Engineering and Technology Journal
Journal homepage: engtechjournal.org

Outdoor Localization in Mobile Robot with 3D LiDAR Based on Principal Component Analysis and K-Nearest Neighbors Algorithm

Hanan A. Atiyah[a]*, Mohammed Y. Hassan [b]

[a] University of Technology-Iraq, Baghdad, Iraq, cse.19.03@grad.uotechnology.edu.iq

[b] University of Technology-Iraq, Baghdad, Iraq, 60003@uotechnology.edu.iq

*Corresponding author.

Submitted: 01/03/2021 Accepted: 14/03/2021 Published: 25/06/2021

K E Y W O R D S

Convolution Neural Network, IHS, K-Nearest Neighbors algorithm, LiDAR, Mobile robot localization, Principal Component Analysis.

A B S T R A C T

Localization is one of the potential challenges for a mobile robot. Due to the inaccuracy of GPS systems in determining the location of the moving robot alongside weathering effects on sensors such as RGBs (e.g. rain and light-sensitivity) . This paper aims to improve the localization of mobile robots by combining the 3D LiDAR data with RGB-D images using deep learning algorithms. The proposed approach is to design an outdoor localization system. It is divided into three stages. The first stage is the training stage where 3D LiDAR scans the city and then reduces the dimensions of 3D LiDAR data to 2.5D image. This is based on PCA method where these data are used as training data. The second stage is the testing data stage. RGB and depth image in IHS method are combined to generate 2.5D fusion image. The training and testing of these datasets are based on using Convolution Neural Network. The third stage consists of using the K-Nearest Neighbor algorithm. This is the classification stage to get high accuracy and reduces the training time. The experimental results obtained prove the superiorly of the proposed approach with accuracy up to 97.52%, Mean Square of Error of 0.057568, and Mean error in distance equals 0.804 meters.

How to cite this article: H. A. Atiyah, and M. Y. Hassan, "Outdoor Localization in Mobile Robot with 3D LiDAR-based on Principal Component Analysis and K-Nearest Neighbors algorithm," Engineering and Technology Journal, Vol. 39, No. 06, pp. 965-976, 2021.
DOI: https://doi.org/10.30684/etj.v39i6.2032

1. INTRODUCTION

Mobile robots have an important part in many areas of situations [1]. Mobile robots sometimes work with insufficient knowledge about the areas. Most studies had been conducted on the environmental study as the robot works through different forms of sensors combined on the robot [2]. Global Positioning System GPS is the most commonly used localization solution, though it suffers

from some limitations, for example, the multipath influence, delay, restricting its use in urban and poor GPS signal due to large buildings [3].

The Simultaneous Localization And Mapping (SLAM). It is commonly used to calculate the location of moving mobile robots at the same time to generate a map of the local area [4]. The researchers have the attention of SLAM and achieved many practical results. Zhang et al. combined a Deep Learning (DL) Algorithm's object detection unit and localization with RGB-D SLAM [5]. But it is always pricey to produce a large-scale outdoor map, however, try to find a simple way [6].

In the field of Machine Learning (ML), the task of enhancing the efficiency of robotics through the integration of (ML) technology has generated new challenges. Interest and efforts in designing machine learning approaches for robotics systems focused on computer vision have grown in recent years for example Keisuke et al. proposed an algorithm based on distance measurement that uses only odometer measurement of distances computed from robot movements using Convolution Neural Network (CNN) algorithm [7]. Junior et al. proposed the solution of using the Internet of Things (IoT) to build a system capable of carrying out this online operation. For the robot to navigate by computer vision, a topological map, CNN, and machine learning methods are used [8].

The Light Detection And Ranging LiDAR as an active sensor and invariant to light. A typical 3D LiDAR, on the other hand, can obtain environmental information with 30 (±15) º in the vertical direction and 360º in the horizontal field of view (FOV) at a scanning rate of around 10 Hz. In an area with long ranges [9], high resolution enables the LiDAR to collect a huge number of good information. In robot systems, these benefits make LiDAR widely used. Li et al. Presented a technique to increase the precision of pose prediction of 3D point clouds from LiDAR by accurately segmenting the surface point and point cloud [9]. Li et al. presented a camera localization workflow based on a highly accurate 3D prior map optimized by RGB-D SLAM method [10]. Kang et al. suggested RGB-D SLAM approach used prior LiDAR point cloud data as a reference for constructing and navigating the indoor 3-D scene [11]. LiDAR-based SLAM approaches have very precise 3D environmental details but also take time to scan and also depend on very simplistic scan-matching approaches that are not very robust [12].

To precise and stable self-localization of mobile robots in the outdoor environment, some approaches fail to correctly identify the location of the mobile robot due to different weather conditions such as rain and snow. In addition to the fact that some sensors such as RGB do not work well in an outdoor environment were very sensitive to light, the LiDAR is seriously disrupted for SLAM problems. So, a proposed a system based on 3D data with Deep learning (DL) algorithm to achieve accuracy and robustness to identify the correct location of the mobile robot.

