Intelligent Automation & Soft Computing

Tech Science Press

DOI: 10.32604/iasc.2024.051999

ARTICLE

Analyzing the Impact of Scene Transitions on Indoor Camera Localization through Scene Change Detection in Real-Time

Muhammad S. Alam[1,5,*], Farhan B. Mohamed[1,3], Ali Selamat[2], Faruk Ahmed[4] and AKM B. Hossain[6,7]

[1]Department of Emergent Computing, School of Computing, Universiti Teknologi Malaysia, Johor Bahru, 81310, Malaysia

[2]Malaysia-Japan International Institute of Technology (MJIIT), Universiti Teknologi Malaysia, Kuala Lumpur, 54100, Malaysia

[3]Media and Game Innovation Centre of Excellence (MaGICX), Universiti Teknologi Malaysia, Johor Bahru, 81310, Malaysia

[4]Department of Engineering Technology, University of Memphis, Memphis, TN, 38152, USA

[5]Department of Computer Science and Artificial Intelligence, College of Computing and Information Technology, University of Bisha, Bisha, 61922, Saudi Arabia

[6]School of Computing, Faculty of Engineering, Universiti Teknologi Malaysia, Johor Bahru, 81310, Malaysia

[7]Department of Information System and Cyber Security, College of Computing and Information Technology, University of Bisha, Bisha, 61922, Saudi Arabia

*Corresponding Author: Muhammad S. Alam. Email: shamsul20@graduate.utm.my

Received: 20 March 2024 Accepted: 25 April 2024 Published: 11 July 2024

ABSTRACT

Real-time indoor camera localization is a significant problem in indoor robot navigation and surveillance systems. The scene can change during the image sequence and plays a vital role in the localization performance of robotic applications in terms of accuracy and speed. This research proposed a real-time indoor camera localization system based on a recurrent neural network that detects scene change during the image sequence. An annotated image dataset trains the proposed system and predicts the camera pose in real-time. The system mainly improved the localization performance of indoor cameras by more accurately predicting the camera pose. It also recognizes the scene changes during the sequence and evaluates the effects of these changes. This system achieved high accuracy and real-time performance. The scene change detection process was performed using visual rhythm and the proposed recurrent deep architecture, which performed camera pose prediction and scene change impact evaluation. Overall, this study proposed a novel real-time localization system for indoor cameras that detects scene changes and shows how they affect localization performance.

KEYWORDS

Camera pose estimation; indoor camera localization; real-time localization; scene change detection; simultaneous localization and mapping (SLAM)

1 Introduction

Indoor camera localization is pivotal in identifying a camera's position and orientation within an environment relative to a specific object. This domain is a foundational aspect of artificial intelligence research, emphasizing indoor robot navigation. The primary objective here is to accurately determine

the camera's position, or its 'pose', within a physical space. Such localization techniques are crucial for various robotic operations, including navigation, scene reconstruction, and object recognition. Different methodologies have been employed in this field, including but not limited to direct, feature-based, deep learning, and hybrid approaches. Particularly, deep learning techniques focus on matching features from input imagery to predict the camera's precise location and orientation. The use of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) are predominantly observed, where CNNs are responsible for feature extraction from images, and RNNs are tasked with predicting the camera's pose. Recent advancements have notably enhanced the accuracy and speed of indoor camera localization. Despite these advancements, applications like indoor robot navigation and surveillance demand even more precise camera pose predictions. Achieving such precision necessitates real-time localization systems that adapt to changes within the scene, improving performance dynamically. Traditional systems typically localize within static environments and fall short during dynamic scene transitions. It highlights the need for a real-time localization strategy that accommodates scene variability, which is crucial for robotic guidance and mobile navigation applications.

Several algorithms for camera localization utilize public indoor datasets to position cameras accurately. PoseNet, notable for being the inaugural real-time method offering six degrees of freedom, employs a single RGB image for prompt indoor camera position prediction [1]. It stands out for its ability to localize cameras in real-time across diverse environments with six degrees of freedom (6-DoF) despite requiring substantial computational resources. Following PoseNet, another method utilizing a CNN detects scene shifts, facilitating feature extraction directly from imagery and negating the need for manual feature identification [2]. However, this reliance on extensive labelled datasets poses challenges in terms of cost and feasibility in industrial settings. Another approach leverages point cloud data for camera positioning, while visual simultaneous localization and mapping (SLAM) account for dynamic indoor elements [3]. Specifically, RGBD-SLAM [4] integrates semantic segmentation to support augmented reality and autonomous robot deployment in real-time within dynamic environments, underlining its relevance for augmented reality and robotic guidance [5]. Additionally, RSANets enhance object recognition by combining residual elements with semantic attention, demonstrating superior performance and accuracy [6]. In SLAM systems, precise real-time object detection is paramount for effective localization and mapping.

Deep learning has largely driven recent advancements in indoor camera localization, showing marked improvements in processing and application performance. However, certain robotic applications, like robot guidance, mobile robotics, and surveillance, require further refinement for enhanced accuracy. It is crucial to acknowledge that real-time deep learning systems processing video input may suffer accuracy degradation due to scene variations within the image sequences. Understanding the impact of these changes on performance in real-time settings is essential. Consequently, there is a need for a deep learning-based indoor camera localization system that evaluates performance fluctuations resulting from scene changes in image sequences.

This research introduces a novel camera localization strategy employing recurrent neural networks to surmount the limitations of current systems. This approach, integrating scene change detection with real-time localization, is tested using the 7-Scenes dataset. The findings reveal that this method achieves superior accuracy and robustness compared to previous studies. The contributions of this research are summarized below:

1. Development of a recurrent neural network-based approach for recognizing scene changes during image sequences.

2. Proposed a novel recurrent neural network-based indoor camera localization approach that performs real-time camera localization with superior accuracy compared to state-of-the-art camera localization approaches.
3. Development of a real-time indoor camera localization system that integrates the proposed recurrent neural network approach with a scene change detection approach.
4. Evaluate the performance of the recurrent neural network on a large indoor dataset that provides real-time performance and excellent accuracy.

In summary, the proposed recurrent neural network-based approach shows improved accuracy and robustness of indoor camera localization performance. This study contributes to the great role of indoor camera localization systems, especially for indoor robot navigation and augmented reality.

2 Related Works

Indoor camera localization is essential for many machine vision applications, such as indoor robot navigation. Recently, numerous deep learning-based camera localization algorithms have been proposed, improving real-time camera localization performance but still losing performance due to scene changes. Many approaches have been proposed for real-time camera localization, including structure-based, feature-based, direct, indirect, and hybrid methods.

2.1 Structure-Based Method

A deep learning-based camera localization system, Faster-RCNN [7], has been proposed for object recognition. It mainly works on semantic segmentation of multiscale targets based on object recognition. Convolutional neural network-based deep learning systems detect scenes where images are captured using mobile devices [8]. The system is designed for environments with weak or unavailable GPS signals, such as indoors. RGB-D SLAM [9] is mapped to multi-level semantic information for dynamic environments, which combines RGB and depth image information [10] and provides reliable and robust localization. It is not easy to evaluate accurately as standard benchmarks are required. An indoor localization system [11] has been used for service robots, integrating machine learning and image processing techniques for object recognition. It dealt with a real-world application but needed to provide a practical solution for object tracking. Another strategy for object detection [12] is to improve indoor camera localization systems that use combined audio and video frequency identification sensors to localize objects indoors. It focuses on multipath effects in indoor navigation but could benefit from further analysis. A neural network-based continuous scene representation method [13] uses raw sensor data to generate a constant scene representation. This research provides a solution to current scene representation problems.

2.2 Feature-Based Method

Feature-based camera localization methods, such as SIFT [14] and ORB [15], are the main approaches for camera localization, which detect and match features from the input images and 3D models of the scenes to predict the camera poses. The methods perform high accuracy and robustness but degrade accuracy when the scene changes. ORB-SLAM [16] is a feature-based method that predicts the camera trajectory using pose graph optimization. It performs real-time localization for large indoor environments and minimizes loop closure; however, it requires significant computational resources to avoid capturing the featureless region. Another feature-based system, EfiLoc [17], efficiently correlates features and 3D points for large indoor environments and provides potential performance improvement of indoor camera localization. Dynamic SLAM for mobile robot

navigation systems uses onboard processors and RGBD cameras that handle dynamic objects in the environment and predict high-performance real-time camera localization. The multi-sensor-based SLAM [18] combines information from multiple sensors and creates an accurate environment map, a promising solution for robust simultaneous localization and mapping.

