Available online at sjuoz.uoz.edu.krd

Science Journal of University of Zakho

Vol. 13, No.3 pp416–425July-September,2025

p-ISSN: 2663-628X
e-ISSN: 2663-6298

INTEGRATING CNN AND DICTIONARY MECHANISMS FOR EFFECTIVE LOOP CLOSURE DETECTION

Ayda Mohammed Sharif [1,*] and Sadegh Abdollah Aminifar[1]

[1]Department of Computer, College of Science, University of Soran, 44008, Kurdistan, Iraq.

*Corresponding author email: aydamhamadamin@gmail.com

Received: 14 May. 2025 Accepted:12 Jun. 2025 Published:06 Jul. 2025 https://doi.org/10.25271/sjuoz.2025.13.3.1579

ABSTRACT:

Loop closure detection (LCD) remains a critical challenge in visual Simultaneous Localization and Mapping (SLAM), particularly in environments with repetitive structures or sparse textures where traditional methods suffer from perceptual aliasing and computational inefficiency. This paper presents a robust and scalable LCD framework that integrates a lightweight Convolutional Neural Network (CNN) with a dictionary-based voting mechanism, optimized for accuracy and real-time performance in resource-constrained settings. The proposed CNN architecture, featuring a single convolutional layer with 32 filters, achieves 98% classification accuracy on the Greenhouse Scene Dataset-a structured agricultural environment. Complementing the CNN, a dynamic dictionary tracks class frequencies to detect loop closures via adaptive thresholding, eliminating the need for complex feature matching or geometric verification. Experimental results demonstrate real-time operation (0.076 seconds per 70 frames) and resilience to spatial distortions, maintaining 92% accuracy under pixel-level shifts. Compared to state-of-the-art methods, our approach reduces computational overhead and memory usage.

KEYWORDS: SLAM, Loop Closure, Machine Learning, CNN, Dictionary.

1. INTRODUCTION

Autonomous navigation in unknown environments relies on the ability of robots to build accurate maps while correcting accumulated localization errors—a task central to Simultaneous Localization and Mapping (SLAM) (Placed, 2023). Among SLAM components, loop closure detection (LCD) is pivotal for recognizing previously visited locations to rectify trajectory drift, ensuring globally consistent maps. Despite advancements, LCD remains challenging in environments with repetitive visual patterns (e.g., ware-houses, greenhouses) or low-texture regions, where hand- crafted features-based methods or geometric verification often struggle due to perceptual aliasing and computational inefficiency (Soncini, 2024). Recent work has explored deep learning to address these limitations, with Convolutional Neural Networks (CNNs) emerging as powerful tools for feature extraction and place recognition. Methods like NetVLAD (Arandjelovic, 2016).leverage large CNN architectures to generate global descriptors, while hybrid approaches such as SymBioLCD (Kim, 2021)fuse semantic and geometric data for robustness. However, these methods often prioritize accuracy at the expense of computational efficiency, rendering them im-practical for resource-constrained platforms. For instance, pretrained-based frameworks (Olid, 2018)require often GPU acceleration, and graph-based matching (Qin, 2021)introduces latency incompatible with real-time applications. Conversely, lightweight solutions like FAB-MAP (Cummins, 2008) accuracy in repetitive environments, achieving only 75% precision on structured datasets.

This study addresses these trade-offs by proposing a lightweight CNN combined with a dictionary-based voting mechanism, Optimized for accuracy and efficiency in repetitive, low-texture settings.

Our approach diverges from existing methods in two key ways:

- Lightweight Architecture: A single-layer CNN with 32 filters achieves 98% classification accuracy on the Greenhouse Scene Dataset and maintains 92% accuracy under pixel-level distortions.
- Dynamic Frequency Tracking: A dictionary mechanism replaces complex feature matching or geometric verification, enabling real-time loop closure detection (0.076s per 70 frames) without GPU reliance.
- The Greenhouse Scene Dataset (Xiao, n.d.),comprising 2,260 images of a structured agricultural environment, serves as the testbed for our experiments. This dataset captures the challenges of repetitive visual patterns and sparse textures, providing a rigorous benchmark for evaluating LCD robustness, our contributions are as follows:
- A computationally efficient LCD framework that com- bines CNN-based classification with dictionary-driven frequency counting, tailored for embedded systems.
- Empirical validation demonstrating 98% accuracy on the Greenhouse dataset and superior real-time performance compared to GPU-based or high computational pretrained methods.

The organization of this study is as follows: The subsequent section **2** provides a detailed review of the related work in the field. This is followed by an in-depth description of the proposed

* Corresponding author

approach in Section **3**. The subsequent sections **4** presents the experiments conducted and the results obtained. Finally, the study concludes with a discussion in Section **5** and summary of the findings and potential directions for future research in Section **6**

RELATED WORK

The evolution of loop closure detection (LCD) in visual SLAM has been shaped by three primary methodologies: handcrafted feature-based methods, deep learning-driven approaches, and hybrid techniques combining visual, se-mantic, and geometric data. Early LCD systems relied on handcrafted features such as SIFT, SURF, and ORB to generate visual descriptors (Barros, 2022).For instance, FAB-MAP (Cummins, 2008) pioneered a probabilistic framework using Bag-of-Words (BoW) models and Chow-Liu trees for place recognition, achieving real-time performance. In another work, RB- SLAM3 (Campos, n.d.)integrated ORB features with geometric verification (PnP/RANSAC) to improve robustness. However, such methods often struggle in repetitive or low-texture environments and incur high computational costs, limiting scalability on resource-constrained platforms (Chen, 2022)While these methods inspired early feature-matching paradigms,
Their dependency on manual tuning and susceptibility to repetitive scenes motivated our shift toward learning-based frequency tracking. As for deep learning-driven approaches, CNNs have revolutionized LCD by automating feature extraction and improving robustness to environmental changes. For instance, (Arandjelovic, 2016) introduced, NetVLAD, a trainable VLAD layer atop CNNs for global descriptor learning, achieving state-of- the-art (SOTA) recall on large-scale datasets. However, its reliance on AlexNet (Krizhevsky, 2017)and VGG16 (Simonyan, 2024)necessitates GPU acceleration, rendering it impractical for embedded systems. Subsequent works, such as DenseLoop (Yu, 2019)employed DenseNet features with locality-sensitive hashing (LSH) to reduce matching complexity but their improved recognition accuracy inherently produces high-dimensional descriptors, which can increase memory demands even after dimensionality reduction techniques. These studies highlight the trade-off between accuracy and computational load. Our lightweight CNN (32 filters, single layer) addresses this by reducing parameters while maintaining 98% accuracy, demonstrating that simpler architectures can suffice in structured environments.