The proposed method is divided into three stages. Each stage is made up of many operations: training, testing, and classification. In the training stage, the conversion of 3D LiDAR point cloud scan into 2.5D image using Principal Component Analysis (PCA) method is done. The extraction of features from the 2.5D images using Convolution Neural Network (CNN) algorithm is implemented. Then, all features data, point cloud data, and data associated with the pre-processing stage are stored to use in the classification stage. This is to get the ground true position of the mobile robot. In the testing stage, an image fusion is performed by combining two images RGB and Depth (D) image, into a single RGB-D image then using CNN to extract the features from the RGB-D image. In the classification stage, the tested image is classified using the K-Nearest Neighbors algorithm to locate the position of the mobile robot.

The organization of the paper is as follows: In Section 2 presents the material and methods. The proposed method in section 3 is described in detail. Section 4 presents the experimental result and discussion. Finally, the conclusion is obtained in section 5.

2. METHODOLOGY

I. Deep Learning (DL)

In the area of Artificial Intelligence, (DL) is a method that falls within a group of machine learning algorithms that operate on the basic idea of learning. For learning, supervised [13] and unsupervised [14], both models can be used. In (DL), Based on previously studied data, a computerized model executes a particular set of classification or pattern analysis tasks. Therefore, the model must first be trained with a set of structured data. (DL) is mainly used to categorize images, texts, or sounds. The

models work without human intervention and are equivalent, and sometimes better than humans. These models are mostly realized through deep neural networks.

II. Convolution Neural Network (CNN)

Convolutional neural network (CNN) is one of the most amazing types of Artificial Neural Network (ANN) designs. (CNN) is a technology that combines ANNs with modern Deep Learning strategies. This (ANN) has been applied to various image recognition tasks over decades and has attracted the attention of researchers from many countries in recent years, as CNN has shown promising performance in various computer vision and machine learning tasks [15].

A CNN consists of an input and output layer, and multiple hidden layers in between. These layers are generally divided into three types: Convolution (CONV), Pooling (POOL), and Full Connected (FC) [16].

A CNN is made up of many convolutional and subsampling layers, which may be joined by fully connected layers as shown in Figure 1., i.e. after several convolutions and pooling layers, one or more fully connected layers are present. Each stage's inputs and outputs are collections of arrays referred to as feature maps.

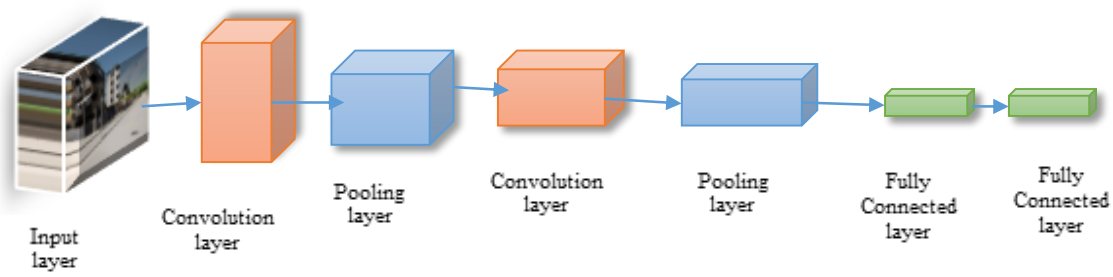


Figure 1: Basic architecture of a Convolutional Neural Network [16]

III. K-Nearest Neighbors (K-NN)

K-NN Classifier is the classification of unlabeled observations by assigning them to the most similar labeled examples. Observation characteristics are collected for both the training and testing dataset. The K-NN algorithm preserves all available data and categorizes new data points based on their similarities. This ensures that as new data appears, the K-NN algorithm will efficiently categorize it into a useful collection.

While the K-NN algorithm can be used for both regression and classification, it is most widely used for classification. For example, consider two categories, Category A and Category B, as well as a new data point X_1 is located [17] So, this data point will lie in which of these categories. A K-NN algorithm is necessary to solve this type of problem. With the help of K-NN, easy identification of the category or class of a particular dataset is obtained. Consider the diagram shown in Figure 2.

First, choose the number of neighbors, so k=5. Next, calculate the Euclidean distance among the data points is done as shown in Figure 3. The distance between two points is known as the Euclidean distance [18].

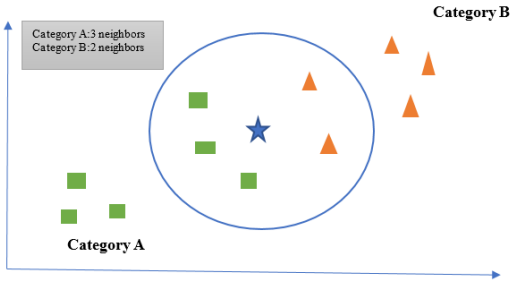


Figure 2: The K-NN principle [17]

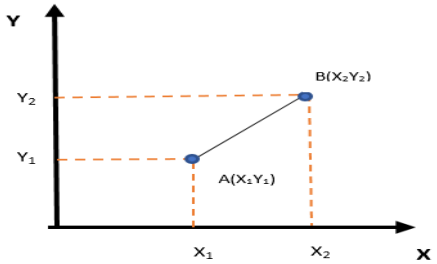


Figure 3: The Euclidean Distance for K-NN [18]

$$\text{Euclidean Distance between A and B} = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2} \quad (1)$$

Using the Euclidean distance formula, the closest neighbors are classified into two categories: category A has three neighbors, and category B has two neighbors. The three nearest neighbors are from category A. Hence, this new data point must belong to category A.