2.3 Direct Methods

Direct monocular SLAM [19] predicts the camera trajectory by minimizing the photometric error, thereby improving the camera localization performance to reduce computational complexity and motion blur and overcome scene change problems. However, they are still sensitive to changes in light brightness and significant camera movement [20]. RDMO-SLAM [21], a real-time visual SLAM, uses optical flow to control dynamic object motion and predicts camera position; it improves real-time performance in a moving object environment while maintaining computational complexity. Another promising research [22] more accurately localizes a camera in known environments using learning maps that better understand the environments, especially defensive structures, and related objects. A multi-session real-time visual SLAM [23] controlled mapping in multiple sessions, where an indoor robot mapped different sessions, and even the robot did not know its initial position. It integrates data from large environments with multiple sensors, but the accurate localization of the large environment with multiple sensors is more complicated. Real-time visual SLAM [24] performs partial dense mapping with a single camera in different environments. Visual SLAM provides a real-time solution for more accurate localization in dynamic environments but requires high hardware installation. Another real-time visual SLAM [25] localized an indoor camera using cameras and depth sensors to monitor the location of an emergency service worker in a building, which requires significant computational resources, such as high-resolution images; otherwise, the performance is degraded.

2.4 Hybrid Methods

The hybrid method integrates direct and feature-based methods to utilize their influence and reduce weaknesses. DSO [26] provides high localization performance for a large environment, controlling scene changes and camera movements, but requires careful tuning of hyperparameters when massive camera movements occur. DSO requires initial feature points for camera motion, but accurately initializing this feature is more challenging. A geometry-constrained scale estimation method [27] estimates the scale factor of camera motion in monocular visual odometry to ensure optimal performance. Still, implementation requires expensive sensors such as LiDAR and radar. Other hybrid camera localization methods, PTAM [28] and ORB-SLAM [16], show high accuracy and robustness. Integrating real-time deep learning algorithms into augmented reality systems offers further advantages but requires high computational resources.

2.5 Deep Learning-Based Method

In recent years, deep learning-based indoor camera localization has attracted the attention of researchers and industry for autonomous robot navigation systems. The most pioneering deep learning-based real-time camera localization method is PoseNet [1]. The convolutional neural network is used for feature extraction from the input images and predicts the 6-DoF camera poses in real time. ViNet [29] introduces a new approach for visual-inertial odometry (VIO) that uses a recurrent neural network [30,31] to determine the mapping between two consecutive images and the pose changes between them. The ViNet is a sequence-to-sequence learning method that resulted in improved performance. However, it led to higher computational complexity in VIO systems and made the

technique less accessible for specific applications. An unsupervised convolutional neural network-based learning method estimates the ego-motion and depth of video scenes without depending on the ground truth [32]. Although highly dependent on training data, it performed less well than supervised learning because it was an unsupervised learning model.

A deep learning [33] approach predicts the camera position using an RGB-D camera that provides location information but requires comprehensive evaluation. A comprehensive evaluation of deep learning camera localization shows visual SLAM's recent developments and challenges. DeepLab [34], an unsupervised learning method with residual components evaluated on different datasets, shows improvements over existing approaches. The benchmark datasets showed outstanding results for semantic segmentation tasks; however, architectural layout and hyperparameters can affect performance. Another deep learning approach [35] describes the scene representation and improves accuracy and robustness. Due to the availability of accurate labeling data, the model required less time and resources. However, without labeling data, the model could achieve a different level of accuracy.

The weight-vision transformation model optimized computational complexity and achieved high localization accuracy [36]. Only specific localization environments or data sets can fully exploit the model's efficiency. Matching captured images with database images localizes an indoor device [36]. It improved the indoor camera localization system using an efficient image retrieval technique, but it needs to be validated more extensively under certain conditions in different indoor environments. A deep learning approach based on modifying geometric loss functions improved the localization performance [37]. It attracted the attention of researchers by introducing a novel technique for estimating the relative camera pose estimation technique. It works incredibly well in a specific dataset but cannot be generalized to unknown data or real-world environments. The indoor camera localization approach has improved localization performance and significantly advanced indoor navigation [38]. Deep attention needs to be fine-tuned with this deep learning approach, which can be challenging. The geometric Clifford algebra-based regression technique for camera position improves the localization performance of the camera [39]. It provides a theoretical basis for camera pose regression using conformal geometric algebra, although it must be empirically validated in real-world datasets or scenarios. A deep learning approach estimates the camera poses using the constraint of Epipolar geometry [40]. The camera localization results improved by using the Epipolar constraints to reduce the search space because noise and outliers in the input data can affect Epipolar approaches.

3 Methodology

3.1 Introduction

This section describes a detailed mechanism for detecting scene changes in image sequences and measuring the impact on the real-time localization performance of indoor cameras. The complete system includes a recurrent neural network-based architecture, detection of scene changes by visual rhythms, and evaluate the effects of scene changes in image sequences. The primary intent of scene change detection is to improve the accuracy and robustness of the localization of indoor cameras.

3.2 Scene Change Detection Using Visual Rhythm

The proposed approach describes the detecting scene changes process using a visual rhythm algorithm [41], as shown in Fig. 1. The following is the scene change detection process and the importance of capturing temporal dependencies in image sequences to identify scene changes accurately. Visual rhythm examines temporal changes in short video sequences and identifies temporal dependencies when scene changes occur. This section explains the scene change detection method in detail.

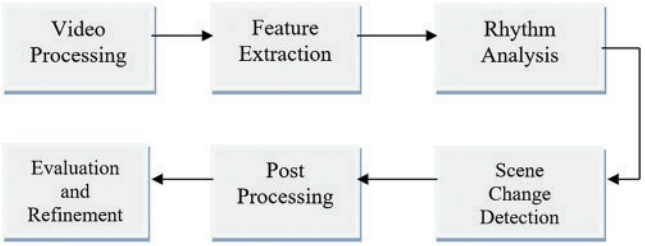

Figure 1: Scene change detection system through visual rhythm

First, short videos (V) are converted into independent images and undergo image preprocessing, such as scaling, converting RGB to grayscale, and enhancing contrast to improve image quality. Extracted visual features from each image and the changes over time are recorded in a short video. The standard method calculates the image difference, which is the difference between two adjacent images, and determines the absolute difference between two consecutive images. Calculate the temporal dependence of video using the extracted features in the rhythm analysis phase. Create a time series by adding the intensity of the pixels corresponding to the image or calculating statistical measures. The low-pass filtering technique is used for temporal smoothing or noise reduction and draws attention to the overall rhythm [42]. The mathematical representation of the visual rhythm is as follows:

$$V = \{f_v(x, y, t)\}, \quad x, y, t \in \{0, 1, 2 \ldots\} \tag{1}$$

V is the input video, t is the time, (x, y) is the pixel location, and $f_v(x, y, t)$ is the pixel value. V_T is the spatially reduced video can be represented as:

$$V_T = \{f_T(x, y, t)\}, \quad x, y, t \in \{0, 1, 2 \ldots\} \tag{2}$$

$f_T(x, y, t)$ is the reduced video pixel location value (x, y). The relationship between the original video and the reduced video can be represented as:

$$f_T(x, y, t) = f_v(rx + k_x, ry + k_y, t) \quad x, y, t \in \{0, 1, 2 \ldots\}, 0 \leq k_x, k_y \leq r - 1 \tag{3}$$

where k_x and k_y are the offsets of the pixel unit, and r is the unit factor. The most crucial phase of visual rhythm detection is detecting scene changes. When a significant scene change occurs, a threshold is set using an adaptive method. An abrupt drop spike is detected when a scene transition occurs, indicating a potential scene change has occurred [41]. The visual rhythm (VR) of the video (V) can be represented as follows:

$$VR = \{f_{VR}(z, t)\} = \{f_T(x(z), y(z), t)\} \tag{4}$$

where $x(z)$ and $y(z)$ are the functions of the independent variable z, the detection of scene changes is shown in Fig. 2. Post-processing is another step that refines the detection of scene changes through additional strategies such as edge detection, histogram analysis, or frame difference. This strategy can support validating and eliminating false alarms and can detect accurate or minimal scene changes. The performance of the scene change detection method is evaluated by comparing the detected scene changes with the ground truth values. Revising the process and testing different feature extraction approaches, thresholds, and post-processing strategies can improve the accuracy and speed of scene change detection.

