A
End Semester
Report
on

Vision Aided Drone Localization in GPS-denied Indoor Corridor Environments

Submitted in partial fulfillment of
the requirements for the award of the degree of

Master of Technology
in
Computer Science and Engineering

Submitted by

216CS1134 Shahzad Ahmad

Under the guidance of
Prof. Dr Pankaj Kumar Sa

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela, ODISHA, India – 769008

Spring Semester 2018

Abstract

Vision based navigation of unmanned aerial vehicles (UAV) has been an active field of research in the past decade. There are many challenges in making the vision system understand the environment in which it is placed. Such an environment can be either indoors or outdoors depending on the task a hand.In This project we deals with navigation of UAV in GPS-denied indoor environment.we consider A.R Parrot drone is our UAV model.Our aim is to localization of drone in corridor during flying time with the help of monocular camera which is attach with drone.In this project we designe navigatin algorithm which will tell us the position of drone in corridor by seeing the images which are taken by drone camera.Our navigation algorithm impliment with help of deep learning models use to perform regression task on the images which are taken by drone camera.

Contents

1 Introduction

1.1 Overview of Deep Learning

Deep learning comes from the concept of human brain having multiple types of representation with simpler features at the lower levels and high-level abstractions built on top of that. Humans arrange their ideas and concepts hierarchically. Humans first learn simple concepts and then compose them to represent more abstract ones. Them human brain is like deep neural network, consisting of many layers of neurons which act as feature detectors, detecting more abstract features as the levels go up. This way of representing information in a more abstract way is easier to generalize for the machines[3].
The main advantage of deep learning is its compact representation of a larger set of functions than shallow networks used by most conventional learning methods. A deep architecture is more expressive than a shallow one provided the same number of non-linear units. But functions compactly represented in k layers may require exponential size when expressed in 2 layers. Formally, it can be proved that a k-layer network can represent functions compactly but a k-1 layer network cannot represent them unless it has an exponentially large number of hidden units. A lot of factors like faster CPUs, parallel CPU architectures, GPU computing enabled training of deep networks and made it computationally feasible. Neural networks are often represented as a matrix of weight vectors and GPUs are optimized for very fast matrix multiplication Deep Learning methods have performed very well in MNSIT digit recognition dataset.Our setting is very similar to the task of digit recognition. Corresponding to the digit labels we have emotion labels. But emotion recognition is much more complicated because digit images are much simpler than face images depicting various expressions.Moreover the variability in the images due to different identities hampers the performance. Human accuracy in facial exression recognition is not as good as in digit recognition and is also aided by other modes of information such as context,prior experience,speech among others[4].

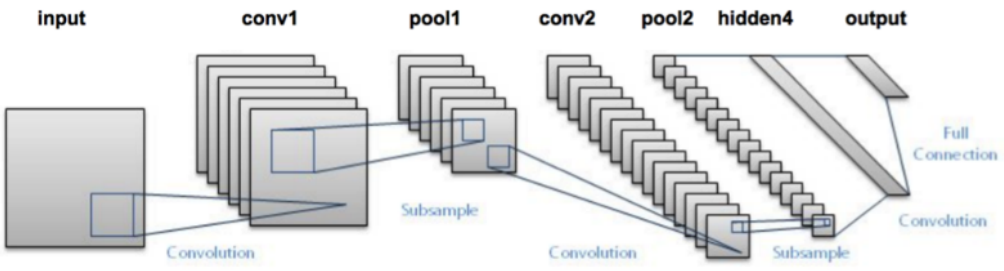

Figure 1: Architecture of CNN

2 Related Work

Literature Survey		
Title	Proposed Work	Merits
ImageNet Classification with Deep Convolutional Networks[1]	The network was made up of 5 conv layers, max-pooling layers, dropout layers, and 3 fully connected layers	Utilizing techniques that are still used today, such as data augmentation and dropout.
Visualizing and Understanding Convolutional Neural Networks[3]	Developed a visualization technique named Deconvolutional Network	discussed the idea that renewed interest in CNNs, accessibility of large training sets and increased computational power with the usage of GPUs.
Deep Residual Learning for Image Recognition[4]	came up with the concept of residual block.	during the backward pass of backpropagation, the gradient will flow easily through the graph, which where distributing the gradient.
Densely Connected Convolutional Networks[5]	propose a dynamic model which connect every layer with all of its previous layer.	fast gradient update.