IV. Principal Component Analysis (PCA)

LiDAR is a high-precision sensor that is used in some applications to calculate the distance to its surroundings and present 3D shape as a point cloud where each point has (x, y, z.) coordinates. Due to the harmful effects of atmospheric particles, the return of multiple paths. The point cloud image obtained by LiDAR sensors affords a lot of noise due to diffuse reflection, as well as diverse weather conditions such as rain and snow. To achieve a high-quality point cloud image, this noise must be removed [19].

(PCA) is a dimensional reduction approach for 3D LiDAR data sets that requires transforming a large number of variables into a smaller set that preserves the most information from the larger set.

The covariance matrix is a symmetric matrix of p × p (where p is the number of dimensions) with the covariance associated with all valid pairs of the original variables as entries. For example, the covariance matrix is a 3 × 3 matrices of this form: for a 3-dimensional data set of 3 variables x, y, and z [20]

$$\begin{bmatrix} \mathrm{Cov}(x,x) & \mathrm{Cov}(x,y) & \mathrm{Cov}(x,z) \\ \mathrm{Cov}(y,x) & \mathrm{Cov}(y,y) & \mathrm{Cov}(y,z) \\ \mathrm{Cov}(z,x) & \mathrm{Cov}(z,y) & \mathrm{Cov}(z,z) \end{bmatrix} \quad (2)$$

As the variance of a variable is its variance with itself (Cov(a, a)=Var(a)), basically have the variances of each original in the main diagonal vector (top left to bottom right). (Cov(a,b)=Cov(b,a) since the covariance is commutative. For the main diagonal, the covariance matrix entries are symmetric, meaning that the upper and lower triangular parts are identical.

The linear algebra principles are needed to calculate from the covariance matrix to evaluate the key components of the data, eigenvectors, and eigenvalues. The first main components (Y_1) are given by the linear combination of variables X_1, X_2,...,X_p [21]:

$$Y_1 = a_{11}X_1 + a_{12}X_2 + \cdots a_{1p}X_p \quad (3)$$

The first main element component is measured in such a method that in the data collection it accounts for as many as a possible variation. The second main component (Y_2) is measured in the same manner, provided that the first principal component is not compared with (i.e. perpendicular to) and that the next largest variation is accounted for.

$$Y_2 = a_{21}X_1 + a_{22}X_2 + \cdots a_{2p}X_p \quad (4)$$

This process is repeated until the sum of the principal components of p equals the original number of variables. At this point, the sum of the variances of all the principal components equals the sum of the variances of all the variables.

$$Y = XA \quad (5)$$

V. Conversion of 3D LiDAR to 2.5D image and rotation around Z

The point cloud shows the Z-values that denote height or depth using the 3D point cloud. The positive Z-value point is above the ground, while the negative Z-value point is below the ground and invisible on the map. This problem is solved by the rotation around the z-axis to align the point cloud along x-axis to get rid of negative Z-values using PCA method. In the case of 3D, three elements can

be represented by Matrices for rotation. If it is required to rotate around the z-axis, use the following matrices [22-23]:

$$R_z = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (6)$$

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \quad (7)$$

$$R_y = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \quad (8)$$

VI. Intensity Hue Saturation (IHS) Transformations

Image Fusion is used to combine and place valuable information from a series of input images into a single output image to make it more effective and useful than all of the input images [24]. One of the most widely used fusion methods for sharpening is the IHS method. It has become a traditional image processing technique for color analysis. This is to have improvement, the perfection of features, enhancement of spatial precision, and the convergence of various data sets. Spectral knowledge is often reflected in the Hue and the Saturation. One can infer from the visual system that the change in amplitude has no effect on the spectral details and is simple to work with [25]. Most pieces of literature accept IHS as a third-order approach because of the RGB IHS conversion model. It uses a 3×3 matrix as its transform kernel. Many published studies indicated that the following definition uses separate IHS transformations, which have some significant variations in the values of the matrix:

$$\begin{bmatrix} I \\ V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ \frac{-1}{\sqrt{6}} & \frac{-1}{\sqrt{6}} & \frac{2}{\sqrt{6}} \\ \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \qquad H = \tan^{-1}\left(\frac{V_2}{V_1}\right) \qquad S = \sqrt{V_1^2 + V_2^2} \quad (9)$$

Two intermediate values are where (V_1 and V_2) are. Special case processing and final scaling of the intensity, hue, and saturation values between 0 and 255 are used in the algorithm.

