and estimate their associated keypoints in a single forward pass, enabling real-time localization.

Alignment Cost Function

This cost aligns the current submap with the global map using weighted error minimization:

$$L_{\text{align}} = \frac{1}{N} \sum_{i=1}^{N} w_i \, (\hat{x}_i - x_i)^2 \tag{3.5}$$

where: w_i is the weight of each particle (likelihood of correct pose), $\hat{x}_i$ is the estimated pose, and x_i is the ground truth pose.

To summarize, this chapter has established the theoretical foundations of both classical probabilistic filters and modern CNN- and YOLO-based approaches detailing their principles, strengths, and limitations—and has outlined the dataset preparation and dual-layer framework that will underpin the subsequent implementation and evaluation of our 2D LiDAR–based relocalization system.

Chapter 4

Experimental Results and Evaluation

4.1 Introduction

Localization is a fundamental capability for autonomous robots, essential for navigation, path planning, and overall autonomy. A particularly challenging scenario is the initial localization or the KRP where a robot is unexpectedly relocated without prior knowledge of its new position. Solving this requires global localization to determine the robot's general location within a map and local localization to refine its exact pose.

This chapter evaluates deep learning based localization methods for addressing this problem. We assess global localization using CNN models (GoogLeNet, VGG16, VGG19, and DenseNet) trained on grid maps and local pose refinement using YOLOv8-Pose on SDT-transformed maps. The performance of these models is compared against traditional approaches such as Monte Carlo Localization (MCL) and feature-based methods (SIFT and ORB). We analyze their accuracy, robustness, and efficiency while discussing the advantages of deep learning in handling occlusions, environmental changes. Finally, we outline the system's limitations and suggest directions for future improvements.

4.2 Results and Disscussion

4.2.1 Global Localization Performance

Table 4.2 summarizes the performance metrics for each CNN model on the global localization task. All models achieved high accuracy, with some differences in generalization performance:

As shown in Table 4.2, the application of data augmentation had a significant impact on improving the generalization capability of all models. GoogLeNet

Metric	**Equation**	**Description**
Average Precision (AP)	$AP = \int_0^1 p(r)\,dr$ where: - $p(r)$: Precision as a function of recall. Variants: AP50 (OKS threshold 0.5), AP75 (OKS threshold 0.75), AP@[.5:.95] (average over OKS thresholds from 0.5 to 0.95 with a step of 0.05).	Calculates the average precision over different recall values by integrating the precision-recall curve. Specifically for pose estimation, AP is computed based on OKS thresholds (similar to IoU thresholds in object detection).
Mean Average Precision (mAP)	$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i$ where: - N: Number of classes or OKS thresholds - AP_i: Average Precision for class i or at OKS threshold i	Computes the average of AP values across different classes or OKS thresholds. For YOLOv8-Pose, mAP@[.5:.95] is often the primary metric, averaging AP values over OKS thresholds from 0.5 to 0.95.
Precision	$Precision = \frac{TP}{TP+FP}$	The proportion of correctly identified keypoints out of all predicted keypoints. TP = True Positives, FP = False Positives. A higher precision indicates fewer false keypoint detections.
Recall	$Recall = \frac{TP}{TP+FN}$	The proportion of correctly identified keypoints out of all actual ground-truth keypoints. TP = True Positives, FN = False Negatives. A higher recall indicates fewer missed ground-truth keypoints.

Table 4.1: Keypoint Detection Metrics for YOLOv8-Pose

achieved a validation accuracy of 97.48% and a test accuracy of 95.24% with augmentation, whereas without augmentation, its test accuracy dropped drastically to 35.62%, demonstrating an improvement ratio of 2.67×. Similarly, VGG16 and VGG19 reached 100% test accuracy when trained with augmented data but only achieved 33.45% and 37.12%, respectively, without augmentation, highlighting their susceptibility to overfitting in limited datasets. The largest improvement ratio was observed in VGG16 (2.99×), indicating that augmentation was particularly beneficial for this model. DenseNet, which reached 100% accuracy across training, validation, and test sets with augmentation, exhibited a significant performance drop to 34.18% without augmentation, resulting in an improvement ratio of 2.79×. These results confirm that while DenseNet demonstrates superior generalization due to

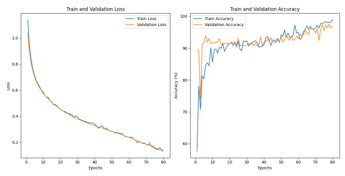
Table 4.2: Performance Metrics for Global Localization with and without Data Augmentation

Model	**Validation Acc.**	**Test Acc. (Scenario Gen.)**	**Test Acc. (No Scenario Gen.)**	**Improv. Ratio**
GoogLeNet	97.48%	95.24%	35.62%	2.67x
VGG16	99.50%	100.00%	33.45%	2.99x
VGG19	99.16%	100.00%	37.12%	2.69x
DenseNet	100.00%	100.00%	34.18%	2.79x

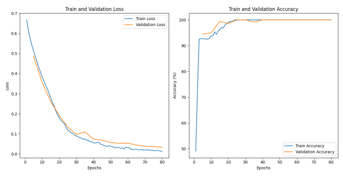
its densely connected architecture that enhances feature reuse and gradient flow, it still relies on data augmentation to avoid overfitting. The observed discrepancies in test accuracy between augmented and non-augmented training indicate that CNN-based localization models require diverse training examples to learn robust spatial features from occupancy grids. Without augmentation, models were prone to memorizing training data rather than learning transferable features, leading to poor generalization on unseen test maps. The convergence behavior of all models further supports this finding, as training stability and generalization improved with augmentation, allowing the models to achieve high accuracy within fewer epochs. The performance gap between augmented and non-augmented training highlights the importance of incorporating transformations such as rotations, scaling, and noise perturbations when training CNN-based localization models on occupancy grid maps. These findings emphasize that, for robust global localization in kidnapped robot scenarios, leveraging augmentation strategies is crucial to ensure the network learns invariant spatial representations rather than overfitting to specific training examples.