Table 1:

3 Problem Statement

Unmanned Aerial Vehicles(UAVs) or which are more popularly known as Drones or Quadcopter have pushed their way into modern day problem solving techniques and by far have successfully proved their usability in Defence Activities, Reconciliation Missions and photography. With the improvements in optimized energy utilization during flight, newer dimensions of UAVs deployment are being explored. These may include Drones in agriculture, disaster management, tracking, area mapping, traffic control, crime monitoring and surveillance. Yet again these problems can be solved again taking into account in what type of environment we want to deploy our drone. These environments may have the property of being windy, high altitude(low pressure), low visibility(fog), night conditions or indoor environments. These conditions are not mutually exclusive of each other.

Navigation of drones are basically dependent on GPS coordinates, but in indoor environments , getting GPS signal is tough. Instead of flying blind, we make use of the on-board Camera and apply deep learning techniques to sense the environment and make a meaningful flight pattern by avoiding obstacles and also gathering maximum information on the go.

4 Objective

In this project we will implement a deep network to guide a drone to autonomously navigate in indoor environment. The problem in indoor environment basically lies in the non-availability of GPS coordinates. Contrary to the present age state of the art navigation methods for UAVs, use GPS localizations alone and manual override wherever necessary.The goal of the project is to develop a way to guide drone such that it can navigate inside a in corridors Today indoor navigation is still an unsolved problem, thus our research attempts to solve it.

In our project we will work on a manufactured Drone from Parrot named A.R Drone version 2.0. We will be working with its monocular HD primary Camera with frame rate of 30fps. We shall have an Ubuntu System with a powerful processor (2.7 or higher clocked) and interfaced with ROS(Robot Operating System).

Deep network in our project will take the raw image as active input,captured from the camera at the drone side, and predict angle between the horizontal axis and axis of the corridor and the distance between point of intersection of horizontal axis and the axis of corridor.

5 Navigation Algorithm

The first phase of the research we design navigation algorithm which help us to find the position of drone in indoor environment.In our research we consider the corridors as indoor environment. Our purposed algorithm work for autonomous navigation of drone in corridors.The idea of algorithm is we take help from bisector of floor in corridor for finding position of drone.In corridor we consider three main position of drone which are drone lies left side ,center and right side of corridor.It is easily visible from given figure-2.

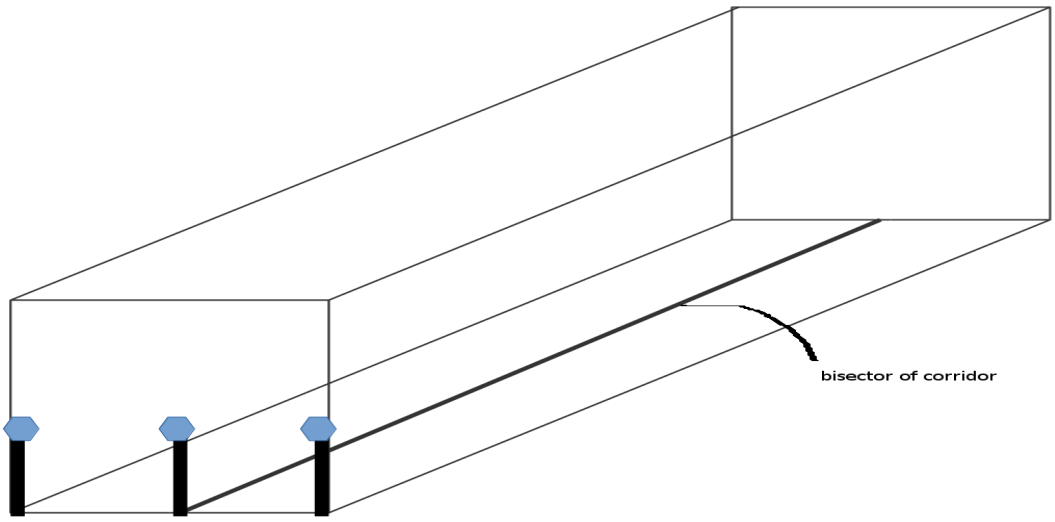

Figure 2: bisector of corridor plane and positions of drone

- If our drone lie left side of corridor and we take image from drone camera then image of bisector make angle less than 90° degree from horizontal axis of image in this case we give right shift command to the drone to bring drone in center.