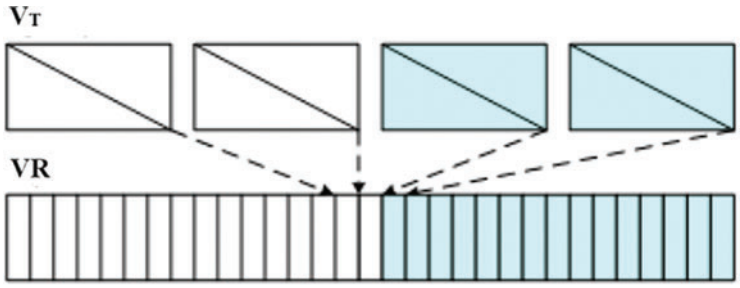

Figure 2: Visual rhythm scene change detection process

It should be noted that scene change detection is still an important area of research in video analytics and that many algorithms exist for scene change detection. A particular implementation specification may differ by the degree of difficulty of the video data and the application requirements.

3.3 Recurrent Deep Architecture for Camera Localization

This section describes a camera localization approach that integrates an indoor camera localization system based on a recurrent neural network with a scene change detection system and explains how to effectively detect scene changes and incorporate them into a localization system to improve the process of camera localization, as shown in Fig. 3. Create fourteen input short videos, including seven with scene changes and seven without, using the visual rhythm algorithm to determine whether there is a scene change in the input image sequence. Next, develop a recurrent neural network architecture for camera pose estimation, which defines various error distributions such as MAE, MSE, RMSE, positional error, and orientational error for each scene. CNN extracts image features from the input image sequence, and LSTM detects pose loss. Refer to one type of RNN as Long-Short-Term Memory (LSTM), which prevents the disappearance of gradients. LSTMs using a technique known as gates can learn long-term dependencies. Several advanced, recurrent architectures, including LSTM [43] and GRU, have addressed the RNN mentioned above. LSTMs are effective in solving sequence-based problems with long-term constraints.

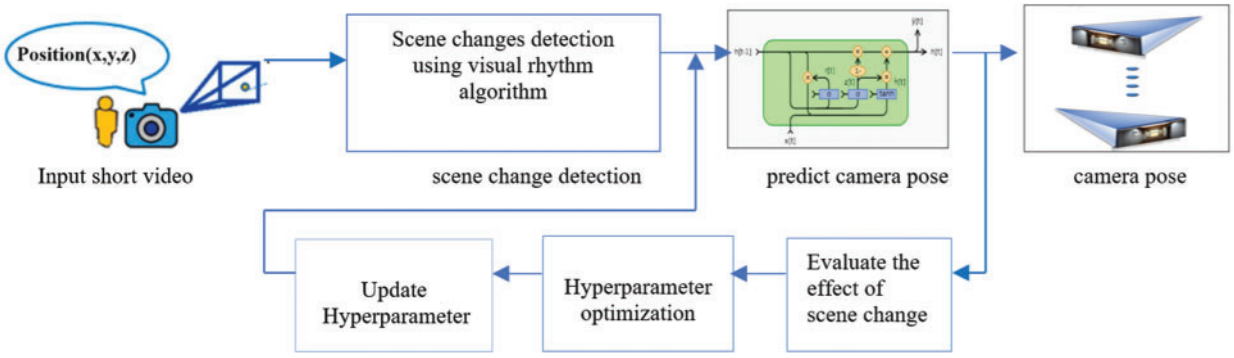

Figure 3: Indoor camera localization process for the proposed approach

Indoor camera localization describes several phases to accurately predict the camera pose from image sequences or videos. First, a short video is provided as input, and the visual rhythm algorithm analyzes the change in the scene. This algorithm successfully identifies the shift from one scene to another, making it easier to perform further analysis. Subsequently, a recurrent neural network (RNN)

predicts the camera pose using information extracted from the input video. Next, it investigated the influence of scene changes on the accuracy and speed of camera localization. The evaluation process helps to improve the functionality of the overall camera localization system. A hyperparameter optimization technique was applied to improve system performance and ensure optimal results. By repeatedly updating the hyperparameters, the camera localization process continuously enhances and increases the ability to predict the camera pose more accurately in different environments.

Here, the measurement technique for localization accuracy is discussed, which mainly uses the Euclidean distance to calculate the pose error (X_{cm}, Y^0) of the proposed camera localization system. The defined thresholds of the three groups, such as best, average, and worst pose error, are (0.25 m, 2^0), (0.5 m, 5^0), and (0.5 cm, 10^0), respectively [44]. The absolute difference between the predicted position, the orientation value, and the actual position and orientation values measures the accuracy of the pose estimation. The position error is calculated based on the Euclidean distance between the expected and the exact original values of the camera. The value for the position error is:

$$P_{error} = \left\| c_{est} - c_{gt} \right\|_2 \tag{5}$$

C_{est} is the estimated value of the camera pose error, and C_{gt} is the actual camera value of the origin [44]. The orientation error $|\alpha|$ is measured by convention:

$$2cos\,|\alpha| = trace(R_{gt}^{-1} R_{est}) \tag{6}$$

Here, $|\alpha|$ is the smallest rotation angle required to align the estimated rotation matrix R_{est} with the ground truth value of the rotation R_{gt}. The deep learning-based pose estimation technique is a classification that only estimates the pose of the images [1]. They represent the camera pose mathematically as:

$$loss(l) = \|\hat{x} - x\|_2 + \beta \left\| \hat{q} - \frac{q}{\|q\|} \right\|_2 \tag{7}$$

where $[x, q]$ is the ground truth pose and β is the hyperparameter that determines the relative weight of the orientation and position errors that depend on the training dataset.

3.4 Datasets

This section describes the data set used for the indoor camera localization system and validates the model. Explain the recorded image sequence or short video with and without scene changes in indoor environments. The Microsoft 7-Scenes dataset [45,46] was used to evaluate the proposed recurrent neural network architecture for indoor camera localization. This dataset is commonly used for RGBD analysis and consists of seven different indoor scenes; example images are shown in Fig. 4. The images were captured with a wearable Kinect camera; the image resolution was 460 × 480 pixels. The Kinect fusion technique was used to predict the camera poses more accurately and to establish a link to the ground truth. Each scene consists of multiple sequences, and each image contains three different files, e.g., RGB, depth, and text files, with a 4 × 4 matrix consisting of intrinsic and extrinsic camera parameters. The dataset is challenging because it has some difficulties, such as motion blur, a texture-less surface, and a repetitive structure.

3.5 Image Pre-Processing and Hyperparameter Setup

The proposed deep architecture based on a recurrent neural network is trained with the Microsoft 7-Scenes dataset. Each image in the dataset was resized to 256 × 256 pixels using a combination

of random and centralized cropping methods, and the image intensity was set between −1 and +1. Some components are trained on the ImageNet [47] dataset to use a modified ResNet34 model for the proposed model. The augmentation process is applied to improve the model's accuracy and speed to generalize for different indoor environments. Python 3.10 programming tools and PyTorch 1.2 [48,49] framework is used in developing the experimental process. The Adam optimization method [50] with a learning rate 5×10^{-5} is used. The model was trained with some hyperparameters on a system, such as epoch is 20, the batch size is 64, training dropout is 0.5, the test dropout rate is 0.0, validation frequency is 5, weight dropout is 0.00, the model learning rate is $10 \times 5e^{-05}$, weight initialization value β is 0.9 and γ is 0.3.