From a learning and convergence standpoint, all models were observed to converge quickly within a few epochs given the relatively small and well-structured dataset. The GoogLeNet and DenseNet models, in particular, trained faster and required fewer epochs to reach high accuracy, whereas VGG16 and VGG19, with their larger number of parameters, needed more epochs and careful tuning of learning rate schedules. Nonetheless, by the end of training, all CNN models provided excellent global localization performance on the simulated initial submaps after the kidnapped robot scenarios. Figure 4.1 illustrates example training and validation accuracy curves for two of the models (placeholder), demonstrating stable training progress and only minimal gaps between training and validation accuracy. This is indicative of good generalization, which is illustrated in Figure 4.2.

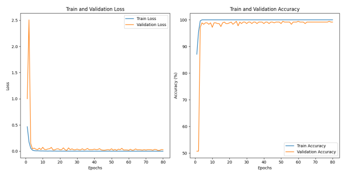
In addition to raw accuracy, we examined the models' confusion behavior between zones. All CNNs exhibited confident predictions, rarely confusing one zone for another. GoogLeNet, being slightly less complex, showed a few misclassifications typically between visually similar adjacent zones. The VGG and DenseNet



(a) GoogLeNet accuracy and losses



(b) DenseNet accuracy and losses



(c) VGGNet accuracy and losses

Figure 4.1: Training and validation accuracy and loss curves for GoogLeNet, DenseNet, and VGGNet on the global localization task.

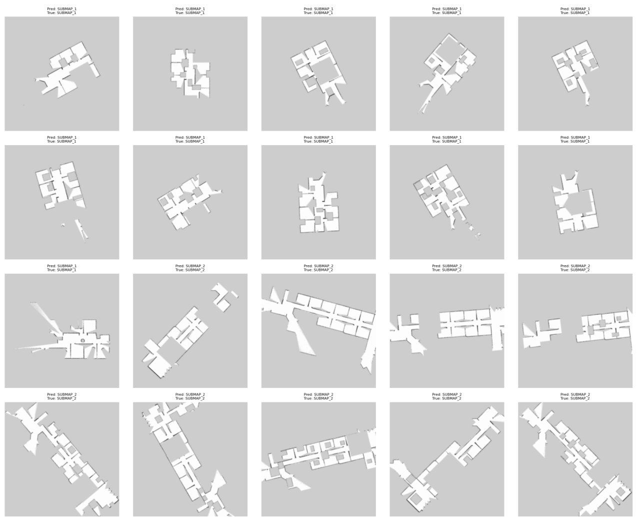


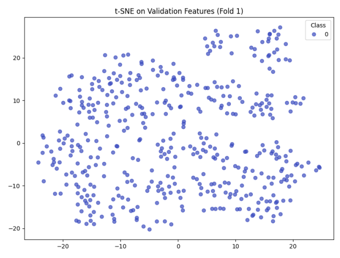
Figure 4.2: Global Zoning Results

models, on the other hand, had virtually zero confusion on the test set, correctly identifying every zone even when images had moderate occlusion or lighting variation. For example, DenseNet successfully recognized a room even when some of its distinctive features were occluded by an object, leveraging other contextual cues in the scene. These results demonstrate that CNN-based global localization can be remarkably accurate for the kidnapped robot problem, effectively solving the global re-localization in a single image inference.

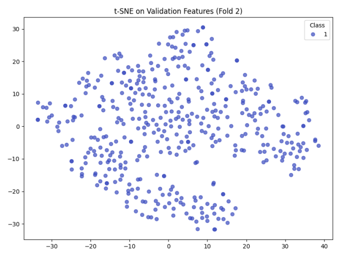
4.2.2 Limitation of the Global Zone Classification

To further inspect the model's ability to learn spatial and semantic features unique to each zone, we performed a leave-one-zone-out validation. Specifically, we trained the model on one zone while validating on (1) the entirety of the other zone and (2) a 10% hold-out from the training zone and vice versa. The model failed to distinguish between zones, overfitting to non-discriminative cues (e.g., color) instead to general. To solve this issue, we may need attention-based learning, using the SDT version of the grid maps, using transfer learning, advanced hyperparameters tuning, and relying on universal deep learning models that are used in medical imaging to learn spatial features may also improve the generalization ability. All this can be addressed in further work.

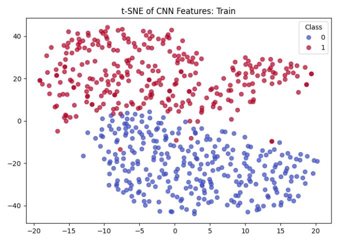
Figures 4.3a and 4.3b show the feature embeddings when training exclusively on Zone 1 and Zone 2, respectively. In both cases, there is a large overlap between the two classes in feature space, which explains the poor discrimination. In contrast, Figure 4.3c illustrates the clear separation of embeddings when the model is trained on samples from both zones simultaneously.