VII. 2.5D image fusion based IHS

In IHS fusion methods, the RGB and depth image are combined taking the same position to produce 2.5D image. Hue, Saturation, and Intensity can be obtained from the RGB color cube. In a color image, the Intensity variable is decoupled from the color carrying data (Hue and Saturation). RGB point converted into a corresponding point is the IHS color by working out the geometrical formulas[26]:

The Hue H is given by $$H = \begin{cases} \theta & \text{if } B \le G \\ 360 - \theta & \text{if } B > G \end{cases} \text{ where} \quad (10)$$

$$\theta = \cos^{-1}\left\{\frac{\frac{1}{2}[(R-G)+(R-B)]}{\sqrt{(R-G)^2+(R-B)(G-B)}}\right\} \quad (11)$$

The Saturation S is given by: $$S = 1 - \frac{3}{(R+G+B)}[\min(R,G,B)] \quad (12)$$

The Intensity I is given by: $$I = \frac{1}{3}(R+G+B) \quad (13)$$

VIII. Measuring of error

The Error Rate (ER), Mean Error of Distance (MED) and Mean Square Error (MSE) are used in the error estimation of an estimated arithmetic circuit. First, the error of Distance (ED) is defined as the difference between the approximate sum S* and the specific sum S, i.e. [27]:

$$\mathrm{ED} = |\mathrm{S}^{*} - \mathrm{S}| \tag{14}$$

The error rate (ER) is the number of input configurations for which the predicted adder delivers incorrect effects. results, i.e., a non-zero error distance. It is determined mathematically as [27]:

$$\mathrm{ER} = \mathrm{P}(\mathrm{ED} \neq 0) \tag{15}$$

The (MED) is the mean value of all distances of error. The (MSE) overall error distances is the Mean value of the squares, measured mathematically as [27]:

$$\mathrm{MED} = \mathrm{E}[\mathrm{ED}] = \sum_{\mathrm{ED_i} \in \Omega} \mathrm{ED_i} \mathrm{P}(\mathrm{ED_i}) \tag{16}$$

$$\mathrm{MSE} = \mathrm{E}[\mathrm{ED}^{2}] = \sum_{\mathrm{ED_i} \in \Omega} \mathrm{ED_i^{2}} \mathrm{P}(\mathrm{ED_i}) \tag{17}$$

where **Ω** is the set of all error distances.

If n predictions are created from a sample of n data, Y is the variable being predicted, with $\widehat{\mathrm{Y}}$ is being the predicted value, then MSE is calculated as [27]:

$$\mathrm{MSE} = \frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} \left(\mathrm{Y_i} - \widehat{\mathrm{Y}}_{\mathrm{i}}\right)^{2} \tag{18}$$

3. PROPOSED OUTDOOR LOCALIZATION SYSTEM

The proposed method in this research for knowing the location of a mobile robot is divided into three stages. Each stage consists of several operations: training, testing, and classification stages. In the training stage, the LiDAR sensor performs a scan to obtain 3D point cloud data. Converting a 3D point cloud to a 2.5D image is using the PCA method. The feature is extracted from 2.5D images using CNN algorithm. It stores all featured data, point cloud data, and all pre-processing-related data as a matrix. In the testing stage, it uses RGB and Depth sensors to get two images of the same location. Then, combine two RGB images and depth (D) into an RGB-D merge image to be a single 2.5D by IHS method. The features are extracted from 2.5D with CNN algorithm. All feature data are then located in a matrix. In the classification stage, the classifier K-NN, the test data is classified with the training data stored to find the correct location of the mobile robot.

Assume the proposed system consists of two sensors (LiDAR and RGB-D) are mounted on a mobile robot in order to collect the dataset for training and testing, as shown in Figure 4.