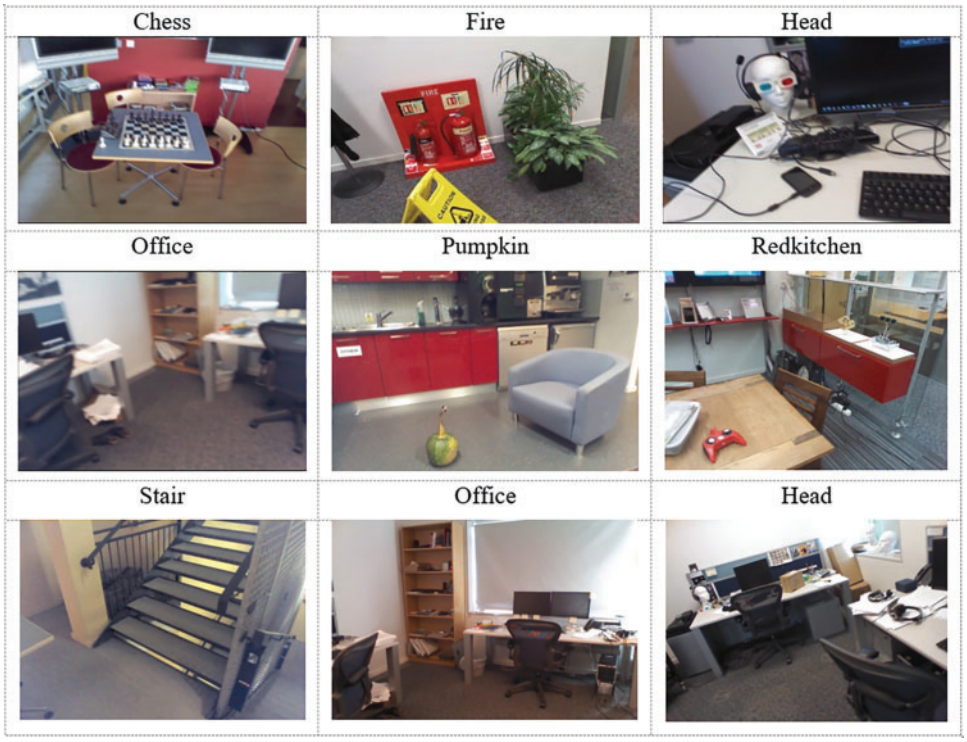

Figure 4: Example images of the dataset

3.6 Evolution Metrics

Evaluate the impact of scene changes on localization performance using more appropriate evaluation metrics about the regression problem. Various error calculation metrics such as mean absolute error (MAE), mean square error (MSE), and root mean square error (RMSE) are the most common metrics. The mathematical equation of the three different error metrics is shown below:

3.6.1 Mean Absolute Error (MAE)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - x_i| \tag{8}$$

where x_i is the calculated value, and y_i is the mean value.

3.6.2 Mean Square Error (MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2 \tag{9}$$

where x_i is the calculated value, and y_i is the mean value.

3.6.3 Root Mean Square Error (RMSE)

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2} \tag{10}$$

where x_i is the calculated value, and y_i is the mean value.

4 Results

4.1 Introduction

This section describes the quantitative analysis of the experiments based on the results obtained. Evaluate the performance of the scene change detection experiments and assessment of the impact on the localization performance of the camera.

4.2 Scene Changes Detection

Determining whether there are scene changes in the input image sequence is very important, as this can affect the performance of camera localization. Here is the visual rhythm algorithm that determines if there is a scene change in an input image sequence and how many scene changes there are; a straightforward "hard cut" is displayed at each scene change. Create a forty-second video with 960 frames for each scene of the seven scenes in the 7-Scenes dataset. Input short video into the visual rhythm algorithm results in a continuous or "hard cut" output. If there is a scene change, a "hard cut" is shown for each scene, and if there is no scene change, the continuous production is shown. When analyzing the given video scene, scene changes were detected in the scenes "Chess", "Heads", "Pumpkin" and "RedKitchen", while no scene changes were detected in the scenes "Fire", "Office" and "Stairs". One scene change was found in the "Chess" scene, three in the "Heads" scene, four in the "Red Kitchen" scene, and five in the "Pumpkin" scene. The scene transition is characterized by a "hard cut", as shown in Fig. 5. Finally, the visual rhythm algorithm successfully identified scene transitions in the input sequence or video, mainly four scenes: "Chess", "Heads", "Pumpkin" and "RedKitchen".

Fig. 5 shows that some scenes have some "hard cut", and some scenes have no "hard cut". A "hard cut" means a scene change has occurred, and no "hard cut" means no scene has been changed. There is no "hard cut" in "Fire", "Office" and "Stairs". On the other hand, there is one in "Chess", three in "Heads", four in "RedKitchen" and five "hard cut" in the "Pumpkin" scene. It means that the image sequence "Fire", "Office" and "Stairs" did not change any scene; on the other hand, there is one cut in "Chess", three in "Heads", four in "Red Kitchen" and five in the "Pumpkin" scene. Now, let us see if the scene changes affect the localization performance. Therefore, train the recurrent deep architecture with two types of datasets: Scene change and without scene change. For "Chess", the combined error for one scene change is 0.0237 and the combined error without scene changes is 0.0253. For "Heads", the combined error for three scene changes is 0.0325 and the combined error without scene changes is 0.0271. For "RedKitchen", the combined error for four scene changes is 0.0309, and the combined error without scene changes is 0.0250. For "Pumpkin", the combined error is 0.0330 for five scene

changes and 0.0250 without scene changes. It shows that the more often the scene is changed, the more significant the difference in error. The scene changes of "Chess", "Heads", "Red Kitchen", and "Pumpkin" are one, three, four, and five, respectively, and the error difference is 0.0016, 0.0054, 0.0058, and 0.0080, respectively. The above data shows that the more frequently the scene changes, the higher the error, which affects the camera's localization performance. Table 1 shows the number of scene changes from Fig. 5 and their effect on localization performance. Table 1 also shows that as the scene changes, the localization error increases, which is evident from the difference in error.

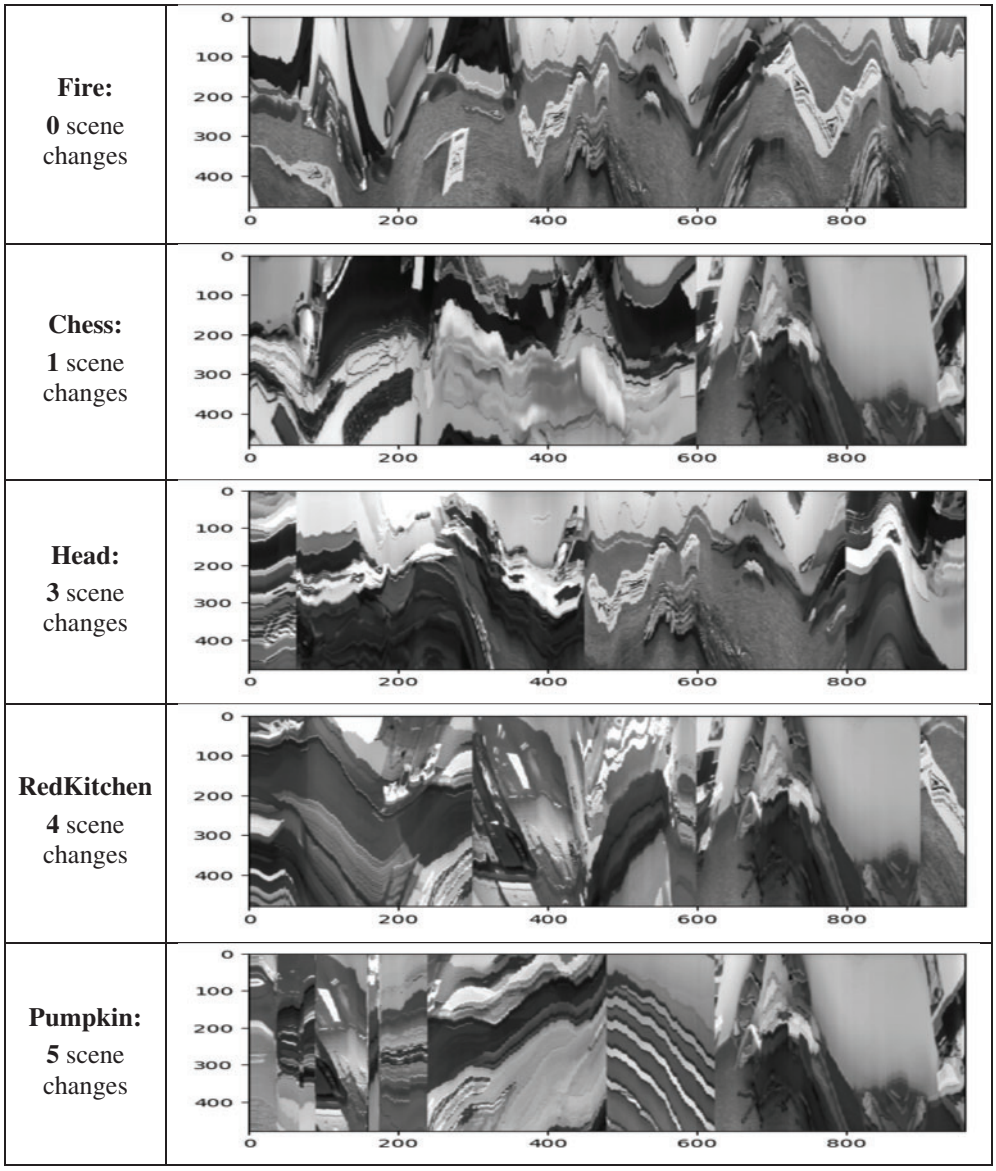

Figure 5: Scene change identification from dataset

Table 1: The scene changes impact camera localization errors

Scenes	No. of scene change	Combined error with scene change	Combined error without scene change	Error difference
Chess	1	0.0253	0.0237	0.0016
Heads	3	0.0325	0.0271	0.0054
RedKitchen	4	0.0309	0.0250	0.0058
Pumpkin	5	0.0330	0.0250	0.0080

