BLG561E

Deep Learning Term Project

CNN-based Localization System for an Autonomous Mobile Robot

Instructor: Dr. Faik Boray Tek

Team Members:

504232106 - Emre Can Contarlı
504232517 - Fulya Yenilmez

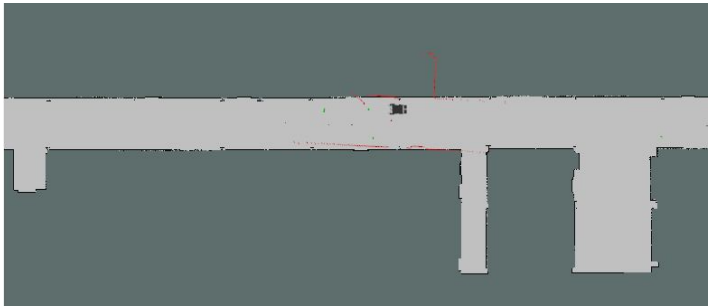
Content

- Problem Definition
- Literature Review
- Dataset Creation
- Motivation
- Methodology/Comparison
- Results

Problem Definition

Main Tasks of an Autonomous Robot:

- Localization *(Focus of this project)*
- Mapping
- Path Planning
- Control



Focus

This project specifically focuses on the Localization task, employing Convolutional Neural Networks to accurately determine the robot's position within a known map.

Literature Review

Localization strategies for autonomous mobile robots: A review
PK Panigrahi, SK Bisoy

Discusses various localization techniques for mobile robots, including neural network approaches. It covers the use of neural networks for estimating the robot's location and environment map.

Scene recognition for indoor localization of mobile robots using deep CNN
P Wozniak, H Afrisal, RG Esparza, B Kwolek

Presents a deep neural network approach for indoor localization by recognizing scenes from images.

Literature Review

Real-time object navigation with deep neural networks and hierarchical reinforcement learning
A Staroverov, DA Yudin, I Belkin, V Adeshkin

Implements a neural network for robot localization using RGB-D images and hierarchical reinforcement learning.

Dynamic-SLAM: Semantic monocular visual localization and mapping based on deep learning in dynamic environment
L Xiao, J Wang, X Qiu, Z Rong, X Zou

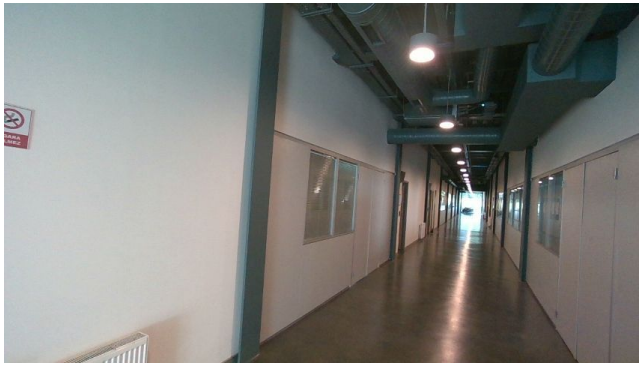
Uses CNN to improve SLAM by integrating semantic information for better localization and mapping in dynamic environments.

Dataset Creation

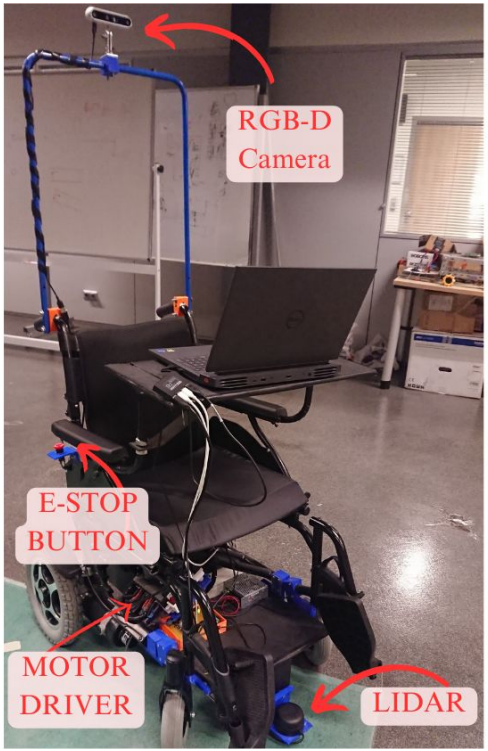
By operating autonomous wheelchair in corridor environment,

- Pictures of corridor
- AMCL pose and heading information

are recorded



X: 0.4272
Y: 11.5382
Yaw: -86.80



Dataset Creation

Total of 19341 images are obtained.