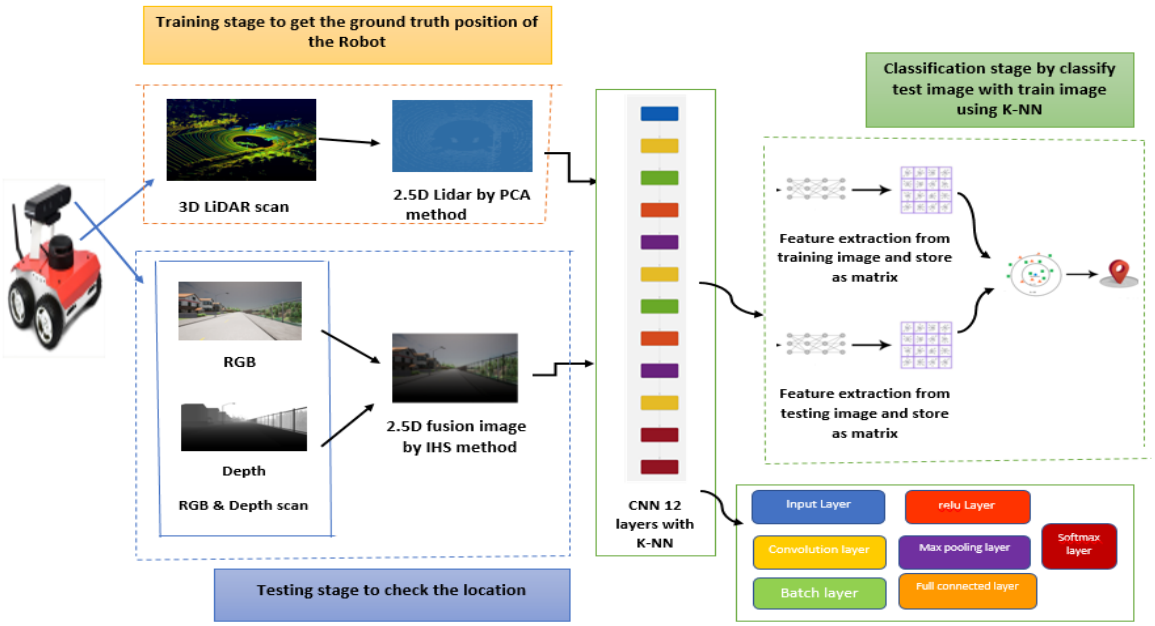


Figure 4: Proposed outdoor localization system

4. EXPERIMENTAL RESULTS

I. Collected Datasets

A large amount of data is typically needed to train a neural network with supervised learning. A dataset contains RGB images, depth images, High-resolution LiDAR scans with corresponding mode designations is required to train and test the device design. This was not, however, found in the public domain and it was agreed to use a simulator to produce the data needed. The CARLA driving simulator is used to produce simulated results. CARLA is an open-source simulator. In CARLA a camera sensor can be connected to a mobile robot, capturing images with a frame rate preset [28]. The camera sensor will create images in both RGB and depth, as seen in Figure 5.

(b) RGB image **(a) Depth image**

Figure 5: RGB image explanation and corresponding Depth map (photos from the simulator)

In CARLA emulated LiDAR sensors are available. All related parameters can be configured, such as the upper and lower fields of vision, number of channels, maximum range, and the number of points per channel. The simulation area can be frozen during a scan capture, resulting in a 360 ° scan without any velocity changes needed. See Figures 6 and 7.

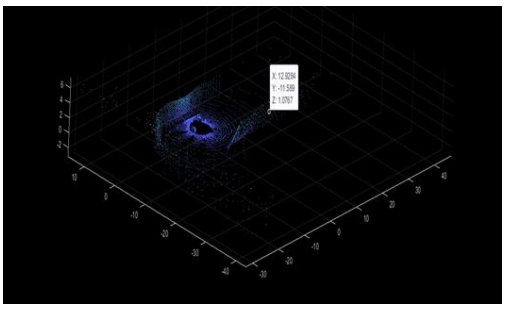


Figure 6: LiDAR scan as a point cloud and each point has X, Y, Z coordinate image processed in MatLab

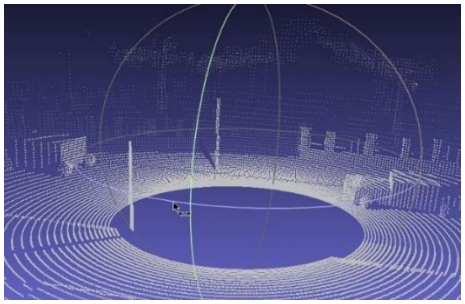


Figure 7: LiDAR output after being processed in MeshLab [28]

A mobile robot with an RGB, a depth camera sensor, and LiDAR sensor attached to it was set to drive around a map on autopilot to produce the data sets used in this research. Approximately 120,000 m^2 of a map is used to produce the data collection including the downtown area, residential areas, and wooded areas. There were two training data sets, each with 7600 image pairs of RGB and Depth and 46,741 frames of LiDAR, acquired. This city is divided into 9 streets in addition to a street between them as shown in Figure 8.

Figure 8: Proposed street numbers used in Carla simulator

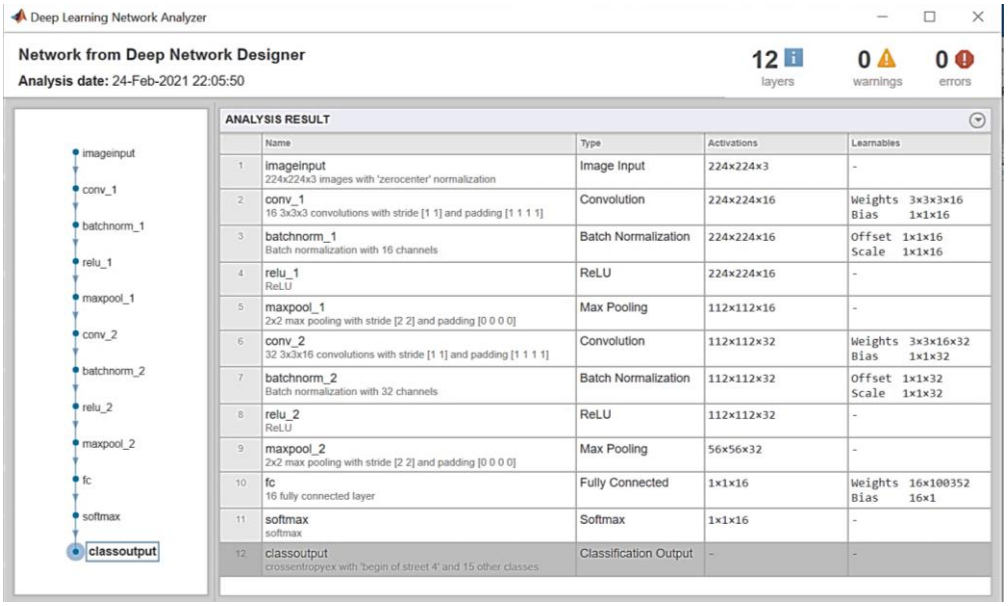
Each street is divided into the beginning and the end of the street, as listed in Table I.