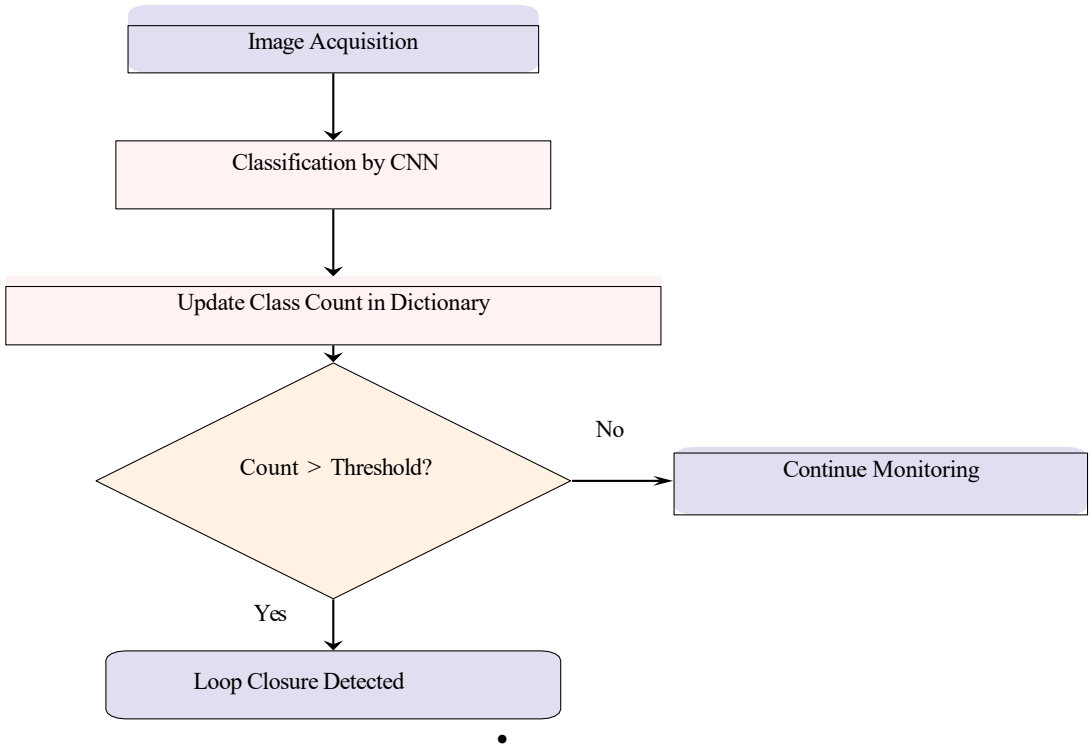
In another studies, authors used hybrid approaches to fuse CNN features with spatial or semantic data to enhance robustness. For instance, SymBioLCD (Kim, 2021) combined CNN-derived object features with BoW, using temporal constraints to reduce false positives in dynamic scenes.

Finally, efforts to optimize LCD for constrained environments have focused on architectural efficiency. For instance, (Xu, 2021)proposed a single- layer CNN (SCNN) for loop detection, with small number of parameters. Similarly, (Li, 2023) leveraged depth wise separable convolutions (DSC) to reduce computation, but their accuracy drops in repetitive settings. Our approach advances these efforts by tailoring the CNN to repetitive agricultural scenes (Greenhouse Dataset) and integrating a dynamic dictionary. This combination achieves higher accuracy (98%) while maintaining low computations cost.

PROPOSED APPROACH

To address the challenges of loop closure detection in visually complex and repetitive environments, we propose a robust and scalable method that integrates Convolutional Neural Networks (CNNs) with a dictionary-based mechanism. This approach leverages the Greenhouse Scene Dataset, which provides a diverse and challenging testbed for evaluating loop closure detection methods in constrained settings.

Figure 1 presents the integrated components of the proposed approach.



- **Figure 1:** Flowchart of the loop closure detection process

CNN-Based Classification Model

The cornerstone of our approach is a CNN-based classification model that identifies distinct visual classes within the dataset, which help a key-value dictionary to track the locations. The dataset comprises 2,260 images, we group the images into sequences of 60 consecutive frames, each representing a unique class or "spot" in the environment, except for the last class, which contains 40 frames. We deliberately left the last class as is to avoid making the model overly strict and to maintain simplicity. The CNN used is a lightweight architecture with a single convolutional layer, designed to balance computational efficiency and accuracy. The key parts are as follows:

- **Grouping of Images:** The dataset is segmented into multiple classes, with most containing 60 consecutive frames, except the last class with 40 frames.
- **Training Process:** The lightweight CNN is trained on these grouped sequences, achieving a classification accuracy of 98% during testing.

Key-Value Dictionary Mechanism:

Complementing the CNN model, we implement a dictionary-based mechanism to track and identify loop closures during deployment. This mechanism operates by maintaining a count of the frequency with which each class is detected in real time. More specifically, the dictionary is implemented
as a simple key-value store in which each key corresponds to a class label predicted by the CNN, and each value is an integer counter representing the number of times that class has been detected during deployment. For every incoming frame, the CNN predicts a class, and the corresponding counter in the dictionary is incremented by one. If the count for any given class surpasses the threshold (set to 70 detections), a loop closure is declared. This update process is intentionally designed to be computationally lightweight and easy to maintain in real-time settings.
The mathematical formulation of the dictionary mechanism is as follows:
Let:

- $C = \{1, 2, ..., N\}$ be the set of class labels (with $N = 38$ in our case),
- $D : C \rightarrow \mathbb{Z}_{\geq 0}$ be the dictionary, mapping each class label to its frequency count,
- $f_t \in C$ be the class predicted by the CNN for the input frame at time t.