4.3 Analyze the Effect of Scene Changes

Here, use seven forty-second short videos with scene changes and seven forty-second short videos of seven scenes without scene changes as input. Seven forty-second videos with and without scene changes should be used to evaluate the proposed scene change detection approach, where each forty-second short video consists of 960 frames of the dataset. The results obtained were evaluated in two different ways. The first option is training using a data set. It involves comparing the seven scenes in a data set with and without scene changes. Another evaluation process result is the calculation of the average position and orientation errors with and without scene changes.

In the first method, train the deep learning approach with each scene input with and without scene change and determine each scene's AME, MSE, and RMSE. When the scene does not change, the error distribution is shown in Fig. 6, and when the scene changes, express the error distribution in Fig. 7. Then, calculate a combined distribution by the AME, MSE, and RMSE standard deviation. Fig. 8 shows the results with and without scene change. From Fig. 8, it is easy to see how much a scene change in the image sequence affects the performance.

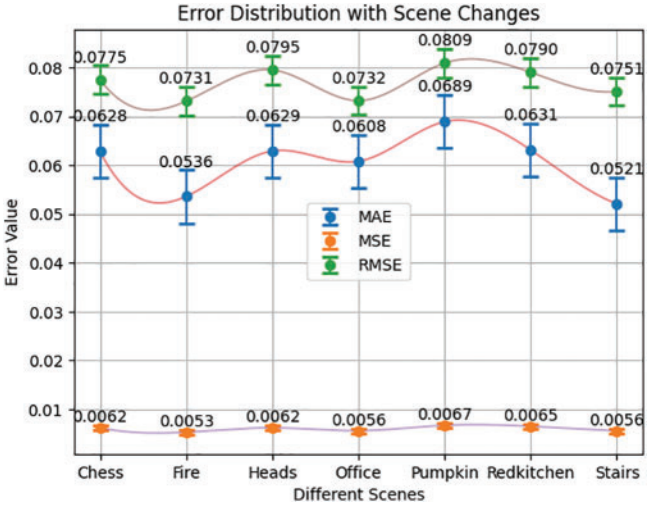

Figure 6: Error distribution of with scene changes

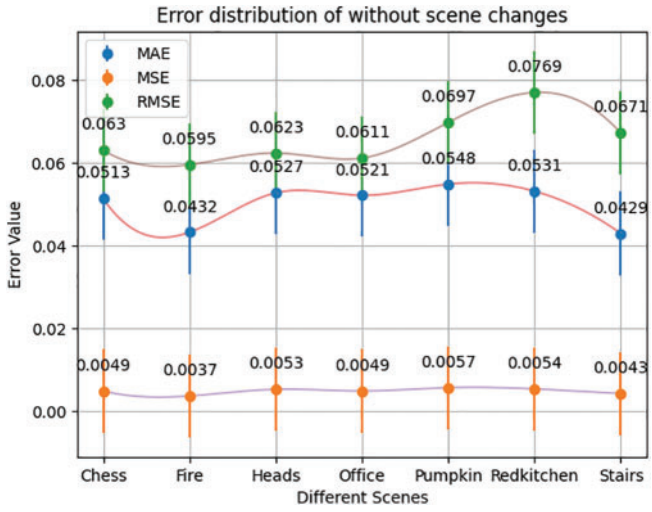

Figure 7: Error distribution of without scene changes

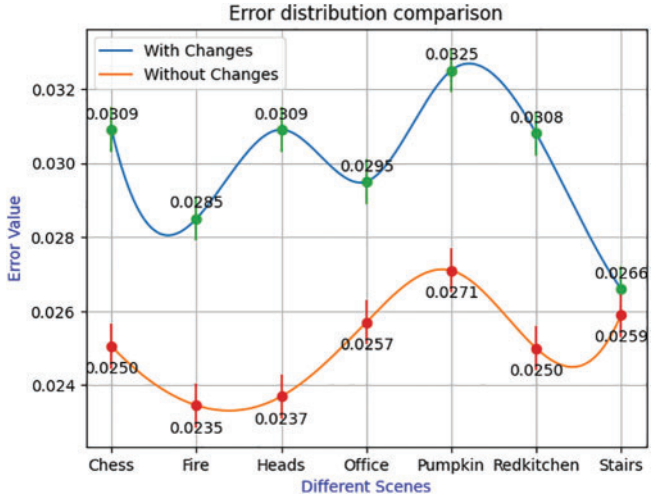

Figure 8: Combined error distribution comparison of scene changes and without scene changes

The result analysis in Fig. 6 shows that the error rate varies between the scenes. Among the seven scenes, the "Stairs" scene has the lowest error with an MAE of 0.0521, an MSE of 0.0056, and an RMSE of 0.0751. The two scenes, "Fire" and "Chess," have comparatively lower error values. "Pumpkin" shows the maximum error distribution with an MAE of 0.0689, an MSE of 0.0067, and an RMSE of 0.0809. The other scenes, such as "Heads", "Office", and "RedKitchen", show comparable error values.

Fig. 7 compares the different error distributions for the various scenes using the 7-Scenes data set. The scenes "Fire" and "Stairs" have relatively low errors, while "RedKitchen" has a higher error than the other scenes. The "Fire" scene has the lowest MAE error at 0.0432, while the MAE error of the "Stairs" scene (0.0429) is very close. The highest MAE error is recorded for the "Pumpkin" scene with an error of 0.0548. "RedKitchen" has the highest MSE error at 0.0057, while "Fire" has the lowest

MSE error at 0.0037. The "Fire" scene has the lowest RMSE error at 0.0595, while "RedKitchen" has the highest RMSE error at 0.0769.

Fig. 8 shows the combined error distribution for the seven scenes and compares the errors for both cases with and without scene changes. For example, in the scene "Chess", the combined error distribution with scene change is 0.0309, and without scene change is a lower error of 0.02505. It shows that the transition to the scene "Chess" from the previous scene affects the overall error of "Chess". Similarly, Fig. 8 shows errors in other scenes, such as "Fire", "Heads", "Office", "Pumpkin", "RedKitchen" and "Stairs", both with and without scene changes. Fig. 8 clearly shows that the combined error values with and without scene change are very different for each of the six scenes. Only in the "Stairs" scene is the difference in error marginal. When the input image sequence changes, it affects the localization performance of the camera and degrades the localization accuracy.

Now shown in a second way how changes in the image sequences affect localization performance. Train a camera localization model with fourteen input videos, seven with and seven without scene changes. Obtain position and orientation error values from the localization model as output. The position error values obtained in both cases are shown in Fig. 9, and the orientation error values in Fig. 10.