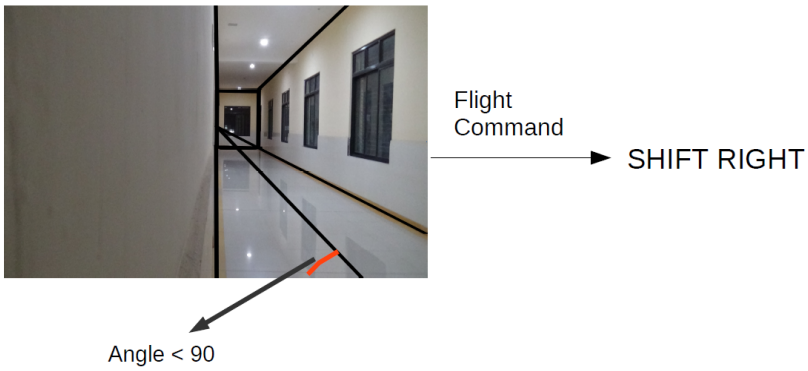

Figure 3: Drone lie left side of corridor

- If our drone lie right side of corridor and we take image from drone camera then image of bisector make angle greater than 90° degree from horizontal axis of image in this case we give left shift command to the drone to bring drone in center.

Figure 4: Drone right left side of corridor

- If our drone lie center of corridor then bisector make angle equal to 90° degree but tilted toward right in this case intersection point of bisector and horizontal axis of image lies first half of the image then we give rotate anticlock wise command.

Figure 5: Drone in center of corridor tilted right

- If our drone lie center of corridor then bisector make angle equal to 90° degree but tilted toward right in this case intersection point of bisector and horizontal axis of image lies first half of the image then we give rotate anti-clock wise command.

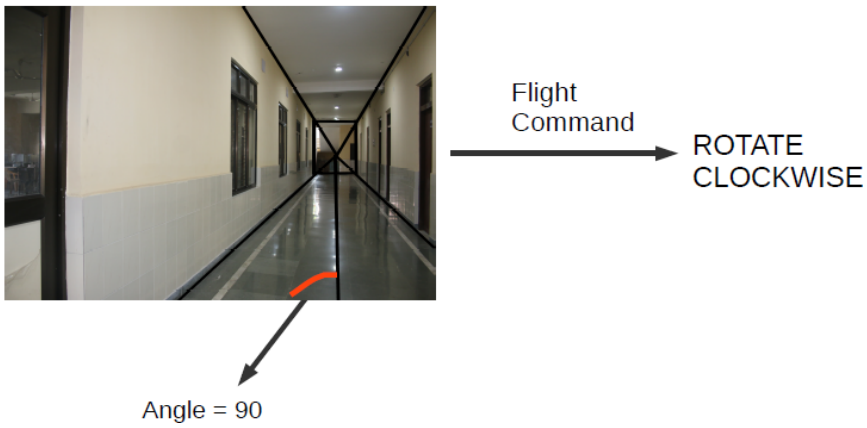

Figure 6: Drone in center of corridor tilted left

- If our drone lie center of corridor then bisector make angle equal to 90° degree and look toward straight in this case intersection point of bisector and horizontal axis of image lies middle of the image then we give go forward command.