(a) Features learned from Zone 1 only



(b) Features learned from Zone 2 only



(c) Features learned from both zones

Figure 4.3: t-SNE visualization of feature embeddings under (a) leave-one-zone-out on Zone 1, (b) leave-one-zone-out on Zone 2, and (c) joint training on both zones.

4.2.3 Local Localization and Keypoint Detection Performance

After determining the coarse location (zone) of the robot via the global localization stage, the next step is to refine the robot's exact pose within that zone. This **local localization** stage was implemented using a keypoint detection approach. We employed **YOLOv8-Pose**, a state-of-the-art one-stage object detection model extended to predict keypoints, to detect specific visual landmarks or features in the robot's environment. Instead of camera images, we used transformed grid maps derived from LiDAR point clouds. These maps were converted into binary occupancy images and then processed using the Signed Distance Transform (SDT) to highlight structural features. The keypoints detected from SDT maps include structural elements such as room corners, door edges, and corridor intersections, which serve as reliable localization landmarks. By identifying correspondences between detected keypoints in the SDT maps and their reference positions in the environment, the robot's precise pose can be computed. YOLOv8-Pose was trained on a dataset of SDT-transformed occupancy maps, where each map had several annotated keypoints relevant for localization.

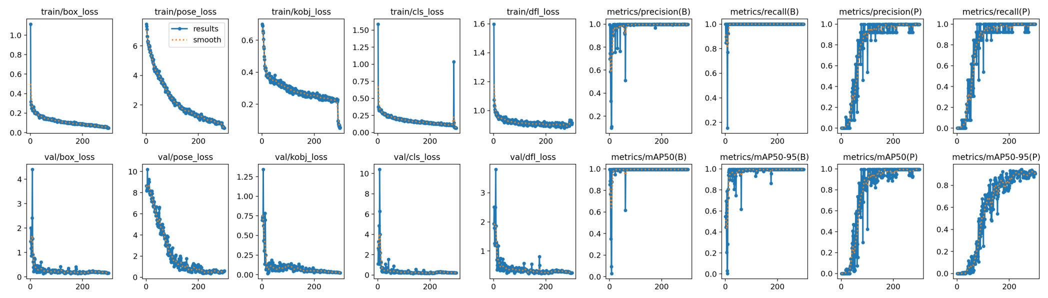
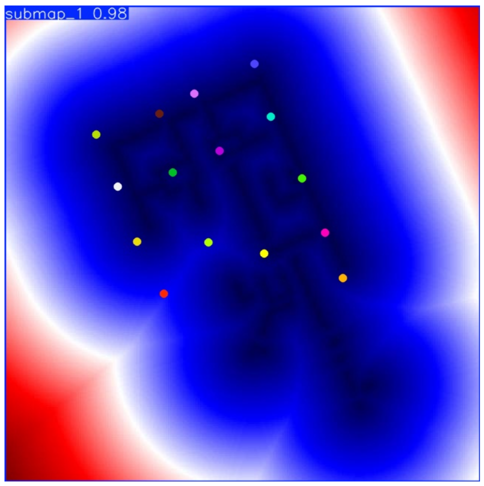


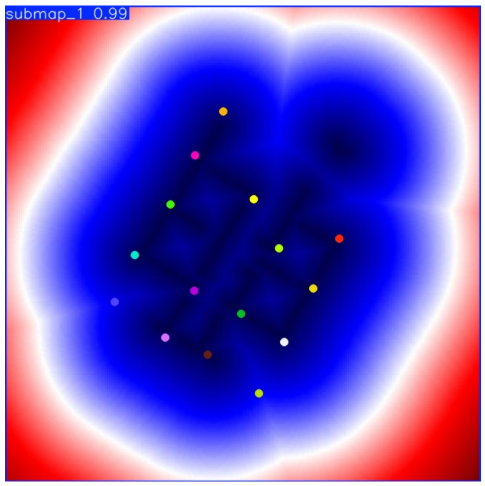
Figure 4.4: Training Curves Yolov8-pose

Table 4.3: Performance Metrics for YOLOv8-Pose Model

Precision (Box)	**Recall (Box)**	**mAP@50 (Box)**	**mAP@50-95 (Box)**	**Precision (Pose)**	**Recall (Pose)**	**mAP@50 (Pose)**	**mAP@50-95 (Pose)**
0.996	1.000	0.995	0.995	0.996	1.000	0.995	0.942



(a) YOLOv8-Pose keypoint detection on SDT-transformed occupancy map (Test Case 1).



(b) YOLOv8-Pose keypoint detection on SDT-transformed occupancy map (Test Case 2).

Figure 4.5: Comparison of YOLOv8-Pose keypoint detection performance on two different SDT-transformed test cases. The model successfully detects key structural landmarks, demonstrating robustness to occlusion and missing regions.