The dictionary update rule is defined as:
$D(f_t) \leftarrow D(f_t) + 1$

This operation is performed in real time for each incoming frame.

Setting Thresholds:

Rather than comparing individual frames, we aggregate frames into distinct classes. This approach mitigates overfitting and reduces computational cost. In this study, we utilize 60 frames per class, providing a balanced and comprehensive representation of each visual class

- The selection of 60 frames per class, or 'spot', was refined through trial and error, including manually checking the images to ensure sequences of this length properly captured distinct environmental locations. This approach is crucial for model performance. Using 60 frames per spot ensures each spot is well-represented in the dataset. This helps the CNN model learn the distinguishing features of each spot more effectively, leading to higher classification accuracy, as presented in Section 4. Additionally, utilizing this number of frames per spot captures more variability within each class, such as different angles and lighting conditions that might occur in real-world scenarios. This diversity aids the model in generalizing better to new, unseen data.
- Using a sufficient number of frames per class helps prevent overfitting. Overfitting occurs when the model learns the training data too well, including noise and minor details, which can negatively impact its performance on new data. A larger dataset helps mitigate this risk.
- Importantly, during the training phase, the model shows a classification accuracy of 96% (worst-case), and during testing with shifted pixels, it achieves 94% accuracy. This indicates a 10% margin for potential misclassifications. Therefore, the threshold for loop closure detection is set to 70, even though the dataset contains 60 frames per class.

If any class count exceeds the threshold of 70 detections, a loop closure is detected, signaling the robot's re-encounter of a previously visited region. The mathematical formulation is as follows:
Let τ denote the threshold for loop closure detection. A loop closure is declared when:
$\exists\, c \in C$ such that $D(c) \geq \tau$
In our implementation, we empirically set $\tau = 70$, providing robustness to occasional misclassifications.
Specifically for the threshold selection justification, the notations are as follows:
Let:

- L be the number of frames per class (typically $L = 60$),
- ϵ be the expected misclassification rate (empirically observed as $\epsilon = 0.10$ in the worst case),
- Δ be a safety buffer to tolerate classification noise.

We estimate the expected number of correct detections in a single traversal as:
$(1 - \epsilon) \cdot L = 0.9 \cdot 60 = 54$
To ensure a reliable loop closure signal that surpasses the number of correct detections in a single traversal (even with partial overlap), we define the threshold as:
$\tau = L + \Delta = 60 + 10 = 70$
This formulation is supported by empirical tests shown in Table 1 (Section 4), where the 60-frame class size with a 10-frame margin (τ =70) produced the highest classification accuracy and most reliable loop detection performance.

Integration of Components Thresholds:

The proposed approach works as follows:

- **Image Classification:** As the robot navigates the environment, each captured image is classified into one of the predefined classes by the trained CNN.
- **Class Frequency Count:** The dictionary dynamically updates a count for each detected class based on CNN's predictions.
- **Thresholding for Loop Detection:** If the count for any class exceeds a predefined threshold of detections, the system identifies a loop closure.

EXPERIMENTS AND RESULTS

The experiments were executed in the jupyter notebook in the Python programming environment utilizing the Pan- das, NumPy, Pillow, scikit-learn, TensorFlow, Keras, and OpenCV libraries on a laptop with 16GB of ram. The dataset comprises 2,260 RGB images, each with a resolution of 640x480 pixels, representing repetitive and visually complex environments. Preprocessing procedures involved converting the images to grayscale and resizing them to 32x32 pixels. The dataset was categorized into 38 distinct classes, with each class containing 60 images, except for the final class, which contained 40 images. To ensure consistency, pixel values were normalized to a range of [0, 1].

Furthermore, the dataset was partitioned into training (80%) and testing (20%) subsets. To ensure this split result is not an artefact of a particular training run, we repeated the entire training and evaluation process several times with different random initializations. In every run the test accuracy converged to 98% on the original images, mirroring the stable learning curves. The experimental procedures are depicted in Figure 2.This choice for using 60 frames per spot ensures each spot is well-represented in the dataset and was further validated by empirical results from a comparative analysis of classification accuracy on original test images. As presented in Table 1, which compared using 50, 60, and 70 frames per spot. The findings indicated that the 60-frame configuration achieved 98% accuracy on original images, outperforming both a 50-frame configuration (97%) and a 40-frame setup (96%). With regards to class size, the 60-frame configuration consistently outperformed the others, yielding the highest accuracy because it balances two opposing factors (1) Shorter class windows (e.g., 50 frames) may increase the number of classes, but may not capture enough intra- class variation, reducing the model's ability to generalize (2) Longer class windows (e.g., 70 frames) reduce the number of classes, but may inadvertently include transitional scenes or overlapping visual regions, increasing intra-class variability and reducing separability. Therefore, the 60-frame configuration provides an optimal balance between within-class consistency and between-class distinctiveness, which is essential in highly repetitive environments like greenhouses.

A lightweight Convolutional Neural Network (CNN) architecture was designed to balance computational efficiency with high classification accuracy. The architecture comprised a single convolutional layer with 32 filters, a kernel size of (3, 3), and ReLU activation, followed by a flattening layer to convert feature maps into a one-dimensional vector. This was succeeded by a dense hidden layer with 68 neurons and ReLU activation, and an outpt layer employing softmax activation for multi-class classification across 38 classes. The model was trained using the Adam optimizer with a learning rate of 0.001, categorical cross-entropy as the loss function, 200 epochs, and a batch size of 32. The training process yielded a classification accuracy of 98% on the test set, demonstrating the effectiveness of the lightweight CNN architecture. Notably, the model reached an accuracy of approximately 98% by the 45th epoch, as shown in Figure 3. To enhance the interpretability of the results presented in Figure 4, a logarithmic scale was applied. Throughout the training process, the model's accuracy consistently remained above 96%, even by the 200th epoch, indicating stable performance. Consequently, an accuracy threshold of 96% is adopted for further analysis.