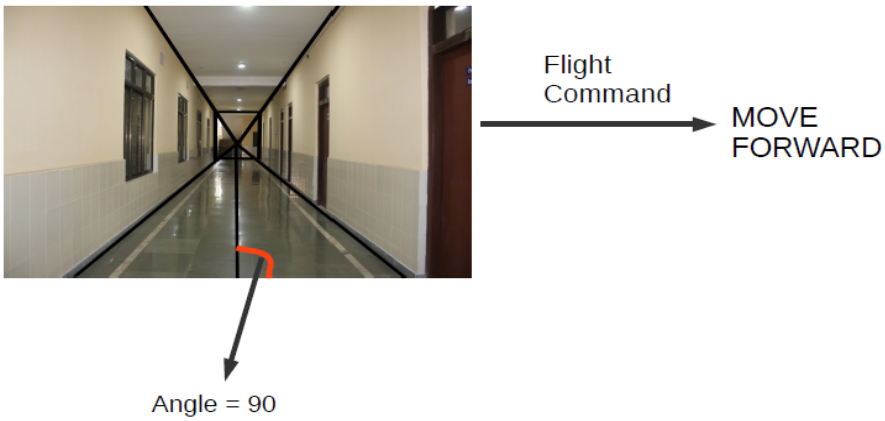

Figure 7: Drone in center of corridor looking straight

Algorithm 1: An algorithm

Input: IMAGE,height,width
Result: move command for drone

```
θ=Algorithm2(IMAGE);
if θ < 90 then
    right shift drone ;
else if θ > 90 then
    left shift drone ;
else
    d=Algorithm3(IMAGE);
if d < width/2 then
    rotate anti-clock wise ;
else if d > width/2 then
    rotate clock wise;
else
    move forward ;
```

Algorithm 2: Algorithm for finding angle

Input: IMAGE
Result: Angle between bisector of plane and horizental axis of image

```
Normalize pixal between 0 to 1;
RGB to BGR conversion;
Normalize with mean and standard deviation of Image Net dataset.;
θ = trained_weights(IMAGE);
return θ;
```

Algorithm 3: Algorithm for finding intersection point

Input: IMAGE
Result: Intersection point of bisector of plane and horizental axis of image
1 Normalize pixel between 0 to 1;
2 RGB to BGR conversion;
3 Normalize with mean and standard deviation of Image Net dataset.;
4 d = trained_weights(IMAGE);
5 **return** d;

In Algorithm 2 and Algorithm 3

- In step 1 normalize the pixels values between 0 to 1 by dividing 255.
- In step 2 convert RGB image BGR used model DenseNet work on BGR images.
- in step 3 normalize image with mean and standard deviation of Image NET dataset RGB mean is [0.406,0.456,0.485] and RGB standard deviation is [0.225,0.224,0.229].
- In step 4 trained_weights is trained Convolution neural network which return predicted value.

6 Data Set Creation

The second phase of the research we create dataset for our deep architecture. we have covered around 63 corridors. We have used augmentation technique to increase the list of the corridor i,e. we have made our drone to capture images from both the ends of the same corridor.

For converting the video taken by the drone camera into active input for our deep network, we have taken frames from the video of the corridor. The overall approach is take out around as many frames from the video and randomly select it for training and test set. Our drone is capable of extracting 30 frames per second. We presently dealing with around 35k images as trainset and around 600 images for testset. Some of our extracted frames from the video of the corridor are as follws:

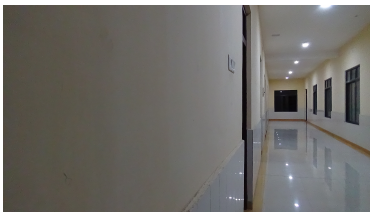

Figure 8: Atleft aligned to left

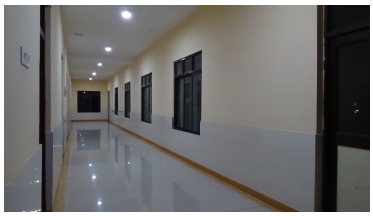

Figure 9: at Left aligned to right

Figure 10: At left aligned to center

The main motive of the data is to capture all possible location of the corridor such that neural network can detect whether the drone is at the left side of the corridor or at middle or at right side. Again there is a possible chance that at left side or right or at middle the drone is not facing straight or simply tilted. To overcome this two challenges we are using our self made concept called line bisector method.