The YOLOv8-Pose model demonstrated excellent performance in detecting these keypoints. It achieved a precision of 99.64% on the test set as shown in the Table 4.3 as we trained only one class for the keypoint detection purpose (Zone 1). In other words, virtually every keypoint present in the SDT map was detected, and almost all detections were correct with negligible false positives precision $\approx$ 99.6%. This high precision and recall indicate the reliability of the local localization component: the model rarely misses a useful landmark and gives very few incorrect detections. This reliability is crucial because false detections could lead to incorrect pose estimates, and missed detections could leave the robot uncertain about its precise location. Figure 4.5 illustrates an example output from YOLOv8-Pose on a test SDT map, showing how the model identifies keypoints (highlighted by markers) even when part of the map is occluded.

Beyond precision metric, the YOLOv8-Pose model was tested under changed environment conditions, such as missing keypoints and artificial noise in the SDT maps as shown in Figure 4.5a and Figure 4.5b. Despite these alterations, the model consistently detected alternative structural keypoints, ensuring successful local localization. Its joint detection and keypoint regression architecture contributed to its robustness, enabling accurate pose refinement even in modified environments. The detected keypoints were then matched with a reference SDT map to compute the robot's precise pose. Due to the model's high accuracy, the resulting positional and orientation errors were minimal. Once the CNN-based global localization identified the correct zone, YOLOv8-Pose effectively refined the pose, reliably addressing the kidnapped robot problem.

4.3 Comparison with Traditional Methods

To contextualize our approach, we compare it with classical localization methods, particularly Monte Carlo Localization (MCL) and feature-based approaches like SIFT and ORB.

We calculated a set of zone-level occupancy grid maps directly from Gazebo simulations by placing random static objects in a known indoor environment. By "dynamic objects," we refer to movable objects that were repositioned between simulation runs but assumed to be static during data capture. These types of objects can typically be filtered out in practice using techniques like Kalman filtering, which allows for tracking and discarding transient or moving elements in the environment.

- **Monte Carlo Localization (MCL):** MCL is a particle filter-based localization approach that estimates the robot's pose by maintaining and updating a distribution of particles according to a motion model and sensor model [28]. However, MCL critically depends on accurate odometry and a good initial distribution of particles [12]. In the kidnapped robot scenario, where the robot is suddenly moved without any odometric trail, MCL often fails to converge efficiently. It either requires a very large number of uniformly distributed particles or extensive resampling to localize correctly, which leads to increased computation time and often delayed or failed recovery. In contrast, our CNN-based global localization approach provides a direct and immediate location estimate from the observation, bypassing the need for particle-based tracking and significantly reducing localization delay.
- **Feature-based Localization (SIFT/ORB):** Classical feature-based methods such as SIFT and ORB operate by extracting keypoints from occupancy or edge-based maps and matching them against a reference database of known locations. While ORB offers better robustness to rotation and scale changes, it is computationally intensive, limiting its use in real-time systems. SIFT, on the other hand, is faster but less discriminative, especially in repetitive or occluded environments. Although both methods sometimes predict the correct zone based on the majority of keypoint correspondences, a closer inspection (see Figure 4.6) reveals that many of these matches are imprecise or incorrect only a small subset of keypoints yield accurate spatial correspondences. Furthermore, when parts of the scene are occluded or altered (e.g., a moved object), these traditional descriptors fail to localize the robot accurately. This is evident in cases where SIFT or ORB cannot locate the correct keypoints in

direct collected data from Gazebo simulator, leading to misclassification as shown in Table 4.4.

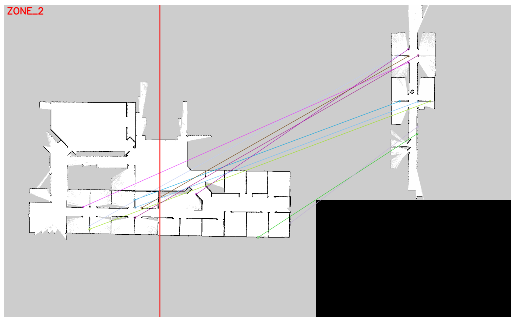
Our proposed deep learning-based method, even though it may not explicitly detect fine-grained geometric correspondences like traditional keypoint-based methods, rarely makes incorrect predictions. It achieves this by learning robust, high-level semantic representations from Signed Distance Transform (SDT) maps that are invariant to occlusion, viewpoint changes, and minor environmental variations. As demonstrated in Figure 4.6, our model consistently outperforms both MCL and feature-based approaches in terms of accuracy and robustness in the context of the kidnapped robot problem.

Table 4.4: Classification Performance across Two Zones (Combined)

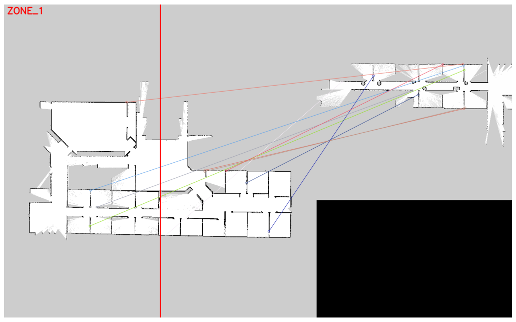
Metric	**SIFT (Zone1+2)**	**ORB (Zone1+2)**	**Ours (GoogLeNet)**
Precision	0.55	0.80	0.975
Recall	0.55	0.80	0.975
F1 Score	0.55	0.80	0.975