The model was trained on 1,808 images and tested on 452 images, following 80:20 split ratio. Notably, we avoided equal class distribution in train-test splits which aligns with real-world scenarios where class imbalances are common. The performance metrics, including precision, recall, and F1-score, per class are presented in Table 2. Moreover, as present in Figure 5, the confusion matrix exhibits a very low degree of false alarms (10 out 0f 452), as indicated by the substantial values along the diagonal. The pattern is consistent across all classes, with most misclassifications occurring between adjacent or similar classes. In an additional experiment, we shuffled the 2,260 training images and shifted them by 1 pixel to the right, setting the vacated leftmost column to zero, thereby altering 32 pixels per image. In this experiment, we use the entire dataset, given they all will change, as presented in Table 3. This manipulation aimed to test the robustness of the classifier. Despite this modification, the CNN achieved 92% accuracy, demonstrating its resilience to slight spatial distortions. The model exhibited the high-performance metrics per class, as presented in Table 4, even after the shifting. The dictionary is implemented as a key-value store, where each key corresponds to a class label predicted by the CNN, and each value is an integer count representing the number of times that class has been detected during runtime. Moreover, the update process is straightforward and computationally inexpensive. For every incoming frame, the CNN outputs a predicted class label. The corresponding count in the dictionary is incremented by 1. If any class's count exceeds the pre-defined threshold (e.g., 70 detections), a loop closure is declared. This mechanism avoids temporal matching or probabilistic reasoning, enabling efficient up- dates in real time with minimal computational overhead.

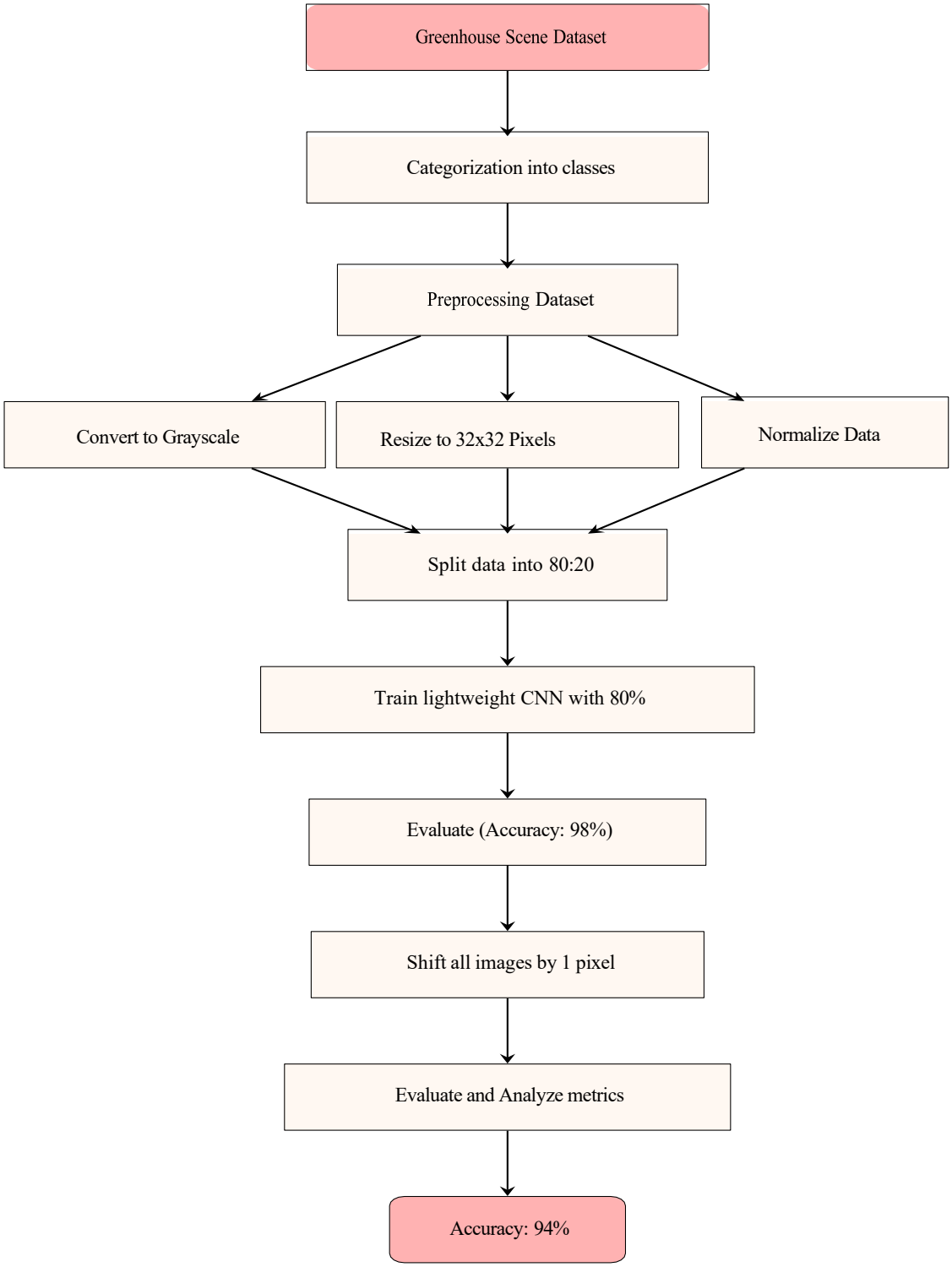


Figure 2: Block diagram of the loop closure detection process.

Table 1: Empirical comparison of classification performance using different numbers of frames per spot (50, 60, and 70), showing accuracy trade-offs and number of resulting classes.

	50	**60**	**70**
Original (%)	97%	98%	97%
Spots/Classes (#)	46	38	33

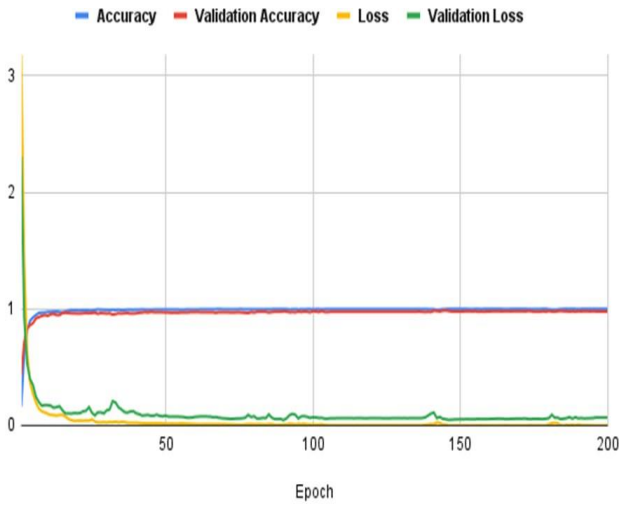


Figure 3: The accuracy during training, validation accuracy, loss function during training, and loss function during validation in 200 epochs.