TABLE I: Streets division and data for each street

Item	Street name	Approximate LiDAR frames for each street	Number of testing images for each street (RGB and Depth)
1	Beginning of street 1	126	56
2	End of street 1	2382	54
3	Beginning of street 2	3453	22
4	End of street 2	3332	67
5	street 3	3632	11
6	Beginning of street 4	749	38
7	End of street 4	3267	25
8	Beginning of street 5	3824	76
9	End of street 5	4643	37
10	Street 6	2463	34
11	Beginning of street 7	3267	53
12	End of street 7	3688	32
13	Street 8	4635	49
14	Beginning of street 9	2377	20
15	End of street 9	3745	21
16	Between	749	31

II. CNN Design and architecture

To improve the accuracy of the results and reduce error with short training time, a 12-layer CNN was designed with an input image of 224 x 224. A number of researchers used Gradient Descent with a Momentum (GDM) algorithm to train the neural network used for backpropagation [29]. In this network, using optimizer Stochastic Gradient Descent with Momentum (SGDM), which is always better and faster than (GDM) algorithm [30]. With the K-NN classifier, 16 classes according to the number of streets are identified in Figure 8. Details of CNN's design are shown in Figure 9.



	Name	Type	Activations	Learnables
1	imageinput 224x224x3 images with 'zerocenter' normalization	Image Input	224×224×3	-
2	conv_1 16 3x3x3 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	224×224×16	Weights 3×3×3×16 Bias 1×1×16
3	batchnorm_1 Batch normalization with 16 channels	Batch Normalization	224×224×16	Offset 1×1×16 Scale 1×1×16
4	relu_1 ReLU	ReLU	224×224×16	-
5	maxpool_1 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	112×112×16	-
6	conv_2 32 3x3x16 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	112×112×32	Weights 3×3×16×32 Bias 1×1×32
7	batchnorm_2 Batch normalization with 32 channels	Batch Normalization	112×112×32	Offset 1×1×32 Scale 1×1×32
8	relu_2 ReLU	ReLU	112×112×32	-
9	maxpool_2 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	56×56×32	-
10	fc 16 fully connected layer	Fully Connected	1×1×16	Weights 16×100352 Bias 16×1
11	softmax softmax	Softmax	1×1×16	-
12	classoutput crossentropyex with 'begin of street 4' and 15 other classes	Classification Output	-	-

Figure 9: Design of CNN with 12 layers and K-NN classifier

III. Training the network & preprocessing

MATLAB code is written using a PC with specifications of Intel(R) Core (TM) i5-8250U CPU @1.60GHz 1.80 GHz UHD Graphics 620, RAM (8 GB). As the dataset includes 46,741 frames LiDAR, it took 4 epochs. Iteration for each epoch is 21 and a maximum of 84 iterations, and an initial learning rate of $3*10^{-4}$. It took 138 minutes for training, with 70 % of the training data and 30 % of the testing. The accuracy obtained is 97.52%, as seen in Figure 10.

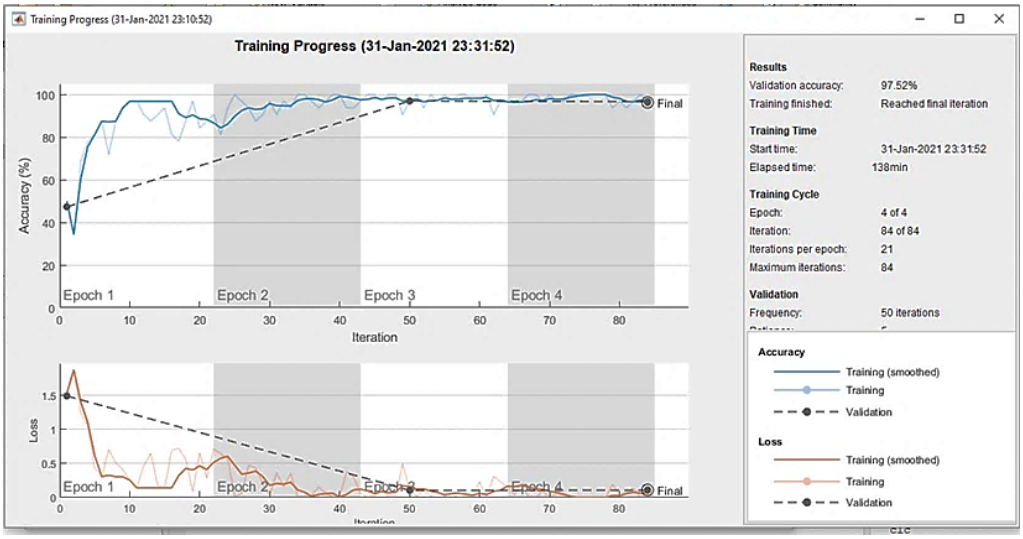


Figure 10: Training time spent and accuracy for training

IV. IHS method results

IHS method affects when merging RGB and Depth images. The Intensity value varies depending on how bright the pixel is the depth image, Hue, saturation, and intensity merge in the 2.5d image, see Figure 11.