Resilience to Occlusion and Environmental Variations

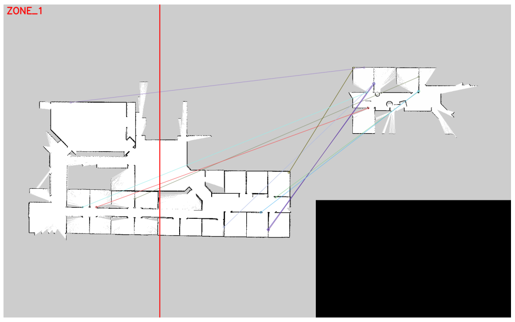
Although our global localization module, based on CNN classification of occupancy grids, does not explicitly learn spatial relationships or topological layout, it is still capable of reliably identifying the correct zone. This suggests that the model is learning high-level structural features or distributional patterns that are sufficiently discriminative across zones. Moreover, the local refinement stage using YOLOv8-Pose proves highly robust to partial occlusions and missing structural elements in the Signed Distance Transform (SDT) maps. Even when key segments are occluded or altered, the pose estimation remains accurate due to YOLOv8's ability to generalize and detect available spatial cues. As demonstrated in Figure 4.7a, the model can infer and localize relevant keypoints despite significant occlusions, outperforming traditional feature-based approaches that fail in the absence of detectable keypoints. This robustness is critical for real-world scenarios where environmental dynamics or sensor noise may corrupt portions of the input data. Our integrated system thus ensures fast, reliable localization even in the presence of visual inconsistencies or structural changes. Accurate keypoint detection greatly enhances the transformation matrix used for pose alignment. YOLOv8-Pose reliably identifies semantic landmarks (e.g., corners, object edges), leading to more precise alignment. As shown in Table 4.5, YOLOv8-Pose achieves a much lower RMSE (0.376) compared to ICP (3.719). ICP's higher error stems from its sensitivity to noise, initial pose, and occlusions, while YOLOv8-Pose leverages learned features for more robust and accurate localization.



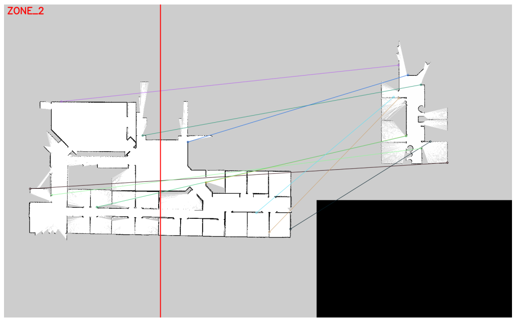
(a) ORB - Zone (Correct)



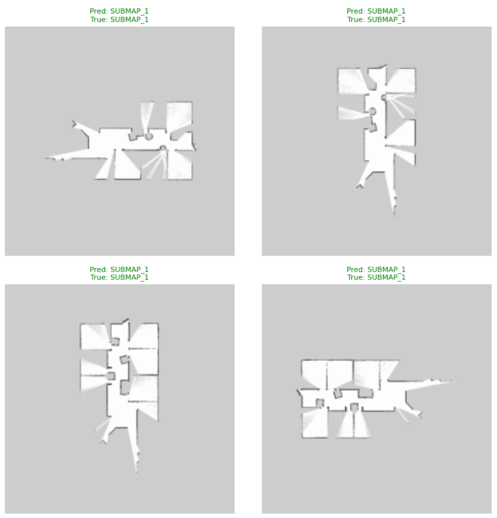
(b) ORB - Zone (Wrong)



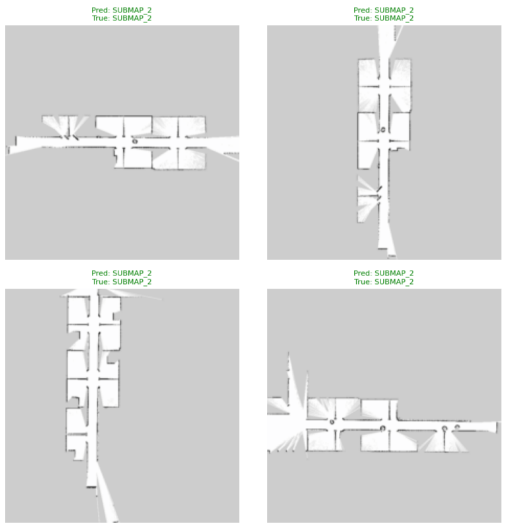
(c) SIFT - Zone 1 (Correct)



(d) SIFT - Zone (Wrong)