Images are then resized to be 224x224.

Dataset Creation

Camera worked with 30 Hz, AMCL worked with 1.61 Hz.

19341 images, 1037 pose information.

In data pairs, missing pose information for some timestamps are obtained by interpolation.

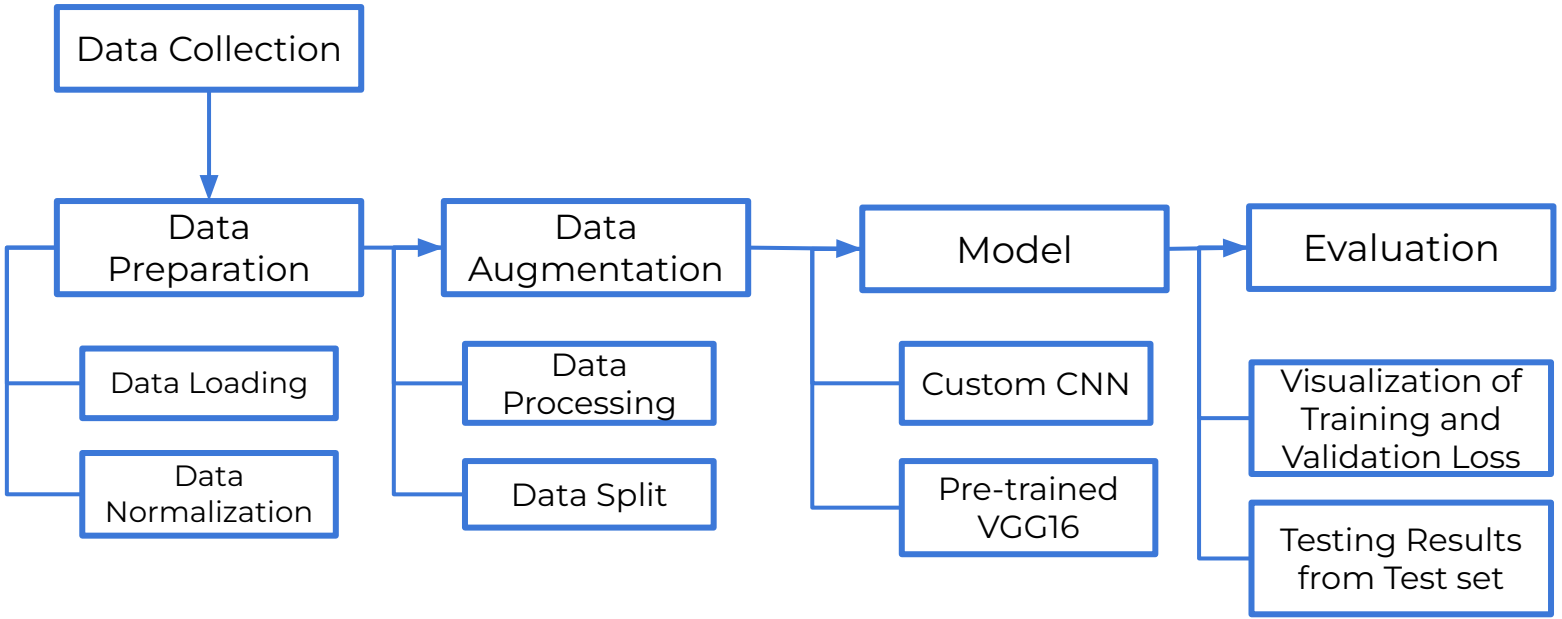
1715524888.23514.png	0.423417130561398	11.6197785161601	-86.8361103454159
1715524888.26974.png	0.423588053008948	11.6161109871625	-86.8347798814291
1715524888.30228.png	0.423748798290445	11.6126618323584	-86.8335286367205
1715524888.33540.png	0.42391241027975	11.6091511658326	-86.832255077497
1715524888.36918.png	0.424079281176412	11.6055705720538	-86.830956150869
1715524888.40179.png	0.424240372724176	11.6021139873211	-86.8297022108163
1715524888.43508.png	0.42440482446807	11.5985853019549	-86.8284221149252
1715524888.46888.png	0.424571794297951	11.5950025853393	-86.8271224181988
1715524888.50182.png	0.424734517066059	11.5915109990723	-86.8258557806921
1715524888.53523.png	0.424899560053716	11.5879696272294	-86.8245710825469
1715524888.5686.png	0.42506440635271	11.5844324757881	-86.8232879154304
1715524888.60195.png	0.425229153718485	11.5808974471836	-86.8220055184121
1715524888.63551.png	0.425394938705287	11.5773401540652	-86.8207150445297
1715524888.66858.png	0.425558303361544	11.5738347946312	-86.8194434105519
1715524888.70162.png	0.425721519617972	11.5703326194524	-86.8181729317216
1715524888.73530.png	0.425887897026313	11.5667626145854	-86.8168778464172