(a) RGB Image **(b) Depth Image** **(c) IHS Image**

Figure 11: Production of 2.5D image by merge two images RGB & Depth in IHS method

V. Performance with PCA and K-NN algorithm

Testing 5 cases of RGB and Depth images by chosen random locations in the city, the MSE is calculated, then taking the average of MSE for the test image, as listed in Table II.

TABLE II: MSE from some cases of the test image

Case	The correct prediction of localization	MSE
1	beginning of street 1	0.09224
2	begin of street 9	0.0326
3	end of street 5	0.0598
4	between	0.0489
5	street 6	0.0543
	Mean error	0.057568
	Median error	0.0489

To calculate the Mean error in distance. Simulator from CARLA recorded 30 to 50 Frame Per Second (FPS) in one meter. From equation (16), MED = 0.804 meters.

Regarding Table III, the proposed method is excellent than that used in research [31]. As the used the iterative closest point (ICP) algorithm, the PointNetLK network is used for registration and GoogleNet for RGB-D Neural network.

TABLE III: Comparison between the proposed method with other methods

Method	Mean error	Training time	Training dataset
ICP+ PointNetLK+ GoogleNet [31]	30.3 meters	4200 minutes	7600
Proposed method CNN +PCA +K-NN	0.804 meters	138 minutes	46741

Improvement in network training time and the error rate is clear because this method [31] took 4200 minutes for 7600 images dataset and mean error of 30.3 meters. The proposed method took a training time of 138 minutes for 46,741 frames of LiDAR dataset and the accuracy is 97.52%, MSE equals to 0.057568, and Mean Error of Distance equals to 0.804 meters using PCA method, IHS is used for fusion image and K-NN classifier. K-NN classifier gives more accuracy results and it is not required for training to obtain the results.

5. CONCLUSION

In this paper, the mobile robot localization system is designed to resolve the issue of robot position loss in the outdoor environment due to many factors in the outdoor that affect the sensors mounted with the robot. This leads to inaccuracies in calculating the position. Therefore, proposing the use of 3D sensors to achieve more accuracy with the aid of Deep Learning algorithms. The proposed design is based on three stages: training, testing, and classification. The method uses PCA for reducing the dimension and rotate the point cloud 3D LiDAR, IHS method is used to make the 2.5D RGB-D fusion image reduced and K-NN algorithm to obtain the results with high accuracy and less training time.

Experimental results are enhanced as compared with results obtained in reference number 31. The training time is reduced to 138 minutes with 97.52% accuracy, MSE equals 0.057568, and MED equals 0.804 meters.

References

[1] R. T. Kamil, M. J. Mohamed, B. K. Oleiwi, Path Planning of Mobile Robot Using Improved Artificial Bee Colony Algorithm, Eng. Technol. J., 38 (2020) 1384–1395. https://doi.org/10.30684/etj.v38i9a.1100

[2] X. Wang, X. Wang, D. M. Wilkes, Machine Learning-based Natural Scene Recognition for Mobile Robot Localization in An Unknown Environment, Springer, 2020.

[3] K. Ohno, T. Tsubouchi, B. Shigematsu, S. Yuta, Differential GPS and odometry-based outdoor navigation of a mobile robot, Adv. Robot., 18 (2004) 611–635. https://doi.org/10.1163/1568553041257431

[4] J. Fuentes-Pacheco, J. Ruiz-Ascencio, J. M. Rendón-Mancha, Visual simultaneous localization and mapping: a survey, Artif. Intell. Rev., 43 (2015) 55–81. https://doi.org/10.1007/s10462-012-9365-8

[5] L. Zhang, L. Wei, P. Shen, W. Wei, G. Zhu, J. Song, Semantic SLAM based on object detection and improved octomap, IEEE Access, 6 (2018) 75545–75559. https://doi.org/10.1109/ACCESS.2018.2873617

[6] S. Oishi, Y. Inoue, J. Miura, S. Tanaka, SeqSLAM++: View-based robot localization and navigation, Rob. Auton. Syst., 112 (2019) 13–21. https://doi.org/10.1016/j.robot.2018.10.014

[7] K. Atsuzawa, Robot Navigation in Outdoor Environments using Odometry and IS2-3 Robot Navigation in Outdoor Environments using Odometry and Convolutional Neural Network, IEEJ international workshop on sensing, actuation, motion control, and optimization (SAMCON) , 2019.

[8] C. M. J. M. D. Junior, S. P. da Silva, R. V. da Nobrega, A. C. Barros, A. K. Sangaiah, P. P. Reboucas Filho, V. H.C.de Albuquerque, A new approach for mobile robot localization based on an online IoT system, Futur. Gener. Comput. Syst., 100 (2019) 859–881. https://doi.org/10.1016/j.future.2019.05.074

[9] X. Li, S. Du, G. Li, H. Li, Integrate point-cloud segmentation with 3d lidar scan-matching for mobile robot localization and mapping, Sensors , 20 (2020). https://doi.org/10.3390/s20010237

[10] J. Li, C. Wang, X. Kang, Q. Zhao, Camera localization for augmented reality and indoor positioning: a vision-based 3D feature database approach, Int. J. Digit. Earth, 13 (2020) 727–741. https://doi.org/10.1080/17538947.2018.1564379