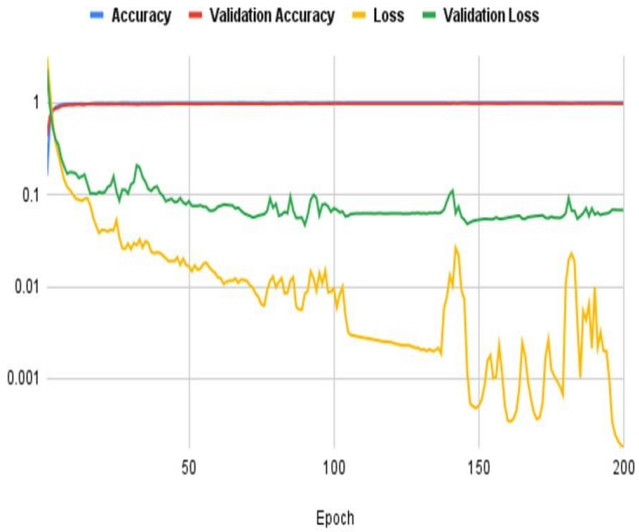


Figure 4: The accuracy during training, validation accuracy, loss function during training, and loss function during validation, when values were logged in 200 epochs.

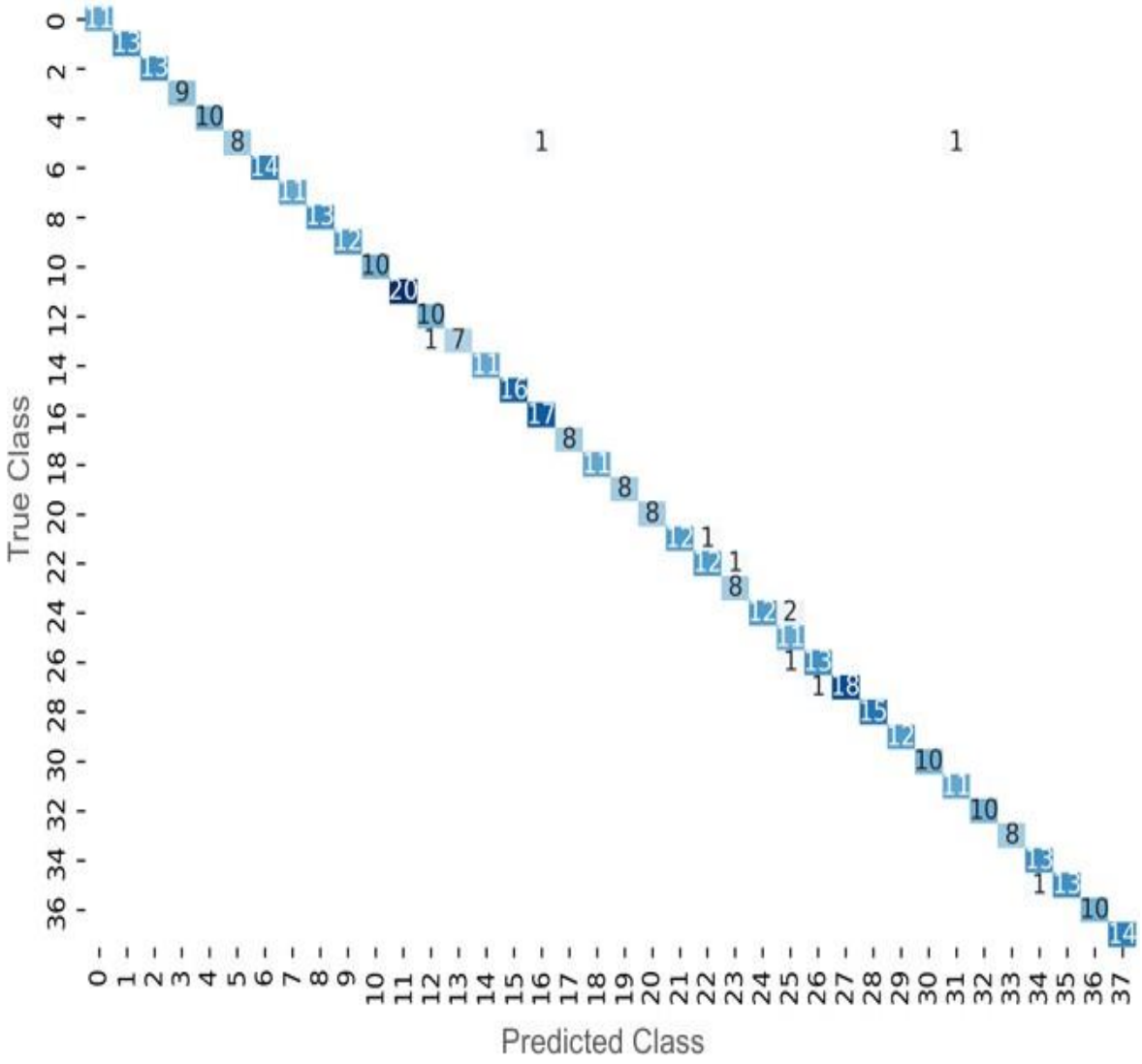


Figure 5: Matrix of the model on the test set where the x-axis represents Predicted Class and the y-axis represents True Class.

Table 2: Performance metrics (precision, recall, F1-score) of the CNN on test data, measured per class.