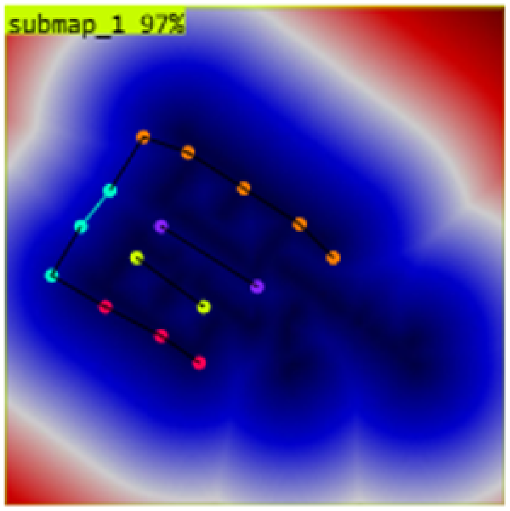


(e) Ours - Zone Prediction

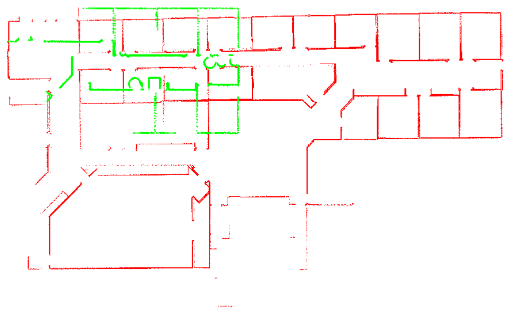


(f) Ours - Zone Prediction

Figure 4.6: Visual comparison of ORB, SIFT, and our GoogLeNet-based approach in zone classification. Correct and incorrect predictions are shown for traditional methods, while our model provides consistent results.



(a) YOLOv8 Pose refinement result



(b) ICP Alignment Result

Figure 4.7: Comparison between YOLOv8 pose-based prediction and traditional ICP alignment for local pose refinement.

Table 4.5: Comparison of Average RMSE for Alignment between YOLOv8-Pose and ICP

Method	**Average RMSE**
ICP (Iterative Closest Point)	3.719
YOLOv8-Pose Based Alignment	0.376

4.4 Conclusion

In this chapter, we experimentally validated the proposed deep learning based localization framework for addressing the kidnapped robot problem. The two-stage system CNN-based global zoning and YOLOv8-Pose-based local pose refinement achieved near-perfect global classification accuracy and highly precise pose estimation, even under challenging conditions like occlusions and dynamic changes. Compared to traditional methods such as MCL and feature-matching techniques (SIFT, ORB), our approach demonstrated faster, more reliable relocalization without the need for prior pose knowledge or long robot movement. These results confirm that 2D LiDAR based deep learning methods can effectively solve the initial localization and relocalization problem, paving the way for real-world deployment

Chapter 5

Conclusion and Future Work

This thesis introduced a deep learning-based localization framework designed to address both initial global localization and relocalization following kidnapping events in mobile robots. The proposed two stage pipeline first performs global zone level classification using convolutional neural networks (CNNs) on occupancy grid submaps derived from 2D LiDAR data, and second, refines the robot's pose locally using YOLOv8-Pose-based semantic keypoint detection. By eliminating the need for odometry or prior pose estimates, the system achieves accurate and robust localization recovery with minimal movement and no external aid.

Experimental results in simulated environments demonstrated that the system achieves high zone classification accuracy and sub-centimeter pose alignment, even under occlusions, dynamic obstacles, and map changes. Compared to traditional methods like MCL, AMCL, ICP, and classical feature-matching algorithms (e.g., SIFT, ORB), the framework showed greater robustness, speed, and accuracy while using only low cost 2D LiDAR data. However, challenges remain. The global zoning stage can overfit to non-distinctive spatial patterns, leading to misclassifications in complex layouts. Future work will explore attention-based learning, transfer learning, and hybrid models to improve feature discrimination and generalization. Additionally, as the number of zones increases, classification performance may degrade due to spatial similarity, necessitating scalability testing in real-world scenarios.

Another limitation is the current reliance on simulated training data. To support real-world deployment, the framework requires a significantly larger and more diverse dataset. Future research will address this through domain adaptation and continuous learning to handle environmental changes over time. A critical next step is deploying the system as a ROS 2-compatible service, enabling seamless integration into robotic platforms and real-time operation. Overall, the proposed framework offers a scalable, cost effective, and modular solution to the Kidnapped Robot Problem, with strong potential for deployment in dynamic indoor environments such as warehouses and office spaces.

Bibliography

[1] A. Abedini, M. Hahn, and F. Samadzadegan, *An investigation into the registration of lidar intensity data and aerial images using the sift approach*, Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., XXXVII (2008), pp. 169–174. Available at: https://www.isprs.org/proceedings/xxxvii/congress/1_pdf/29.pdf.

[2] J. M. Beer, A. D. Fisk, and W. A. Rogers, *Toward a framework for levels of robot autonomy in human–robot interaction*, J. Human-Robot Interact., 3 (2014), pp. 74–99. DOI: https://doi.org/10.5898/JHRI.3.2.Beer.

[3] P. J. Besl and N. D. McKay, *A method for registration of 3-d shapes*, IEEE Trans. Pattern Anal. Mach. Intell., 14 (1992), pp. 239–256. DOI: https://doi.org/10.1109/34.121791.

[4] I. Bukhori and Z. H. Ismail, *Detection of kidnapped robot problem in monte carlo localization based on the natural displacement of the robot*, Int. J. Adv. Rob. Systems, 14 (2017), pp. 1–6. DOI: https://doi.org/10.1177/1729881417717469.

[5] W. Burgard, D. Fox, and S. Thrun, *Active mobile robot localization*, in Proc. Second EUROMICRO Workshop on Advanced Mobile Robots, Brescia, Italy, 1997, pp. 1346–1352. DOI: https://doi.org/10.1109/EURBOT.1997.633623.

[6] J. D. Camacho, C. Villaseñor, C. Lopez-Franco, and N. Arana-Daniel, *Neuroplasticity-based pruning method for deep convolutional neural networks*, Appl. Sci., 12 (2022), p. 4945. DOI: https://doi.org/10.3390/app12104945.