Figure 11: At right aligned to left

Figure 12: at right aligned to right

Figure 13: At right aligned to center

Figure 14: At center aligned to left

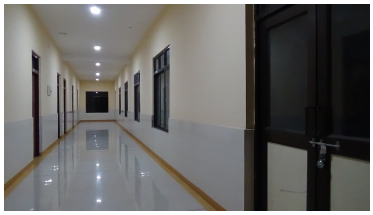

Figure 15: at center aligned to right

Figure 16: At center aligned to center

Through some analysis we found that when the drone is at the left side irrespective of its tilted position the angle between the axis of the floor of the corridor and the horizontal axis is greater than 90° while when the drone is at right side fo the corridor the angle is less than 90° ,also when the drone is at the center irrespective of its tilt this angle is 90°. This feature is kept effective as label for the trainset and testset.

7 Working Model

Coming to the implementation perspective, we have used DenseNet161[3] for our regression to estimate the angle between the bisector of the corridor and the horizontal axis of image. Since DenseNet[3] is a winner of 2016 in Imagenet Large Scale Visual Recognition Challenge(ILSVRC) 2016, and a state of art in present scenario and performing very well around various computer vision challenges. We use DenseNet-161[6] as our working model with some modification.DenseNet is used for classification problem, but we will use transfer learning technique for using it as regression technique. We have augmented three sequential blocks consisting of BatchNormalization followed by ReLU and Convolution Layer also one FC layer is also added.