Motivation

- Humans can accurately determine their position and heading in a house (or corridor).
- In such scenarios, the Bayes error (min. possible error rate) is very close to 0.

Project goal: Why shouldn't we achieve similar results with robots by processing images from robots point of view? Also by eliminating the need for LIDAR and odometry sources.

Methodology / Diagram of Pipeline



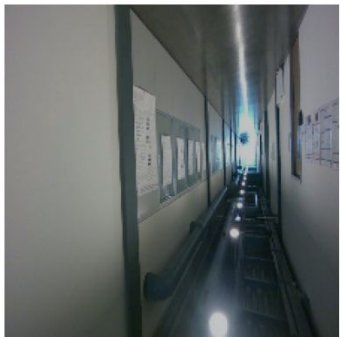
Methodology

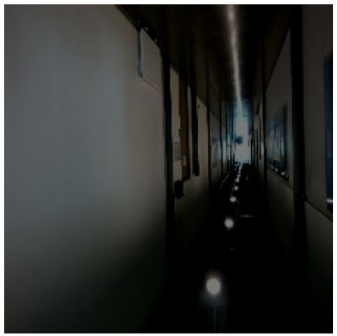
Normalization and Image Preprocessing

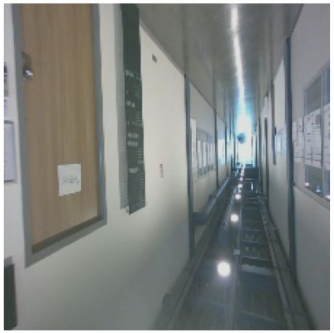
- Normalization: Scaling x and y values between 0 and 1.
- Image Preprocessing: Resized to 224x224, and normalizing pixel values.

Data Augmentation

- Random vertical flips
- Random brightness adjustments
- Contrast adjustments
- Saturation adjustments



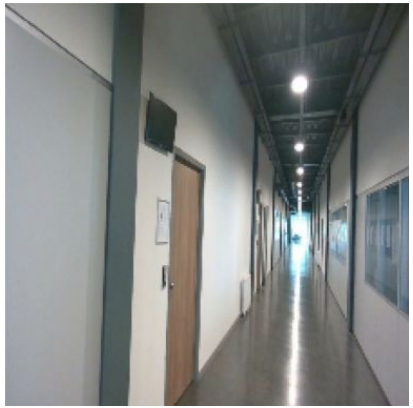




Methodology

Data Preparation

- %80 training, %10 validation, %10 testing
- Coordinates x and y are normalized between [0,1] for regression
- Orientation yaw values are classified as 0 (North) or 1 (South) for classification