[7] H.-H. Chen and C.-H. G. Li, *Atopnet: Robust visual localization for amr navigation*, in Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics (AIM), 2022, pp. 275–281. DOI: https://doi.org/10.1109/AIM52237.2022.9863347.

[8] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, *A solution to the simultaneous localization and map building (slam) problem*, IEEE Trans. Robot. Autom., 17 (2001), pp. 229–241. DOI: https://doi.org/10.1109/70.938381.

[9] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, *3-d mapping with an rgb-d camera*, IEEE Trans. Robot., 30 (2014), pp. 177–187. DOI: https://doi.org/10.1109/TRO.2013.2282218.

[10] X. Feng, H. Yao, and S. Zhang, *An efficient way to refine densenet*, Signal Image Video Process., 13 (2019), pp. 959–965. DOI: https://doi.org/10.1007/s11760-019-01433-4.

[11] D. Fox, *Adapting the sample size in particle filters through kld-sampling*, Int. J. Robot. Res., 22 (2003), pp. 985–1003. DOI: https://doi.org/10.1177/0278364903022012001.

[12] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, *Monte carlo localization: Efficient position estimation for mobile robots*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 16, Orlando, FL, 1999, AAAI Press, pp. 343–349. Available at: https://www.aaai.org/Papers/AAAI/1999/AAAI99-050.pdf.

[13] R. Gao, M. Chen, and Z. Zhao, *Real-time Detection Algorithm of Aircraft Landing Gear based on improved YOLOv8*. Research Square preprint, 2024.

[14] A. Gil, Ó. Reinoso, M. A. Vicente, C. Fernández, and L. Payá, *Monte carlo localization using sift features*, in Proc. Iberian Conf. Pattern Recognition and Image Analysis (IbPRIA), vol. 3522 of Lecture Notes in Computer Science, Springer, 2005, pp. 623–630. DOI: https://doi.org/10.1007/11492429_75.

[15] G. Grisetti, C. Stachniss, and W. Burgard, *Improved techniques for grid mapping with rao-blackwellized particle filters*, IEEE Trans. Robot., 23 (2007), pp. 34–46. DOI: https://doi.org/10.1109/TRO.2007.889486.

[16] M. Hentschel and B. Wagner, *An adaptive memory model for long-term navigation of autonomous mobile robots*, J. Robotics, 2011 (2011), p. 506245. DOI: https://doi.org/10.1155/2011/506245.

[17] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, *Densely connected convolutional networks*, in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4700–4708. DOI: https://doi.org/10.1109/CVPR.2017.243.

[18] N. S. Johnson, P. S. Vulimiri, A. C. To, X. Zhang, C. A. Brice, B. B. Kappes, and A. P. Stebner, *Machine learning for materials developments in metals additive manufacturing*, Addit. Manuf., 36 (2020). DOI: https://doi.org/10.1016/j.addma.2020.101641.

[19] S. J. Julier and J. K. Uhlmann, *New extension of the kalman filter to nonlinear systems*, in Proc. SPIE Signal Process., Sensor Fusion, and Target Recognition VI, vol. 3068, 1997, pp. 182–193. DOI: https://doi.org/10.1117/12.280797.

[20] R. E. Kalman, *A new approach to linear filtering and prediction problems*, J. Basic Eng., 82 (1960), pp. 35–45. DOI: `https://doi.org/10.1115/1.3662552`.

[21] P. Kardas, F. Bielec, M. Brauncajs, P. Lewek, D. Timler, E. Łojewska, M. Chiurazzi, N. N. Dei, G. Ciuti, R. J. Ros, V. S. Estevan, A. Maccaro, L. Pecchia, B. Merino, A. Medrano, T. Penzel, and G. Fico, *Evaluation of disinfection methods for autonomous mobile robots used in hospital logistics in emergency departments*, J. Hosp. Infect., (2025). In Press. DOI: `https://doi.org/10.1016/j.jhin.2025.05.006`.

[22] H. Kuang, X. Chen, T. Guadagnino, N. Zimmerman, J. Behley, and C. Stachniss, *Ir-mcl: Implicit representation-based online global localization*, IEEE Robot. Autom. Lett., 8 (2023), pp. 1627–1634. DOI: `https://doi.org/10.1109/LRA.2023.3240929`.

[23] J. Levinson and S. Thrun, *Robust vehicle localization in urban environments using probabilistic maps*, in Proc. IEEE Int. Conf. Robotics and Automation (ICRA), 2010, pp. 4372–4378. DOI: `https://doi.org/10.1109/ROBOT.2010.5509336`.

[24] P. Li, R. Wang, Y. Wang, and W. Tao, *Evaluation of the icp algorithm in 3d point cloud registration*, IEEE Access, 8 (2020), pp. 68030–68048. DOI: `https://doi.org/10.1109/ACCESS.2020.2985584`.

[25] Y. Liu, S. Wang, Y. Xie, T. Xiong, and M. Wu, *A review of sensing technologies for indoor autonomous mobile robots*, Sensors, 24 (2024), p. 1222. DOI: `https://doi.org/10.3390/s24041222`.

[26] D. G. Lowe, *Distinctive image features from scale-invariant keypoints*, Int. J. Comput. Vision, 60 (2004), pp. 91–110. DOI: `https://doi.org/10.1023/B:VISI.0000029664.99615.94`.