Class	Precision	Recall	F1-Score	Samples
1	1	1	1	11
2	1	1	1	13
3	1	1	1	13
4	1	1	1	9
5	1	1	1	10
6	1	0.8	0.89	10
7	1	1	1	14
8	1	1	1	11
9	1	1	1	13
10	1	1	1	12
11	1	1	1	10
12	1	1	1	20
13	0.91	1	0.95	10
14	1	0.88	0.93	8
15	1	1	1	11
16	1	1	1	16
17	0.94	1	0.97	17
18	1	1	1	8
19	1	1	1	11
20	1	1	1	8
21	1	1	1	8
22	1	0.92	0.96	13
23	0.92	0.92	0.92	13
24	0.89	1	0.94	8
25	1	0.86	0.92	14
26	0.79	1	0.88	11
27	0.93	0.93	0.93	14
28	1	0.95	0.97	19
29	1	1	1	15
30	1	1	1	12
31	1	1	1	10
32	0.92	1	0.96	11
33	1	1	1	10
34	1	1	1	8
35	0.93	1	0.96	13
36	1	0.93	0.96	14
37	1	1	1	10
38	1	1	1	14

Table 4: Performance metrics (precision, recall, F1-score) of the CNN on 1 pixel shifted test data, measured per class.

Class	Precision	Recall	F1-Score	Samples
1	0.96	0.90	0.93	60
2	0.97	0.95	0.96	60
3	1.00	0.67	0.80	60
4	0.59	1.00	0.74	60
5	0.94	0.80	0.86	60
6	0.95	0.97	0.96	60
7	0.98	0.98	0.98	60
8	1.00	0.92	0.96	60
9	0.91	1.00	0.95	60
10	0.95	0.97	0.96	60
11	0.98	0.90	0.94	60
12	1.00	0.93	0.97	60
13	0.98	1.00	0.99	60
14	0.97	0.95	0.96	60
15	0.96	0.73	0.83	60
16	1.00	0.75	0.86	60
17	0.98	0.97	0.97	60
18	0.63	1.00	0.77	60
19	1.00	0.90	0.95	60
20	0.98	0.93	0.96	60
21	0.95	0.93	0.94	60
22	0.86	0.98	0.91	60
23	0.83	0.97	0.89	60
24	1.00	0.85	0.92	40
25	1.00	0.98	0.99	60
26	0.97	1.00	0.98	60
27	0.72	0.97	0.83	60
28	1.00	0.58	0.74	60
29	0.87	1.00	0.93	60
30	1.00	0.90	0.95	60
31	0.97	0.98	0.98	60
32	0.95	0.87	0.90	60
33	0.96	0.85	0.90	60
34	1.00	0.87	0.93	60
35	0.86	1.00	0.92	60
36	1.00	0.87	0.93	60
37	0.93	0.95	0.94	60
38	0.90	1.00	0.94	60

Table 3: Accuracy of CNN across all classes on the original test dataset vs. the whole dataset subjected to a 1-pixel shift.

Condition	Number of Samples	Accuracy
No shifts (Original Image)	452	98%
Shift by 1 pixel	2260	92%

2. DISCUSSION

The single-layer Convolutional Neural Network (CNN) achieved an impressive accuracy of 98% on the Greenhouse Scene Dataset, utilizing 32 filters and a relatively simple architecture. This high performance can be attributed to the structured nature of the greenhouse environment, characterized by repetitive patterns and controlled lighting conditions. Such factors likely facilitated the model's ability to extract relevant features, enabling the shallow network to perform effectively without requiring extensive depth or complexity. The minimalistic architecture, comprising a single convolutional layer and a small dense layer, mitigated the risk of overfitting while still maintaining sufficient discriminative power for class-specific features. These results suggest that complex neural network architectures are not always necessary to achieve high performance in specialized environments. The lightweight design makes this model particularly suitable for deployment on resource-constrained systems where computational efficiency is a priority. Notably, the confusion matrix indicates that misclassifications occurred between adjacent classes, suggesting that errors were likely made between visually similar or

structurally similar environments, rather than between entirely dissimilar ones. This pattern implies that the model's discriminative power remains strong, but refinement between similar classes could enhance performance further. When subjected to a shift of 1 pixel, simulating spatial distortions commonly encountered in real-world settings, the model retained an accuracy of 92%. This robustness to minor perturbations indicates that the model focuses on higher-level structural features, such as object shapes and layout, rather than being overly dependent on exact pixel positions. This behavior is indicative of the CNN's ability to generalize well despite variations in image positioning. Furthermore, preprocessing steps such as converting the images to grayscale and resizing them to 32x32 pixels likely contributed to the model's invariance to noise and spatial distortions. This makes the model well-suited for real-world applications where minor camera jitter or environmental changes might occur, ensuring consistent performance de- spite such challenges.

The system demonstrated real-time processing capability, handling 70 frames in just 0.076 seconds on a CPU (Intel i7-8750H). This remarkable efficiency is largely due to the lightweight CNN architecture, which minimizes computational load by avoiding complex operations while still maintaining high performance.

Additionally, the use of a dictionary mechanism for classifying frames, instead of re- lying on resource-intensive feature matching or geometric verification, further reduces processing overhead. Regarding memory footprint, the dictionary's requirements are mod- est. With 38 distinct classes, only 38 integer counters are maintained. Assuming 32-bit integers, this results in a total memory usage of approximately 152 bytes—well within the capabilities of even the most constrained embedded systems. The approach does not rely on storing high-dimensional feature descriptors or managing large image buffers, which further supports its deployment in low-power platforms.

As for handling class overlaps, we mitigated this risk during dataset preparation by carefully segmenting the 2,260 frames into non-overlapping sequences of 60 consecutive frames, each designated as a distinct "spot." Manual inspection was used to ensure that each class represents a visually coherent and spatially distinct segment of the environment.

Given that in repetitive environments such as greenhouses, some visual similarity between classes is unavoidable, the loop closure detection threshold was set conservatively at 70, providing a margin that helps filter out occasional misclassifications. This design prioritizes robustness over aggressive detection and ensures that loop closures are only declared when there is strong and sustained evidence of revisitation. The confusion matrix in the results section supports this, showing that misclassifications are rare and tend to occur only between visually adjacent classes. Overall, the dictionary mechanism is intended to be minimalistic yet effective, ensuring high performance without introducing computational or memory burdens. Additionally, the dictionary-based loop closure mechanism is computationally lightweight—look up and up- date operations are O (1) per frame. Thus, latency does not scale with the number of classes in a significant way, even with hundreds of classes.