[11] Y. Zheng, S. Chen, H. Cheng, Real-time cloud visual simultaneous localization and mapping for indoor service robots, IEEE Access, 8 (2020) 16816–16829. https://doi.org/10.1109/ACCESS.2020.2966757

[12] C. Debeunne, D. Vivet, A review of visual-lidar fusion based simultaneous localization and mapping, Sensors , 20 (2020) 1-20 . https://doi.org/10.3390/s20072068

[13] L. Wellhausen, A. Dosovitskiy, R. Ranftl, K. Walas, C. Cadena, M. Hutter, Where should i walk ?Predicting terrain properties from images via self-supervised learning, IEEE Robot. Autom. Lett., 4 (2019) 1509–1516. https://doi.org/10.1109/LRA.2019.2895390

[14] A. Kanezaki, Y. Matsushita, Y. Nishida, RotationNet for Joint Object Categorization and Unsupervised Pose Estimation from Multi-View Images, IEEE Trans. Pattern Anal. Mach. Intell., 43 (2021) 269–283. https://doi.org/10.1109/TPAMI.2019.2922640

[15] Z. Qiu, Y. Zhuang, F. Yan, H. Hu, W. Wang, RGB-DI Images and Full Convolution Neural Network-Based Outdoor Scene Understanding for Mobile Robots, IEEE Trans. Instrum. Meas., 68 (2019) 27–37. https://doi.org/10.1109/TIM.2018.2834085

[16] A. A. Abdulhussein, F. A. Raheem, Hand Gesture Recognition of Static Letters American Sign Language (ASL) Using Deep Learning, Eng. Technol. J., 38 (2020) 926–937. https://doi.org/10.30684/etj.v38i6a.533

[17] N. Ali, D. Neagu, P. Trundle, Evaluation of k-nearest neighbour classifier performance for heterogeneous data sets, SN Appl. Sci., 1 (2019) 1–15. https://doi.org/10.1007/s42452-019-1356-9

[18] Y. S. Park, Diagnostic cluster analysis of mathematics skills, Evaluation, 4 (2007) 75–108.

[19] Y. Duan, C. Yang, H. Chen, W. Yan, H. Li, Low-complexity point cloud denoising for LiDAR by PCA-based dimension reduction, Opt. Commun., 482 (2021). https://doi.org/10.1016/j.optcom.2020.126567

[20] A. Daffertshofer, C. J. C. Lamoth, PCA in studying coordination and variability : a tutorial, Clin. Biomech., 19 (2004) 415-428 . https://doi.org/10.1016/j.clinbiomech.2004.01.005

[21] S. M. Holland, Principal components analysis (PCA) : A tutorial in the R-Studio environment. Department of Geology, University of Georgia, Athens, (2008) 30602–32501.

[22] Z. A. Deng, G. Wang, Y. Hu, D. Wu, Heading estimation for indoor pedestrian navigation using a smartphone in the pocket, Sensors, 15 (2015) 21518–21536. https://doi.org/10.3390/s150921518

[23] M. Kazhdan, An approximate and efficient method for optimal rotation alignment of 3D models, IEEE Trans. Pattern Anal. Mach. Intell., 29 (2007) 1221–1229. https://doi.org/10.1109/TPAMI.2007.1032

[24] N. A. Mohamed, M. S. H. Al-Tamimi, Image fusion techniques: A review, Int. J. Psychosoc. Rehabil., 24 (2020) 2194–2214. https://doi.org/10.37200/IJPR/V24I10/PR300238

[25] F. A. Al-Wassai, N. V. Kalyankar, A. A. Al-Zuky, The IHS Transformations Based Image Fusion, arXiv,2011. https://doi.org/10.48550/arXiv.1107.4396

[26] R. A. Jayashree, RGB to HSI color space conversion via MACT algorithm, Int. Conf. Commun. Signal Process., (2013) 561–565. https://doi.org/10.1109/iccsp.2013.6577117

[27] Y. Wu, Y. Li, X. Ge, Y. Gao, W. Qian, An Efficient Method for Calculating the Error Statistics of Block-Based Approximate Adders, IEEE Trans. Comput., 68 (2019) 21–38. https://doi.org/10.1109/TC.2018.2859960

[28] CARLA Simulator. http://carla.org / (accessed Feb. 09, 2021).

[29] M. Y. Hassan, G. Kothapalli, Comparison between neural network based PI and PID controllers, 2010 7th Int. Multi-Conference Syst. Signals Devices, (2010) 1-6. https://doi.org/10.1109/SSD.2010.5585598

[30] B. Zhou, J. Liu, W. Sun, R. Chen, C. Tomlin, Y. Yuan, pbSGD: Powered stochastic gradient descent methods for accelerated nonconvex optimization, International Joint Conferences on Artificial Intelligence Organization, (2020) 3258–3266. https://doi.org/10.24963/ijcai.2020/451

[31] S. Bastås, R. Brenick, Outdoor global pose estimation from RGB and 3D data, M.Sc. thesis, in Systems, Control and Mechatronics, Dept. of Mechanics and Maritime Sciences, Chalmers University Of Technology, Gothenburg, Sweden .2019. http://publications.lib.chalmers.se/records/fulltext/256908/256908.pdf