[27] MathWorks Inc., *Developing Autonomous Mobile Robots Using MATLAB and Simulink*, MathWorks Inc., 2023.

[28] ——, *Monte Carlo Localization (MCL)*, 2024. Retrieved May 2025.

[29] E. B. Olson, *Real-time correlative scan matching*, in Proc. 2009 IEEE Int. Conf. Robotics and Automation (ICRA), 2009, pp. 4387–4393. DOI: `https://doi.org/10.1109/ROBOT.2009.5152706`.

[30] D. Pakkala, N. Känsäkoski, T. Heikkilä, J. Backman, and P. Pääkkönen, *On design of cognitive situation-adaptive autonomous mobile robotic applications*, Comput. Ind., 152 (2025), p. 104263. DOI: `https://doi.org/10.1016/j.compind.2025.104263`.

[31] S. T. Pfister, K. L. Kriechbaum, S. I. Roumeliotis, and J. W. Burdick, *Weighted range sensor matching algorithms for mobile robot displacement estimation*, in Proc. IEEE Int. Conf. Robotics and Automation (ICRA), vol. 2, 2002, pp. 1667–1674. DOI: https://doi.org/10.1109/ROBOT.2002.1014782.

[32] V. H. Phung and E. J. Rhee, *A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets*, Applied Sciences, 9 (2019), p. 4500.

[33] A. H. Reynolds, *Convolutional neural networks (cnns)*. Available from: https://anhreynolds.com/blogs/cnn.html. Accessed: 4 March 2025.

[34] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, *Orb: An efficient alternative to sift or surf*, in Proc. Int. Conf. Computer Vision (ICCV), 2011, pp. 2564–2571. DOI: https://doi.org/10.1109/ICCV.2011.6126544.

[35] A. Segal, D. Haehnel, and S. Thrun, *Generalized icp*, in Proceedings of Robotics: Science and Systems (RSS), Seattle, WA, USA, June 2009, pp. 435–442. DOI: https://doi.org/10.15607/RSS.2009.V.021.

[36] T. Shan, B. Englot, F. Duarte, C. Ratti, and D. Rus, *Robust place recognition using an imaging lidar*, in Proc. IEEE Int. Conf. Robotics and Automation (ICRA), 2021, pp. 5469–5475. DOI: https://doi.org/10.1109/ICRA48506.2021.9560890.

[37] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, arXiv preprint arXiv:1409.1556, (2014). DOI: https://doi.org/10.48550/arXiv.1409.1556.

[38] N. Stathoulopoulos, E. Pagliari, L. Davoli, and G. Nikolakopoulos, *Redundant and loosely coupled lidar-wi-fi integration for robust global localization in autonomous mobile robotics*, in Proc. 21st Int. Conf. Advanced Robotics (ICAR), 2023, pp. 121–127. DOI: https://doi.org/10.1109/ICAR58858.2023.10406402.

[39] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, *Multi-view convolutional neural networks for 3d shape recognition*, in Proc. IEEE Int. Conf. Computer Vision (ICCV), 2015, pp. 945–953. DOI: https://doi.org/10.1109/ICCV.2015.114.

[40] X. Sun, M. Yue, H. Wang, Y. Liu, and X. Zhao, *An integrated framework for multi-amr based cl-cbs and mpc-apf in warehousing scenario*, Simul. Model. Pract. Theory, 142 (2025), p. 103122. DOI: https://doi.org/10.1016/j.simpat.2025.103122.

[41] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, *Going deeper with convolutions*, in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1–9. DOI: https://doi.org/10.1109/CVPR.2015.7298594.

[42] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*, MIT press, 2005.

[43] M. A. V. Torres, A. Braun, and A. Borrmann, *Occupancy grid map to pose graph-based map: Robust bim-based 2d-lidar localization for lifelong indoor navigation in changing and dynamic environments*, arXiv preprint arXiv:2308.05443, (2023). Conference version: eWork and eBusiness in AEC: ECPPM 2022. DOI: https://doi.org/10.48550/arXiv.2308.05443.

[44] I. Ullah, D. Adhikari, H. Khan, M. S. Anwar, S. Ahmad, and X. Bai, *Mobile robot localization: Current challenges and future prospective*, Computer Science Review, 53 (2024), p. 100651. DOI: https://doi.org/10.1016/j.cosrev.2024.100651.

[45] Ultralytics, *v8PoseLoss*, retrieved May 2025.

[46] K. Wang, S. Jia, Y. Li, X. Li, and B. Guo, *Research on map merging for multi-robotic system based on rtm*, in Proc. IEEE Int. Conf. Information and Automation (ICInfA), 2012, pp. 156–161. DOI: https://doi.org/10.1109/ICInfA.2012.6246852.

[47] S. Yu, F. Yan, Y. Zhuang, and D. Gu, *A deep-learning-based strategy for kidnapped robot problem in similar indoor environment*, J. Intell. Robot. Syst., 100 (2020), pp. 765–775. DOI: https://doi.org/10.1007/s10846-020-01216-x.

[48] B. Zhang, Z. Peng, B. Zeng, and J. Lu, *2dliw-slam: 2d lidar–inertial–wheel odometry with real-time loop closure*, Meas. Sci. Technol., 35 (2024), p. 075205. DOI: https://doi.org/10.1088/1361-6501/acd2c9.