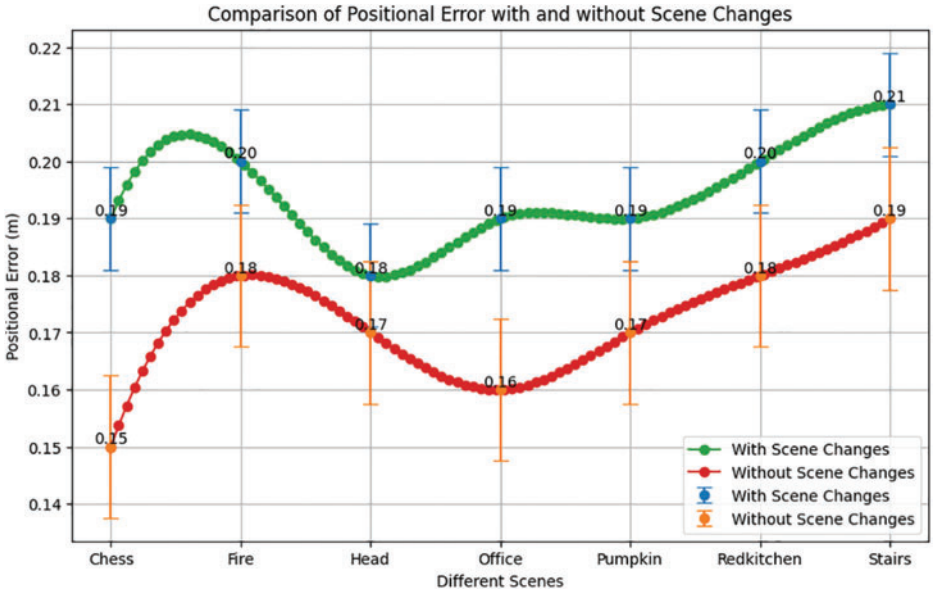

Figure 9: Positional error comparison of with and without scene changes

The positional error obtained by training the recurrent neural network model with scene-changed and without scene-changed datasets is shown in Fig. 9. In case of scene change, the positional error is between 0.18 and 0.21 m. In this case, the minimum positional error is 0.18 m for the "Head" scene, and the positional maximum error is 0.21 m for the "Stairs" scene. The positional errors of the other five scenes are pretty close to each other. On the other hand, the positional error for the scene without scene change is between 0.15 and 0.19 m. In this case, the lowest positional error is 0.15 m for the "Chess" scene, and the highest is 0.19 m for the "Stairs" scene. The positional errors of the other five scenes are pretty close to each other. Fig. 9 clearly shows that the positional error is comparatively lower for each scene, which is relatively minor if the scenes in the input image sequences do not change.

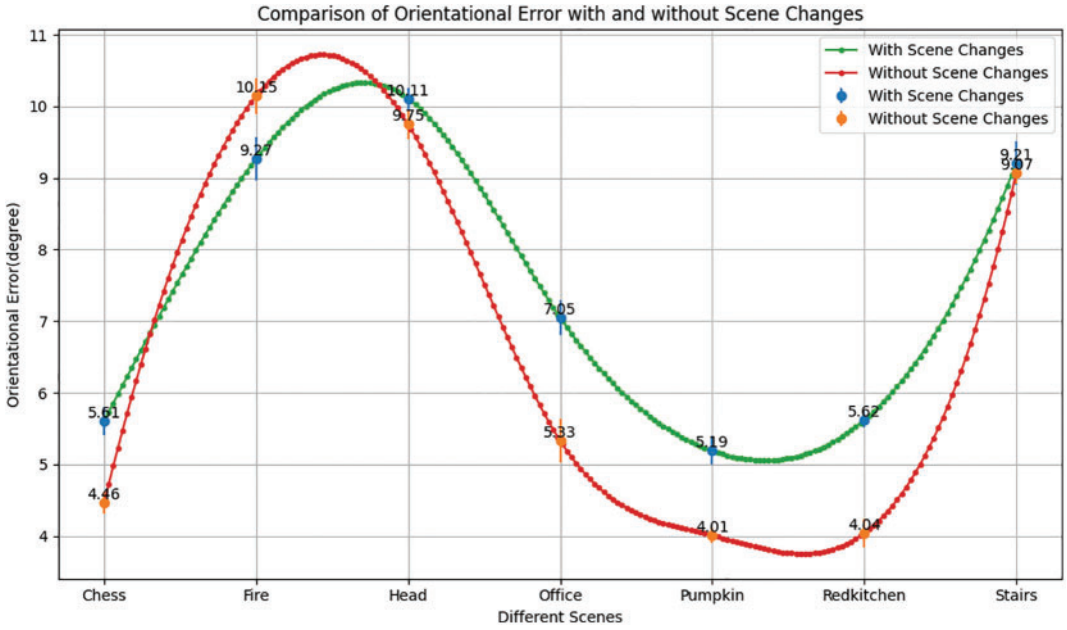

Figure 10: Orientational error comparison of with and without scene changes

The orientational error obtained by training the recurrent neural network model with scene change and without scene change is shown in Fig. 10. The orientational error with scene change is between 5.61^0 and 10.11^0. In this case, the lowest orientational error is 5.61^0 for the "Chess" scene, and the highest is 10.11^0 for the "Head" scene. On the other hand, the orientational error in the case without scene change is between 4.01^0 and 10.15^0. In this case, the minimum orientation error is 4.01^0 for the "Pumpkin" scene, and the positional maximum error is 10.15^0 for the "Fire" scene. In the case of six scenes, the orientation error is lower if the scene does not change within the image sequences; the "Fire" scene is an exception. For the "Fire" scenes, the orientational error is 9.27^0 if the scene is changed and 10.15^0 if the scene is not changed; such deviations can sometimes occur if the input image contains motion blur, light changes, viewing angles, and other errors.

According to the above discussion in Figs. 9 and 10, it can be seen that the pose error increases as the scene changes within the input image sequences. The more frequently the scene changes within an image sequence, the greater the pose loss, resulting in a higher error and, therefore, lower performance.

4.4 Compare with Existing Researches

Table 2 compares the results of the proposed recurrent deep architecture with state-of-the-art research. It shows seven scenes' individual and average pose errors from the proposed study and state-of-the-art research using the 7-Scenes dataset. Recent research results include PoseNet, LSTM-PoseNet, MapNet, AtLoc, EpiLoc, and CGAPoseNet. First, examine the effects of camera pose error in scene changes on simulation results and compare the results obtained with deep architecture with those of existing state-of-the-art research.

Table 2: Pose errors in existing research and proposed architecture

Researches	Chess	Fire	Head	Office	Pumpkin	RedKitchen	Stairs	Average pose error
PoseNet [1]	0.32 m, 4.06^0	0.47 m, 7.33^0	0.29 m, 12.0^0	0.48 m, 6.00^0	0.47 m, 4.21^0	0.59 m, 4.32^0	0.47 m, 6.93^0	0.45 m, 9.84^0
LSTM-PoseNet [51]	0.24 m, 5.77^0	0.34 m, 11.9^0	0.21 m, 13.7^0	0.31 m, 8.07^0	0.33 m, 7.0^0	0.37 m, 8.82^0	0.41 m, 13.7^0	0.31 m, 9.85^0
MapNet [22]	0.08 m, 3.25^0	0.27 m, 11.69^0	0.18 m, 13.25^0	0.17 m, 5.15^0	0.22 m, 4.02^0	0.23 m, 4.93^0	0.30 m, 12.08^0	0.21 m, 7.78^0
AtLoc [52]	0.10 m, 4.07^0	0.25 m, 11.4^0	0.16 m, 11.8^0	0.17 m, 5.34^0	0.21 m, 4.37^0	0.23 m, 5.42^0	0.26 m, 10.5^0	0.20 m, 7.56^0
EpiLoc [40]	0.07 m, 2.71°	0.24 m, 9.18°	0.14 m, 12.6°	0.18 m, 4.45°	0.18 m, 3.22°	0.23 m, 4.60°	0.24 m, 11.0°	0.18 m, 6.82°
CGAPoseNet [39]	0.26 m, 6.34°	0.28 m, 10.3°	0.17 m, 7.98°	0.26 m, 7.23°	0.22 m, 5.18°	0.55 m, 16.7°	0.17 m, 12.0°	0.27 m, 9.39°
Proposed (Scene changes)	0.19 m, 5.61^0	0.20 m, 9.27^0	0.18 m, 10.11^0	0.19 m, 7.05^0	0.19 m, 5.19^0	0.20 m, 5.62^0	0.21 m, 9.21^0	0.19 m, 7.43^0
Proposed (Without scene changes)	0.15 m, 4.46^0	0.18 m, 10.15^0	0.17 m, 9.75^0	0.16 m, 5.33^0	0.17 m, 4.01^0	0.18 m, 4.04^0	0.19 m, 9.07^0	0.17 m, 6.68^0

Train the recurrent deep architecture twice, once when the scene changes in the image sequences and the other time when the scene does not change in the image sequences. It can be seen that the pose error increases when the scene changes in an image sequence. Here, it can be seen that the positional error is 0.19 m when the scene changes and the average positional error is 0.17 m when the scene does not change. The positional error decreases when the scene does not change. Again, the average orientation error is 7.73^0 when the scene changes in the image sequences and 6.68^0 when the scene does not change. The orientational error is also relatively high here when the scene changes. The difference between a scene change in an image sequence and no scene change is (0.02 m, 1.05^0); from this result, it can be concluded that a scene change in an image sequence affects the localization performance of the camera and degrades the performance.