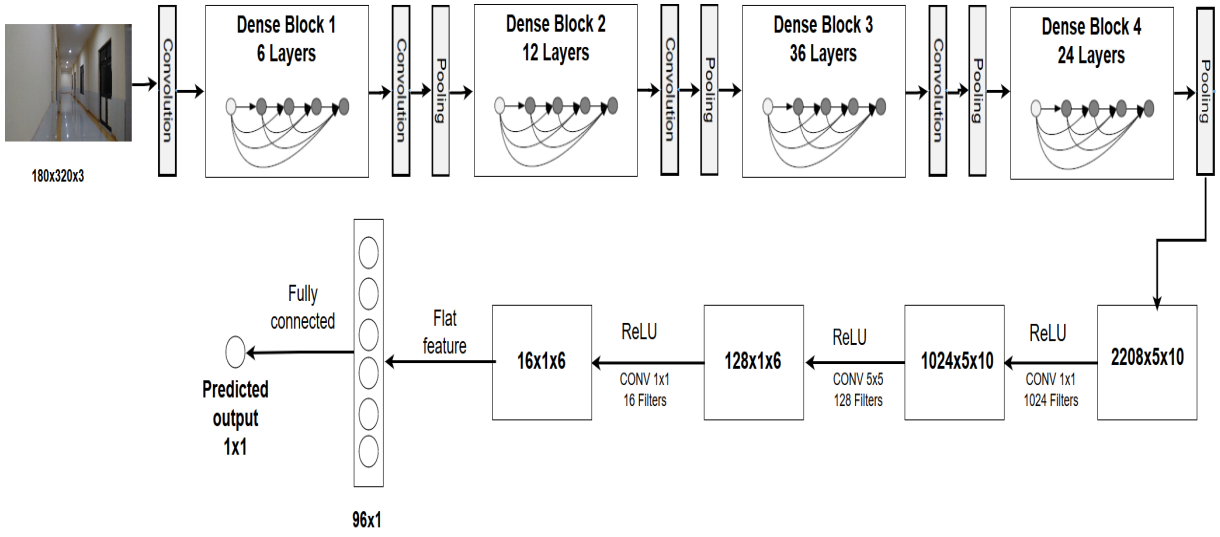

Figure 17: Using DenseNet-161 with Transfer Learning

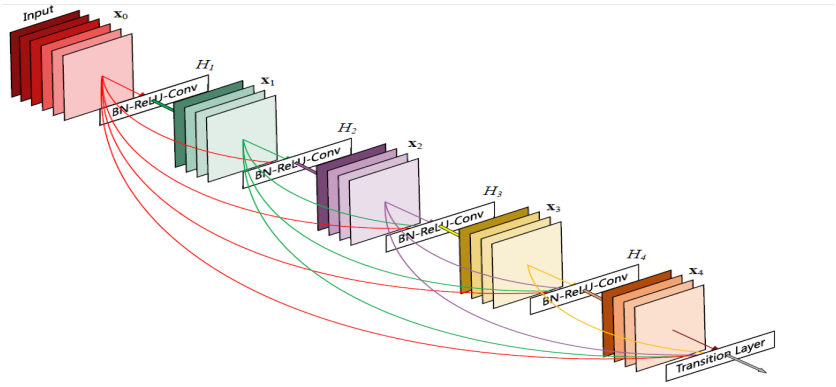

Figure 18: Block of DenseNET

8 System Setup

Our primary quadcopter is the AR Parrot Drone(figure 19). This quadcopter is currently available to the general public. The drone contains a single forward-facing camera, an ultrasound sensor for measuring ground altitude. Commands and images are exchanged via a WiFi connection between our host machine and the drone. The WiFi connection has a signal range of 250 meters (0.155 miles). From the stream connection, we receive an image of resolution of 640x368 pixels. We run our network on the host machine 32GB RAM ,XEON processor 3.33 GHz (NVIDIA GeForce GT 1080,11GB memory), running Ubuntu 14.04.

Figure 19: A R Parrot Drone

9 Experiment for finding angle between bisector and horizontal axis of plane

The angle between bisector and horizontal axis of plane help us to find position of drone in corridor. We use the three different loss functions for train our network Euclidean loss , Berhu loss[1, 2] and Mean Absolute loss.With the help of these function compute the loss of every batches and back propagate into to the network to update the the weights of network.

- We use 35000 images for training and 600 images for test
- We use three loss functions for training

9.1 Euclidean Loss

The Euclidean loss computes the sum of squares of differences of predicted values and target values.

$$Euclidean\ Loss(\hat{y}, y) = \frac{1}{2n}\sum_{t=1}^{n}(\hat{y}_t - y_t)^2$$

Where $\hat{y}$ and y are predicted and ground-truth value vector of batch respectively.

9.1.1 Convergence Graph using Euclidean Loss

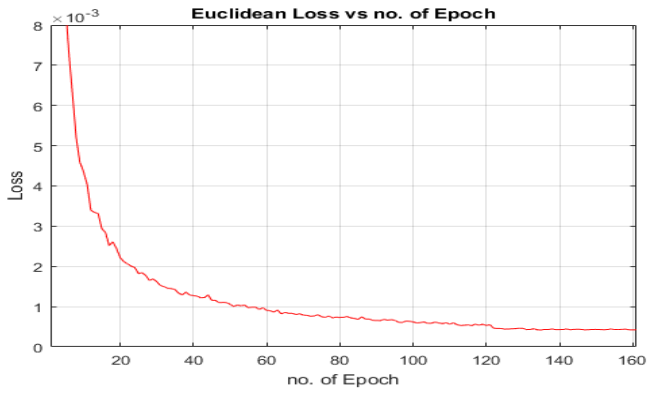

Figure 20: Euclidean loss vs. No. of epoch

9.2 berHu Loss[1, 2]

The berHu loss[1, 2] poses a good trade-off between the L_1 and L_2 norm so as to provide higher weights to pixels having higher residual.The berHu[1, 2] loss between the predicted value and the ground-truth value can be given as,

$$berHu(\hat{y}, y) = \begin{cases} L_1(\hat{y}, y), & \text{if } |\hat{y} - y| \in [-t, t] \\ \frac{1}{2t}(L_2(\hat{y}, y) + t^2), & \text{otherwise} \end{cases}$$

where

$$L_1(\hat{y}, y) = \sum_{t=1}^{n} |\hat{y}_t - y_t|$$

$$L_2(\hat{y}, y) = \sum_{t=1}^{n} (\hat{y}_t - y_t)^2$$

$$t = 0.2max(|\hat{y}_t - y_t|)$$

Where $\hat{y}$ and y are predicted and ground-truth value vector of batch respectively.