By grouping frames into classes (60 frames per class), the system reduces the computational demands of processing each frame individually. This efficiency ensures that the model can be deployed on low-power embedded systems, such as agricultural robots or autonomous vehicles, which typically lack GPUs, making it ideal for real-time applications in resource-constrained environments.

For loop closure detection, a threshold of 70 detections was established, despite the classes being defined with 60 frames. This threshold was chosen to accommodate potential misclassifications, acknowledging that the worst-case accuracy of the model is 90%. The additional buffer (70 detections instead of 60, although the frames are 60 not 100) ensures that the system remains robust to transient errors while avoiding false positives. More specifically, the 70 frames margin also serves to ensure that a loop closure is not declared prematurely during a partial revisit or due to visual ambiguity. It prioritizes specificity over sensitivity, which aligns with our system's objective of ensuring reliable operation in resource-constrained and visually ambiguous environments. This adaptive approach to thresholding strikes an optimal balance between sensitivity and specificity, ensuring reliable loop closure detection in environments where repetition and environmental variations are prevalent. Such a strategy is crucial for applications involving navigation and path recognition in repetitive environments, where the accurate detection of loop closures is essential for maintaining system reliability.

With regards to CPU/GPU specifications and power consumption, all experiments were conducted without any GPU acceleration to simulate the performance of higher- end embedded or low-power CPUs. The CNN inference and dictionary operations were performed entirely on the CPU, supporting our claim of GPU-independent real-time operation. Moreover, since our framework uses only the CPU and avoids GPU-heavy tasks (e.g., feature matching, geometric verification), the power consumption remains significantly lower than comparable GPU-based LCD systems. Notably, the model achieves real-time performance of 0.076 seconds per 70 frames on CPU, which translates to approximately 920 frames per second. This high throughput, achieved with- out GPU support, further emphasizes the model's suitability for real-time applications on constrained hardware.

COMPARISON WITH STATE-OF-THE-ART METHODS

In contextualizing our method against several state-of-the-art studies, emphasizing efficiency, and applicability to repetitive scenes, our method contributes to bridge the gap between accuracy and efficiency in LCD for repetitive environments.

(Zhou, 2025)proposed a lightweight Siamese capsule network which leverages spatial hierarchies and attention to features, achieving robustness in dynamic urban environments but at the cost of higher complexity. Moreover, their method achieves 0.074s per frame but relies on GPU acceleration during training. Their pruning strategy mitigates computational load but still requires more resources than our dictionary-based voting.

(Shi, 2024) employs a multi-stage deep learning pipeline (MixVPR + SuperPoint + Light- Glue), offering robustness in diverse environments at the cost of complexity. Their method achieves 0.074s per frame on GPU (NVIDIA Jetson) but requires significant computational resources, limiting deployment on low-end hardware. Additionally, they leveraged MixVPR for holistic global descriptors and LightGlue for adaptive feature matching, enabling robustness to viewpoint changes.

(Li, 2023) introduced CoCALC, a self-supervised visual place recognition (VPR) approach combining appearance and geo- metric information, achieving state-of-the-art performance on benchmarks. While CoCALC excels in long-term VPR with minimal memory footprint, its geometric verification introduces latency during post-processing. In contrast, our method eliminates feature matching entirely through a dynamic dictionary mechanism, achieving 0.076s per 70 frames on CPU without geometric checks, making it more suitable for repetitive agricultural environments.

(Mehta, 2021)proposed MobileViT which mixes depth-wise CNN layers with lightweight self-attention, giving it global context without ballooning parameters. MobileViT demonstrated stability across classification, detection and dense segmentation (without heavy augmentation tricks) suggests strong generalization, complementing the pixel-shift or viewpoint-centric robustness of existing LCD methods. However, different from ours, MobileViT is mainly about general-purpose vision transformers for mobile devices, not specifically loop closure detection.

Comparative Analysis of the three studies and our method is presented in **Table 5**.

Table 5: Comparison of our method with several notable studies.

Aspect	Our Method	Zhou and Sun (2025)	Shi et al. (2024)	Li et al. (2023)	Mehtaand Rastegari (2021)
Architecture	Single-layer CNN + dictionary	Pruned-Capsule network	MixVPR+ LightGlue	DSC + geometric check	MobileNet+V2 blocks
Accuracy	98% (Greenhouse)	94.7% (CityCentre)	0.473mATE (KAIST)	AUC 0.98, Rp 1000.51($Nordland$)	74.8% (ImageNet)
Efficiency	1.085 ms/frame (CPU)	74 ms/frame (GPU)	12.5ms/frame (GPU)	86.27 ms/frame (CPU)	7.25 ms/frame (iPhone12 NE)
Robustness	Pixel shifts	Viewpoint changes	Seasonal changes	viewpoint changes	weight-decay changes
Dataset	Greenhouse	NewCollege, CityCentre KITTI	EuRoC, KAIST 4Seasons	Nordland,SPEDTest, Gardens Point ous Loop, Cross-Seasons	ImageNet, COCO PASCAL

CONCLUSION

The proposed methodology integrates the advantages of CNN-based image classification with a simple yet efficient dictionary-based counting mechanism, thereby achieving robust loop closure detection. By exploiting the structured nature of the Greenhouse Scene Dataset, which features repetitive patterns and controlled lighting conditions, our approach ensures accurate and reliable detection of loop closures. This, in turn, enhances robotic navigation in repetitive environments, where reliable detection of previously visited locations is crucial. The model's high accuracy, reaching 98% in training and 92% when tested with spatial distortions, highlights its effectiveness in complex, visually challenging scenarios. Additionally, the model's robustness to pixel-level shifts, achieving 94% accuracy even with 1- pixel image shifts, demonstrates its resilience to real-world perturbations, such as minor camera jitter.