x : 0.88836191

y : 0.79869645

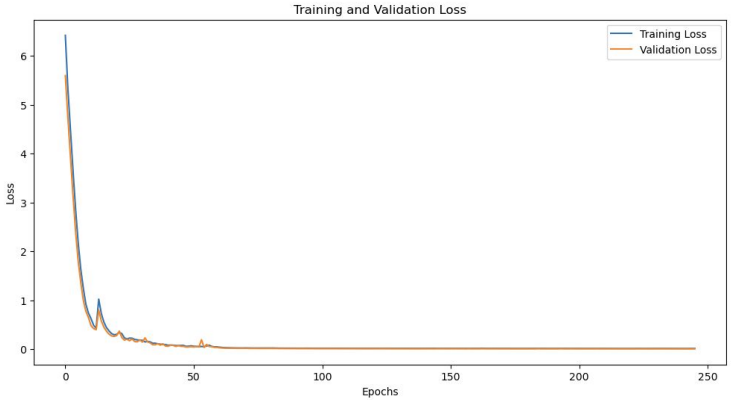
yaw : 1

Custom CNN Model Architecture

Layer (type)	Output Shape	Param #	Connected to
input_layer_2 (InputLayer)	(None, 224, 224, 3)	0	–
conv2d_8 (Conv2D)	(None, 222, 222, 32)	896	input_layer_2[0]…
max_pooling2d_8 (MaxPooling2D)	(None, 111, 111, 32)	0	conv2d_8[0][0]
batch_normalizatio… (BatchNormalizatio…	(None, 111, 111, 32)	128	max_pooling2d_8[…
conv2d_9 (Conv2D)	(None, 109, 109, 64)	18,496	batch_normalizat…
max_pooling2d_9 (MaxPooling2D)	(None, 54, 54, 64)	0	conv2d_9[0][0]
batch_normalizatio… (BatchNormalizatio…	(None, 54, 54, 64)	256	max_pooling2d_9[…
conv2d_10 (Conv2D)	(None, 52, 52, 128)	73,856	batch_normalizat…
max_pooling2d_10 (MaxPooling2D)	(None, 26, 26, 128)	0	conv2d_10[0][0]
batch_normalizatio… (BatchNormalizatio…	(None, 26, 26, 128)	512	max_pooling2d_10…
conv2d_11 (Conv2D)	(None, 24, 24, 256)	295,168	batch_normalizat…
max_pooling2d_11 (MaxPooling2D)	(None, 12, 12, 256)	0	conv2d_11[0][0]
batch_normalizatio… (BatchNormalizatio…	(None, 12, 12, 256)	1,024	max_pooling2d_11…
flatten_2 (Flatten)	(None, 36864)	0	batch_normalizat…
dense_3 (Dense)	(None, 128)	4,718,720	flatten_2[0][0]
dropout_2 (Dropout)	(None, 128)	0	dense_3[0][0]
xy_output (Dense)	(None, 2)	258	dropout_2[0][0]
yaw_output (Dense)	(None, 1)	129	dropout_2[0][0]

Total params: 5,109,443 (19.49 MB)
Trainable params: 5,108,483 (19.49 MB)
Non-trainable params: 960 (3.75 KB)

Table: Custom CNN architecture



Graph: Training and Validation loss

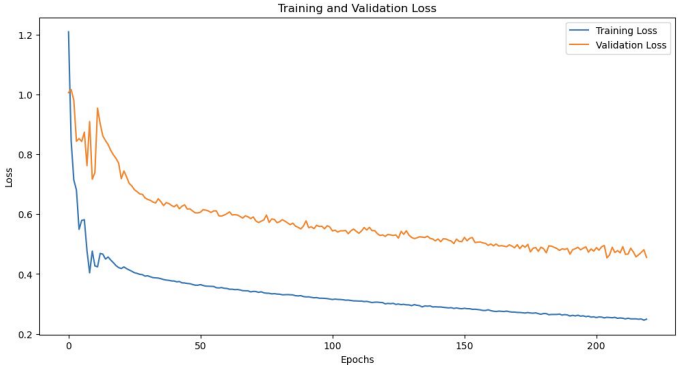
```
Test loss: 0.009984875097870827
Test XY MSE: 0.0006472535314969718
Test XY RMSE: 0.02544117787165075
Test Yaw Accuracy: 1.0
```

Pre-trained VGG16 model

Layer (type)	Output Shape	Param #	Connected to
input_layer_1 (InputLayer)	(None, 224, 224, 3)	0	–
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1,792	input_layer_1[0]…
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36,928	block1_conv1[0][…
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0	block1_conv2[0][…
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73,856	block1_pool[0][0]
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147,584	block2_conv1[0][…
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0	block2_conv2[0][…
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295,168	block2_pool[0][0]
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590,080	block3_conv1[0][…
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590,080	block3_conv2[0][…
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0	block3_conv3[0][…
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1,180,160	block3_pool[0][0]
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2,359,808	block4_conv1[0][…
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2,359,808	block4_conv2[0][…
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0	block4_conv3[0][…
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2,359,808	block4_pool[0][0]
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2,359,808	block5_conv1[0][…
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2,359,808	block5_conv2[0][…
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0	block5_conv3[0][…
flatten_1 (Flatten)	(None, 25088)	0	block5_pool[0][0]
dense_1 (Dense)	(None, 128)	3,211,392	flatten_1[0][0]
dropout_1 (Dropout)	(None, 128)	0	dense_1[0][0]
xy_output (Dense)	(None, 2)	258	dropout_1[0][0]
yaw_output (Dense)	(None, 1)	129	dropout_1[0][0]