9.2.1 Convergence Graph using berHu Loss

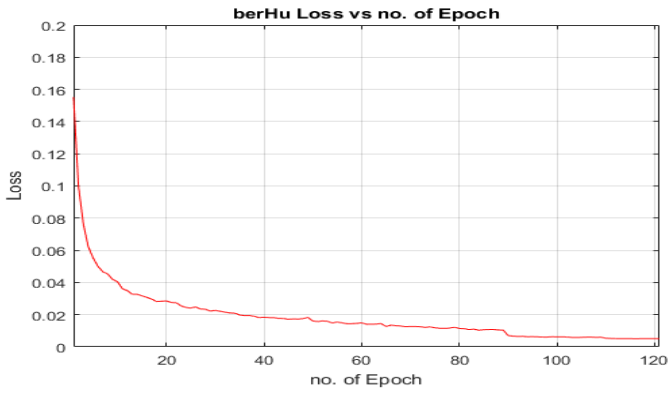

Figure 21: berHu loss vs. no. of epoch

9.3 Mean Absolute Loss

Mean Absolute Loss is a measure of difference between two continuous values

$$Mean\ Absolute\ Loss(\hat{y}, y) = \frac{1}{n}\sum_{t=1}^{n}|\hat{y}_t - y_t|$$

Where $\hat{y}$ and y are predicted and ground-truth value vector of batch respectively.

9.3.1 Convergence Graph using Mean Absolute Loss

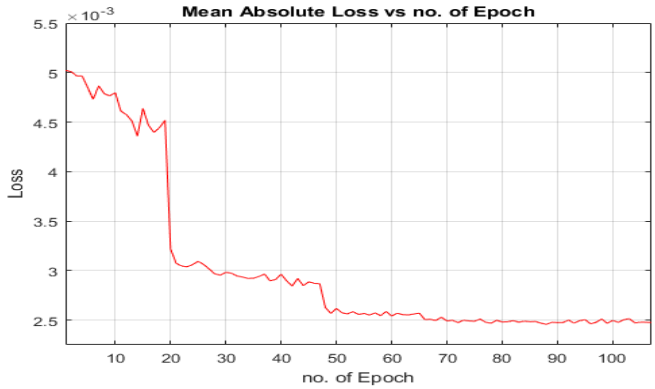

Figure 22: Mean Absolute loss vs. no. of epoch

9.4 Results

We use three error metric for measure the accuracy of model on test images.The error metrics are Mean Square Error,Mean Absolute Error and Mean Relative Error.

$$Mean\ Square\ Error(\hat{y}, y) = \frac{1}{n}\sum_{t=1}^{n}(\hat{y_t} - y_t)^2$$

$$Mean\ Absolute\ Error(\hat{y}, y) = \frac{1}{n}\sum_{t=1}^{n}|\hat{y_t} - y_t|$$

$$Mean\ Relative\ Error(\hat{y}, y) = \frac{1}{n}\sum_{t=1}^{n}\frac{|\hat{y_t} - y_t|}{y_t}$$

Where $\hat{y}$ and y are predicted and ground-truth value vector of test images respectively.

Result for angle prediction in Degree			
Loss Functions	Mean Square Error	Mean Absolute Error	Mean Relative Error
Euclidean Loss	4.254	11.016	8.978
berHu Loss	2.973	8.275	6.799
Mean Absolute loss	**0.057**	**1.326**	**1.087**

Table 2:

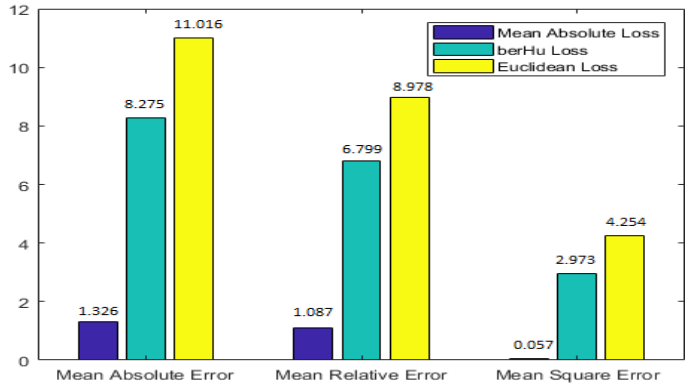

Figure 23: Comparison between all three loss functions

- Mean Absolute loss give least error.
- We can easily conclude that Mean Absolute loss suitable loss function for our model .