Table 2 shows that the average positional error in the available tests is between 0.18 and 45 m; the highest average positional error is 0.45 m for PoseNet, and the lowest average positional error is 0.18 m for EpiLoc. The average positional error achieved in our test is 0.17 m, lower than in all the studies listed in the table. Regarding orientational error, the highest average among the existing researchers is 9.85^0 degrees for LSTM-PoseNet, and the lowest is 6.82^0 degrees for EpiLoc. However, the average orientational error obtained in our experiment without scene change is 6.68^0 degrees, which is lower than the error of EpiLoc. From the discussion of the above results, it can be seen that the results of our recurrent deep architecture (0.17 m, 6.68^0) without scene change in the image sequence have the lowest pose error, which is very important for applying more accurate camera localization.

The 7-Scenes datasets used to determine the localization of indoor cameras have many complex problems, such as motion blur, illumination variations, viewing angles, and so on. Image sequences with more of these problems have higher pose errors. It can be seen from Table 2 that the orientation error for "Head" (9.75^0), "Fire" (10.15^0), and "Stairs" (9.07^0) is much higher than other scenes.

5 Discussion

This research study proposes a new recurrent deep architecture for an indoor camera localization system that performs camera localization in real time and evaluates the effects of scene changes during image sequences. The proposed system is trained with an annotated image dataset with corresponding baseline values. The recurrent deep architecture achieves high pose accuracy even if the scene changes during the sequence. The predicted position error is less than 0.20 m, and the orientation error is

less than 7.0^0; analyzed the effects of scene changes and found that small scene changes, such as moving objects, can cover the system. However, more significant changes in the scene, such as lighting conditions and points, affect the localization performance. The results showed the effectiveness of the proposed system, which performs real-time camera localization and handles scene changes during the sequence. The proposed recurrent deep architecture is essential for various applications, such as indoor robot navigation, augmented reality, and autonomous driving.

6 Conclusion

In summary, the recurrent neural deep architecture for a real-time indoor camera localization system evaluates the impact of detecting scene changes during image sequences. Microsoft 7-Senses dataset is used to create customized input datasets for two cases with and without scene changes to train and evaluate the approach. The proposed approach performed well and can accurately determine the impact of scene changes in the input image sequence on performance. It also ensured the flexibility and reliability of the system by implementing it on devices and evaluating it in real-life situations. Overall, the proposed RNN-based camera localization system is helpful for various robotics applications, such as indoor robot navigation, augmented reality, and indoor surveillance systems. Future work could investigate the integration of multiple cameras into an indoor localization system. The proposed recurrent deep architecture provides a real-time camera localization system that continuously improves performance and minimizes computational costs. Future work could investigate how to optimize the localization performance, e.g., through hardware improvements, enhanced feature extractions, etc. Future directions could explore the application of real-world environments, such as indoor robotic navigation systems, augmented reality, and self-driving cars.

Acknowledgement: The authors are grateful for support from the Faculty of Computing, Universiti Teknologi Malaysia, Malaysia.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: Study conception and design: Muhammad S. Alam, Farhan B. Mohamed; data collection: Faruk Ahmed and AKM B. Hossain; analysis and interpretation of results: Farhan B. Mohamed and Ali Salamat; draft manuscript preparation: Muhammad S. Alam and Faruk Ahmed.

Availability of Data and Materials: Not applicable.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] A. Kendall, M. Grimes, and R. Cipolla, "PoseNet: A convolutional network for real-time 6-DOF camera relocalization," in *Proc. IEEE Int. Conf. Comput. Vis.*, Santiago, Chile, 2015, pp. 2938–2946. doi: 10.1109/ICCV.2015.336.

[2] A. Atghaei, E. Rahnama, K. Azimi, and H. Shahbazi, "Industrial scene change detection using deep convolutional neural networks," arXiv preprint arXiv:2212.14278, 2022.

[3] S. Zhang, W. Hu, W. Guo, and C. Liu, "Neural-network-based pose estimation during noncooperative spacecraft rendezvous using point cloud," *J. Aerosp. Inf. Syst.*, vol. 20, no. 2, pp. 462–472, 2023. doi: 10.2514/1.I011179.

[4] T. Ji, C. Wang, and L. Xie, "Towards real-time semantic RGB-D SLAM in dynamic environments," in *2021 IEEE Int. Conf. Robot. Autom. (ICRA)*, Xi'an, China, 2021, pp. 11175–11181. doi: 10.1109/ICRA48506.2021.9561743.
[5] M. S. Alam, F. B. Mohamed, and A. K. M. B. Hossain, "Self-localization of guide robots through image classification," *Baghdad Sci. J.*, vol. 21, no. 2, pp. 832, 2024. doi: 10.21123/bsj.2024.9648.
[6] F. Zhou, L. Zhang, C. Deng, and X. Fan, "Improved point-line feature based visual SLAM method for complex environments," *Sensors*, vol. 21, no. 13, p. 4604, 2021. doi: 10.3390/s21134604.
[7] D. Jiang, G. Li, C. Tan, L. Huang, Y. Sun and J. Kong, "Semantic segmentation for multiscale target based on object recognition using the improved Faster-RCNN model," *Future Gener. Comput. Syst.*, vol. 123, pp. 94–104, 2021. doi: 10.1016/j.future.2021.04.019.
[8] B. A. Labinghisa and D. M. Lee, "Indoor localization system using deep learning based scene recognition," *Multimed. Tools Appl.*, vol. 81, no. 20, pp. 28405–28429, 2022. doi: 10.1007/s11042-022-12481-3.
[9] Y. Qin, T. Mei, Z. Gao, Z. Lin, W. Song and X. Zhao, "RGB-D SLAM in dynamic environments with multi-level semantic mapping," *J. Intell. Robot Syst.*, vol. 105, no. 4, p. 90, 2022. doi: 10.1007/s10846-022-01697-y.
[10] F. Alhamazani, Y. K. Lai, and P. L. Rosin, "3DCascade-GAN: Shape completion from single-view depth images," *Comput. Graph*, vol. 115, pp. 412–422, 2023. doi: 10.1016/j.cag.2023.07.033.
[11] K. J. Singh, D. S. Kapoor, K. Thakur, A. Sharma, and X. Z. Gao, "Computer-vision based object detection and recognition for service robot in indoor environment," *Comput. Mater. Contin.*, vol. 72, no. 1, pp. 197–213, 2022. doi: 10.32604/cmc.2022.022989.
[12] J. Braun, J. Mendes, A. I. Pereira, J. Lima, and P. Costa, "Object detection for indoor localization system," in *Second Int. Conf., OL2A 2022*, Póvoa de Varzim, Portugal, Springer, Oct. 24–25, 2023, pp. 788–803. doi: 10.1007/978-3-031-23236-7_54.
[13] S. Y. Gadre, K. Ehsani, S. Song, and R. Mottaghi, "Continuous scene representations for embodied AI," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, New Orleans, LA, USA, 2022, pp. 14849–14859. doi: 10.1109/CVPR52688.2022.01443.
[14] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004. doi: 10.1023/B:VISI.0000029664.99615.94.
[15] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *2011 Int. Conf. Comput. Vis.*, Barcelona, Spain, IEEE, 2011, pp. 2564–2571. doi: 10.1109/ICCV.2011.6126544.
[16] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, 2017. doi: 10.1109/TRO.2017.2705103.
[17] N. Li and H. Ai, "EfiLoc: Large-scale visual indoor localization with efficient correlation between sparse features and 3D points," *Vis. Comput.*, vol. 38, no. 6, pp. 2091–2106, 2022. doi: 10.1007/s00371-021-02270-8.
[18] P. Alliez *et al.*, "Indoor localization and mapping: Towards tracking resilience through a multi-slam approach," in *2020 28th Mediterranean Conf. Cont. Automat. (MED)*, Saint-Raphaël, France, IEEE, 2020, pp. 465–470. doi: 10.1109/MED48518.2020.9182794.
[19] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *13th Eur. Conf.*, Zurich, Switzerland, Springer, Sep. 6–12, 2014, pp. 834–849. doi: 10.1007/978-3-319-10605-2_54.
[20] Z. Bao, G. Fu, L. Duan, and C. Xiao, "Interactive lighting editing system for single indoor low-light scene images with corresponding depth maps," *Vis. Inform.*, vol. 6, no. 4, pp. 90–99, 2022. doi: 10.1016/j.visinf.2022.08.001.
[21] Y. Liu and J. Miura, "RDMO-SLAM: Real-time visual SLAM for dynamic environments using semantic label prediction with optical flow," *IEEE Access*, vol. 9, pp. 106981–106997, 2021. doi: 10.1109/ACCESS.2021.3100426.
[22] S. Brahmbhatt, J. Gu, K. Kim, J. Hays, and J. Kautz, "Geometry-aware learning of maps for camera localization," in *2018 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, 2018, pp. 2616–2625. doi: 10.1109/CVPR.2018.00277.