Total params: 24,350,027 (92.89 MB)
Trainable params: 3,211,779 (12.25 MB)
Non-trainable params: 14,714,688 (56.13 MB)

Table: Pre-trained VGG16 architecture



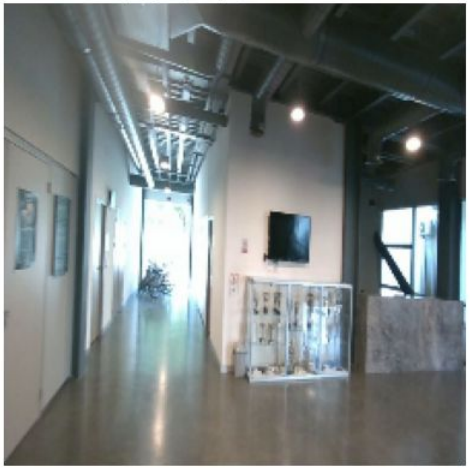
Graph: Training and Validation loss

```
Test loss: 0.157634481787681 58
Test XY MSE: 0.027530176565051 08
Test XY RMSE: 0.165922200338143 64
Test Yaw Accuracy: 0.996382415294647 2
```

Results

Custom CNN Model Performance on Test Set

Actual values (x, y, yaw): [0.84759884 0.1604469], 1
Predicted values (x, y, yaw): [0.8351339 0.17262709], 1



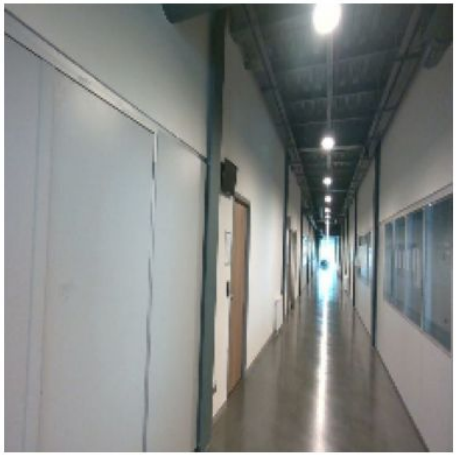
Actual values (x, y, yaw): [0.91778887 0.5087811], 0
Predicted values (x, y, yaw): [0.8858749 0.49853435], 0

Actual values (x, y, yaw): [0.97603217 0.78405958], 1
Predicted values (x, y, yaw): [0.9518992 0.7869341], 1

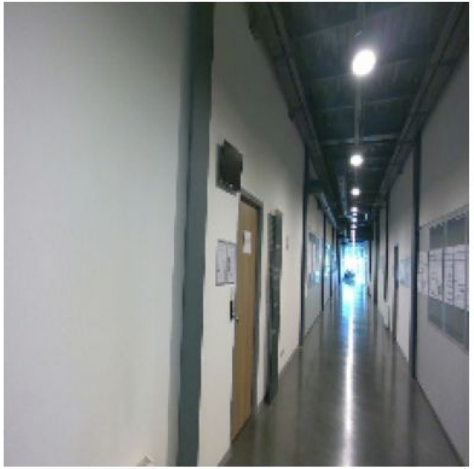
Results

Pre-trained VGG16 Model Performance on Test Set

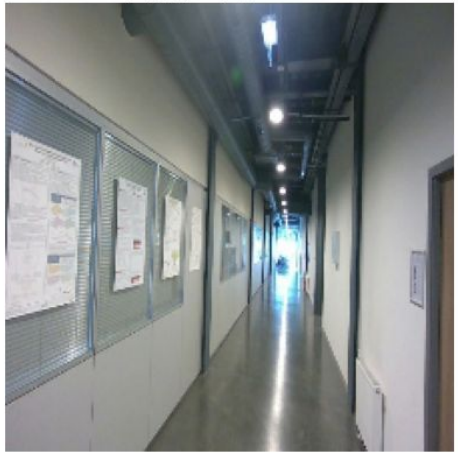
Actual values (x, y, yaw): [0.88927975 0.8266567], 1
Predicted values (x, y, yaw): [0.85331833 0.49721748], 1



Actual values (x, y, yaw): [0.90194757 0.69123418], 1
Predicted values (x, y, yaw): [0.8423177 0.47713402], 1



Actual values (x, y, yaw): [0.82385622 0.47862006], 1
Predicted values (x, y, yaw): [0.8052581 0.4122523], 1



Any questions?