10 Experiment for finding intersection point of bisector and horizontal axis of plane

Intersection point of bisector and horizontal axis of plane help us to find tilted position of drone in corridor.We use Mean Absolute loss functions for train our network.With the help of these function compute the loss of every batches and back propagate into to the network to update the the weights of network.

- We use 35000 images for training and 600 images for test
- We use three loss functions for training

10.1 Mean Absolute Loss

Mean Absolute Loss is a measure of difference between two continuous values

$$Mean\ Absolute\ Loss(\hat{y}, y) = \frac{1}{n}\sum_{t=1}^{n}|\hat{y}_t - y_t|$$

Where $\hat{y}$ and y are predicted and ground-truth value vector of batch respectively.

10.1.1 Convergence Graph using Mean Absolute Loss

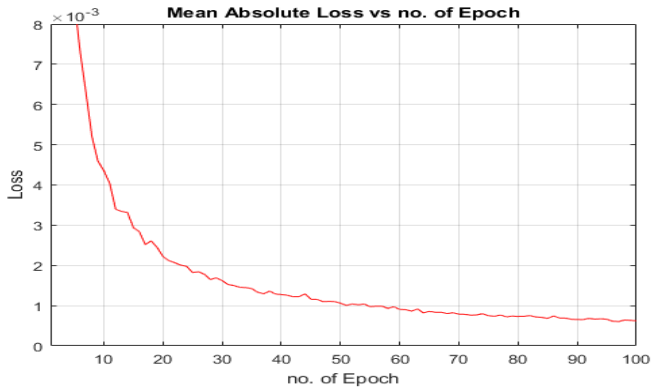

Figure 24: Mean Absolute loss vs. no. of epoch

10.2 Results

We use three error metric for measure the accuracy of model on test images.The error metrics are Mean Square Error,Mean Absolute Error and Mean Relative Error.

$$Mean\ Square\ Error(\hat{y}, y) = \frac{1}{n}\sum_{t=1}^{n}(\hat{y}_t - y_t)^2$$

$$Mean\ Absolute\ Error(\hat{y}, y) = \frac{1}{n}\sum_{t=1}^{n}|\hat{y}_t - y_t|$$

$$Mean\ Relative\ Error(\hat{y}, y) = \frac{1}{n}\sum_{t=1}^{n}\frac{|\hat{y}_t - y_t|}{y_t}$$

Where $\hat{y}$ and y are predicted and ground-truth value vector of test images respectively.

Result for intersection point prediction in no. of pixels			
Loss Functions	Mean Square Error	Mean Absolute Error	Mean Relative Error
Mean Absolute loss	0.032	2.440	1.557

- Mean Absolute loss function give satisfactory result for intersection point prediction.

11 Conclusion

Experiment observation is quite good to find at least the location of the drone in the corridor.When the location of the drone is detected,and the drone is not at the center,our approach is to make angle between the axis of the floor and horizontal axis.such that drone will be able to look straight and then the next approach will be to bring drone to the center of the corridor about the horizontal axis. We see from the above experimental result shown that our modified DenseNet with Mean Absolute Loss will give better result for finding angle of the bisector in image plane and intersection point of bisector and horizontal axis of image.

12 Future work

My future work will be deploy our learned network in Drone.We will use the approach discussed above to navigate drone in indoor environment basically at the corridor.

References

[1] Yokila Arora, Ishan Patil, and Thao Nguyen. Fully convolutional network for depth estimation and semantic segmentation.

[2] Laurent Zwald and Sophie Lambert-Lacroix. The berhu penalty and the grouped effect. *arXiv preprint arXiv:1207.6868*, 2012.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[4] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[6] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, volume 1, page 3, 2017.