Our experimental results confirm that the proposed method is well-suited for use in resource-constrained robotic systems. The lightweight CNN architecture, which minimizes computational overhead while maintaining high performance, enables real-time operation on low-power embedded systems. This efficiency, coupled with the robustness demonstrated by the model, makes it a promising solution for visual SLAM applications in environments with limited computational resources, such as agricultural robots and autonomous vehicles. The model's ability to maintain high classification accuracy despite misclassifications occurring primarily between visually similar classes further supports its reliability for practical applications. The adaptive thresholding for loop closure, set at 70 detections to accommodate potential misclassifications, balances sensitivity and specificity, ensuring robust performance even in the presence of minor errors.

Finally, our methodology represents a significant advancement in visual SLAM, addressing the persistent challenges of loop closure detection in repetitive and visually complex environments. The simplicity, efficiency, and reliability of the approach offer a practical solution that advances the state-of-the-art in visual SLAM systems.

FUTURE WORK

Future research could focus on several key areas to augment the robustness, scalability, and versatility of the proposed loop closure detection framework. The overarching goal is to facilitate its deployment in increasingly diverse and dynamic operational contexts. A significant avenue of investigation could be the development and integration of automatic labeling techniques, which offer the potential to streamline the dataset preparation phase required for training and evaluation.

A critical aspect of future work could involve rigorous validation of the method's performance in environments characterized by less structure and greater dynamism than the Greenhouse Scene Dataset. Consequently, a priority will be to expand the empirical evaluation to diverse datasets, including comprehensive testing on established public bench- marks such as KITTI, Nordland, and 4Seasons. This ex- tensive testing is of significance for assessing the method's generalization capabilities and its efficacy in managing high scene variability.

Finally, exploration into the hybrid integration of complementary sensory modalities could be undertaken to fuse additional environmental data with visual inputs, thereby bolstering the overall robustness and reliability of the loop closure detection system. These research trajectories are envisioned to culminate in a more universally applicable and resilient loop closure detection solution, thereby advancing the capabilities of visual SLAM systems for a broader spectrum of autonomous navigation application.

Acknowledgment:

This research constitutes a component of Ayda Mohammed Sharif's Mater study, undertaken as part of the requirements for the completion of her Master's degree.

Ethical Approval:

This study did not involve any experiments on humans or animals and did not require ethical approval. All data used in this research were obtained from publicly available datasets and used in accordance with the terms and conditions stated by the dataset providers.

Author Contributions:

A.M.S., conceptualized the study, implemented the method, performed the experiments, and wrote the manuscript. S.A., supervised the research, provided feedback on methodology and analysis, and reviewed the final manuscript. Both authors have read and approved the submitted version.

Declaration:

The authors declare that there is no conflict of interest regarding the publication of this paper.

Funding:

This research received no external funding and was conducted as part of the MSc program requirements at Soran University.

REFERENCES

Arandjelovic, R. G. (2016). NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. Retrieved from CVPR 2016 Open Access Repository:

Barros, A. M. (2022, February 10). A Comprehensive Survey of Visual SLAM Algorithms. Retrieved from MDPI: https://doi.org/10.3390/robotics11010024

Campos, C. E. (n.d.). ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM. Retrieved from IEEE Xplore: https://doi.org/10.1109/TRO.2021.3075644

Chen, Y. Z. (2022, July 21). Fast and robust loop-closure detection using deep neural networks and matrix transformation for a visual SLAM system. Retrieved from SPIE Digital Library: https://doi.org/10.1117/1.JEI.31.6.061816

Cummins, M. &. (2008, June 1). FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance. Retrieved from SAGE *Journals:* https://doi.org/10.1177/0278364908090961

Kim, J. J. (2021). SymbioLCD: Ensemble-Based Loop Closure Detection using CNN-Extracted Objects and Visual Bag-of-Words. Retrieved from IEEE Xplore: https://doi.org/10.1109/IROS51168.2021.9636622

Krizhevsky, A. S. (2017, May 24). ImageNet classification with deep convolutional neural networks. Retrieved from ACM Digital Library: https://doi.org/10.1145/3065386

Li, K. W. (2023). CoCALC: A Self-Supervised Visual Place Recognition Approach Combining Appearance and Geometric Information. Retrieved from IEEE Xplore: https://doi.org/10.1109/ACCESS.2023.3246803

Mehta, S. &. (2021, October 5). MobileViT: Light-weight, General-purpose, and Mobile-friendly Vision Transformer. Retrieved from arXiv: https://doi.org/10.48550/arXiv.2110.02178

Olid, D. F. (2018, August 20). Single-View Place Recognition under Seasonal Changes. Retrieved from arXiv: https://doi.org/10.48550/arXiv.1808.06516

Placed, J. A. (2023, March). A Survey on Active Simultaneous Localization and Mapping: State of the Art and New Frontiers. Retrieved from IEEE Xplore: https://doi.org/10.1109/TRO.2023.3248510

Qin, C. Z. (2021, April). Semantic loop closure detection based on graph matching in multi-objects scenes. Retrieved from ScienceDirect: https://doi.org/10.1016/j.jvcir.2021.103072

Shi, Y. L. (2024). A Robust and Lightweight Loop Closure Detection Approach for Challenging Environments. Retrieved from MDPI – Drones: https://doi.org/10.3390/drones8070322

Simonyan, K. &. (2024, September 4). Very Deep Convolutional Networks for Large-Scale Image Recognition. Retrieved from arXiv: https://doi.org/10.48550/arXiv.1409.1556

Soncini, N. C. (2024, August 12). Addressing the challenges of loop detection in agricultural environments. Retrieved from Wiley Online Library: https://doi.org/10.1002/rob.22414

Xiao, B. L. (n.d.). The greenhouse remote sensing image dataset. Retrieved from IEEE DataPort: https://dx.doi.org/10.21227/y6j3-fq20

Xu, B. Y. (2021, September 19). Loop Closure Detection for Visual SLAM Using Simplified Convolution Neural Network. Retrieved from SpringerLink: https://doi.org/10.1007/978-3-030-31967-0_6

Yu, C. L. (2019). A DenseNet feature-based loop closure method for visual SLAM system. Retrieved from IEEE Xplore: https://doi.org/10.1109/ROBIO49542.2019.8961714

Zhou, Y. &. (2025, March 4). A visual SLAM loop closure detection method based on lightweight siamese capsule network. Retrieved from Scientific Reports: https://doi.org/10.1038/s41598-025-90511-4