[23] J. McDonald, M. Kaess, C. Cadena, J. Neira, and J. J. Leonard, "Real-time 6-DOF multi-session visual SLAM over large-scale environments," *Rob Auton. Syst.*, vol. 61, no. 10, pp. 1144–1158, 2013. doi: 10.1016/j.robot.2012.08.008.

[24] S. Wang, J. Yue, Y. Dong, R. Shen, and X. Zhang, "Real-time omnidirectional visual SLAM with semi-dense mapping," in *2018 IEEE Intell. Veh. Symp. (IV)*, Changshu, China, IEEE, 2018, pp. 695–700. doi: 10.1109/IVS.2018.8500460.

[25] P. Y. Tseng, J. J. Lin, Y. C. Chan, and A. Y. Chen, "Real-time indoor localization with visual SLAM for in-building emergency response," *Autom Constr.*, vol. 140, pp. 104319, 2022. doi: 10.1016/j.autcon.2022.104319.

[26] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 611–625, 2017. doi: 10.1109/TPAMI.2017.2658577.

[27] P. Zhang, M. Zhang, and J. Liu, "Real-time HD map change detection for crowdsourcing update based on mid-to-high-end sensors," *Sensors*, vol. 21, no. 7, pp. 2477, 2021. doi: 10.3390/s21072477.

[28] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *2007 6th IEEE and ACM Int. Symp. Mix Augment. Real.*, Nara, Japan, IEEE, 2007, pp. 225–234. doi: 10.1109/ISMAR.2007.4538852.

[29] R. Clark, S. Wang, H. Wen, A. Markham, and N. Trigoni, "VINet: Visual-inertial odometry as a sequence-to-sequence learning problem," in *Proc. AAAI Conf. Artif. Intell.*, San Francisco, CA, USA, 2017. doi: 10.1609/aaai.v31i1.11215.

[30] M. S. Alam, A. K. M. B. Hossain, and F. B. Mohamed, "Performance evaluation of recurrent neural networks applied to indoor camera localization," *Int. J. Emerg. Technol. Adv. Eng.*, vol. 12, no. 8, pp. 116–124, 2022. doi: 10.46338/ijetae0822_15.

[31] M. S. Alam, F. B. Mohamed, A. Selamat, and A. B. Hossain, "A review of recurrent neural network based camera localization for indoor environments," *IEEE Access*, vol. 11, pp. 43985–44009, 2023. doi: 10.1109/ACCESS.2023.3272479.

[32] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Honolulu, HI, USA, 2017, pp. 1851–1858. doi: 10.1109/CVPR.2017.700.

[33] H. Hakim, Z. Alhakeem, and S. Al-Darraji, "Goal location prediction based on deep learning using RGB-D camera," *Bull. Electr. Eng. Inform.*, vol. 10, no. 5, pp. 2811–2820, 2021. doi: 10.11591/eei.v10i5.3170.

[34] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, 2017. doi: 10.1109/TPAMI.2017.2699184.

[35] V. Guizilini, K. H. Lee, R. Ambruş, and A. Gaidon, "Learning optical flow, depth, and scene flow without real-world labels," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 3491–3498, 2022. doi: 10.1109/LRA.2022.3145057.

[36] Z. Xiao, C. Chen, S. Yang, and W. Wei, "EffLoc: Lightweight vision transformer for efficient 6-DOF camera relocalization," arXiv preprint arXiv:2402.13537, 2024.

[37] Y. Cho, S. Eum, J. Im, Z. Ali, H. Choo and U. Park, "Deep photo-geometric loss for relative camera pose estimation," *IEEE Access*, vol. 11, pp. 130319–130328, 2023. doi: 10.1109/ACCESS.2023.3325661.

[38] A. Abozeid, A. I. Taloba, R. M. Abd El-Aziz, A. F. Alwaghid, M. Salem and A. Elhadad, "An efficient indoor localization based on deep attention learning model," *Comput. Syst. Sci. Eng.*, vol. 46, no. 2, pp. 2637–2650, 2023. doi: 10.32604/csse.2023.037761.

[39] A. Pepe and J. Lasenby, "CGA-PoseNet: Camera pose regression via a 1D-Up approach to conformal geometric algebra," arXiv preprint arXiv:2302.05211, 2023.

[40] L. Xu, T. Guan, Y. Luo, Y. Wang, Z. Chen and W. Liu, "EpiLoc: Deep camera localization under epipolar constraint," *KSII Trans. Intern. & Inform. Syst.*, vol. 16, no. 6, pp. 2044–2059, 2022. doi: 10.3837/tiis.2022.06.014.

[41] H. Kim, J. Lee, J. H. Yang, S. Sull, W. M. Kim and S. M. H. Song, "Visual rhythm and shot verification," *Multimed. Tools Appl.*, vol. 15, pp. 227–245, 2001. doi: 10.1023/A:1012452131892.

[42] K. D. Seo, S. J. Park, J. S. Kim, and S. M. H. Song, "Automatic shot-change detection algorithm based on visual rhythm extraction," in *Int. Conf. Image Anal. Recognit.*, Berlin, Heidelberg, Springer, 2006, pp. 709–720. doi: 10.1007/11867586_65.

[43] M. Brattinga, "LSTM-based indoor localization with transfer learning," Bachelor thesis, Dept. of Comp. Sc., Univ. of Twente, AE Enschede, Netherland, 2022.

[44] J. Garsten, "6DOF camera localization through a short image sequence," Masters thesis, Dept. of Elect. Eng., Chalmers Univ. of Technology, Gothenburg, Sweden, 2020.

[45] B. Glocker, S. Izadi, J. Shotton, and A. Criminisi, "Real-time RGB-D camera relocalization," in *2013 IEEE Int. Symp. Mixed Augment. Real. (ISMAR)*, Adelaide, SA, Australia, IEEE, 2013, pp. 173–179. doi: 10.1109/ISMAR.2013.6671777.

[46] J. Shotton *et al.*, "Scene coordinate regression forests for camera relocalization in RGB-D images," in *IEEE Conf. Comput. Vis. Pattern Recognit.*, Portland, OR, USA, 2013, pp. 2930–2937. doi: 10.1109/CVPR.2013.377.

[47] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conf. Comput. Vis. Pattern Recognit.*, Miami, FL, USA, IEEE, 2009, pp. 248–255. doi: 10.1109/CVPR.2009.5206848.

[48] S. Imambi, K. B. Prakash, and G. R. Kanagachidambaresan, "PyTorch," in *Programming with TensorFlow*, 2021, pp. 87–104. doi: 10.1007/978-3-030-57077-4_10.

[49] A. Paszke, S. Gross, S. Chintala, and G. Chanan, "Pytorch: Tensors and dynamic neural networks in python with strong GPU acceleration," 2017. Accessed: Feb. 15, 2024. [Online]. Available: https://github.com/pytorch/pytorch.

[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.980, 2014.

[51] F. Walch, C. Hazirbas, L. Leal-Taixé, T. Sattler, S. Hilsenbeck and D. Cremers, "Image-based localization using LSTMs for structured feature correlation," in *IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, 2017, pp. 627–637. doi: 10.1109/ICCV.2017.75.

[52] B. Wang, C. Chen, C. X. Lu, P. Zhao, N. Trigoni and A. Markham, "AtLoc: Attention guided camera localization," *Proc. AAAI Conf. Artif. Int.*, vol. 34, no. 6, pp. 10393–10401, 2020. doi: 10.1609/aaai.v34i06.6608.