UAV Pose Estimation in Indoor Corridor Using Monocular Vision and Deep Learning

Sachin Verma

Department of Computer Science and Engineering
National Institute of Technology Rourkela

UAV Pose Estimation in Indoor Corridor Using Monocular Vision and Deep Learning

Thesis submitted in partial fulfillment

of the requirements for the degree of

Master of Technology

in

Computer Science and Engineering
(*Specialization*: *Computer Science*)

by

Sachin Verma

(Roll Number: 216CS1147)

based on research carried out

under the supervision of

Prof. Pankaj K. Sa

May, 2018

Department of Computer Science and Engineering
National Institute of Technology Rourkela

Department of Computer Science and Engineering
National Institute of Technology Rourkela

Prof. Pankaj K. Sa
Associate Professor

May 25, 2018

Supervisor's Certificate

This is to certify that the work presented in the thesis entitled *UAV Pose Estimation in Indoor Corridor Using Monocular Vision and Deep Learning* submitted by *Sachin Verma*, Roll Number 216CS1147, is a record of original research carried out by him under my supervision and guidance in partial fulfillment of the requirements of the degree of *Master of Technology* in *Computer Science and Engineering*. Neither this thesis nor any part of it has been submitted earlier for any degree or diploma to any institute or university in India or abroad.

_____
Pankaj K. Sa

dedicated to my 'family' & 'friends'

Declaration of Originality

I, *Sachin Verma*, Roll Number *216CS1147* hereby declare that this thesis entitled *UAV Pose Estimation in Indoor Corridor Using Monocular Vision and Deep Learning* presents my original work carried out as a postgraduate student of NIT Rourkela and, to the best of my knowledge, contains no material previously published or written by another person, nor any material presented by me for the award of any degree or diploma of NIT Rourkela or any other institution. Any contribution made to this research by others, with whom I have worked at NIT Rourkela or elsewhere, is explicitly acknowledged in the dissertation. Works of other authors cited in this dissertation have been duly acknowledged under the sections "Reference" or "Bibliography". I have also submitted my original research records to the scrutiny committee for evaluation of my dissertation.

I am fully aware that in case of any non-compliance detected in future, the Senate of NIT Rourkela may withdraw the degree awarded to me on the basis of the present dissertation.

May 25, 2018
NIT Rourkela

Sachin Verma

Acknowledgment

Though an individual contribution, this M.Tech. thesis has actually been benefited in various ways from several people. The list may be finite, however, their continued support is innumerable.

The supervision of Prof. Pankaj Kumar Sa has inspired me to go beyond my limits and improve my skills to analyze the feasible solutions for the research challenges.

The profound insight and the invaluable advise of Dr. Sambit Bakshi has been a source of motivation for nourishing my intellectual maturity.

I am also grateful to Prof. Bibhudatta Sahoo for his unfailing support and assistance. Without his superior knowledge and experience, the Project would like in quality of outcomes, and thus his support has been essential.

Overwhelming thanks to Ram Sir and Nayan Sir, doctorate candidate of our lab, for their guidance to explore my knowledge in learning the art of new technologies.

I especially thank to my friends for their continuous encouragement and generosity throughout my masters research.

Last but not the least, my heartfelt thanks to my family for their unconditional love and support. Words fail me to express my gratitude to my beloved parents, who sacrificed their comfort for my betterment.

May 25, 2018
NIT Rourkela

Sachin Verma
Roll Number: 216CS1147

Abstract

Unmanned Aerial Vehicles (UAV's, commonly referred as drone) in recent times have acquired large attention in research community. Due to their ability of agility, drones have got potential industrial or civil task.

The control command for any UAVs in transit depends on the several factors, for e.g. position, velocity and orientation of the vehicle. These factors specification are not always available directly. Majority of the past research measured these factors using dedicated sensors and sophisticated hardwares. We proposed a model that uses only vision cue to automate the pose correction of drones in indoor flat corridor. The intrinsic features of the environment of the transit, earlier obtained by dedicated hardwares, in our case is obtained from the image of the environment using deep, feed-forward artificial neural networks called as convolutional neural network. Our approach is effective enough to enable any (robotic autonomous) device, to localize itself to face straight when it is at the centre of the navigating environment.

Keywords*: *Pose Estimation*; *Localization*; *Deep learning*; *Indoor Corridor*; *Monocular Camera*; *Convolutional neural network*; *DNN*; *Regression

Contents

Supervisor's Certificate ii

iii

Declaration of Originality iv

Acknowledgment v

Abstract vi

List of Figures ix

List of Tables x

1 Introduction 1

1.1 Background . . . 1

1.1.1 Parrot AR.Drone . . . 2

1.1.2 Technical Parameter . . . 2

1.2 Challenges . . . 4

1.3 Current State of the Art . . . 4

1.4 Motivation and Objectives . . . 5

1.5 Contribution . . . 6

1.6 Thesis Layout . . . 6

2 Overview of Deep Learning 8

2.1 Convolutional Neural Network (ConvNet/CNN) [1] . . . 9

2.2 Recent CNN Models . . . 11

2.3 Transfer Learning in Deep Learning . . . 13

3 Drone Pose Estimation using Deep Learning Models 15

3.1 Solution Hypothesis . . . 15

3.2 Proposed Methodology . . . 15

3.3 Localization Algorithm . . . 16

3.4 Proposed Architecture . . . 19

3.5 System Setup 20
3.5.1 Hardware Platform 20
3.5.2 Dataset 21
3.6 Performance Evaluation 23
3.6.1 Train Set Result 23
3.6.2 Test Set Result 26

4 Discussion and Conclusion **29**

References **31**

List of Figures

1.1 Degree of freedom for drone 3

2.1 Neurons Attached in 3 Dimensions 10
2.2 Convolutional Layer 10
2.3 Max Pool Layer 11
2.4 Fully Connected Layer 11
2.5 CNN Model 11
2.6 AlexNet [1] Model 12
2.7 Inception Block 13

3.1 Drone at Center tilted Left 17
3.2 Drone at Center facing Straight 17
3.3 Drone at Center tilted Right 18
3.4 fig explains the role of working model 19
3.5 images captured from the onboard camera of drone when it is at the center of the corridor. The first, second and third columns show the images, when the UAV is left, center and right tilted, respectively. 22
3.6 labelled images presented in fig.3.5 The first, second and third columns show the images, when the UAV is left, center and right tilted, respectively. The red line is drawn in order to find the ground truth. 22
3.7 image captured from onboard camera is flipped 23
3.8 image captured from onboard camera is zoomed variably 23
3.9 figures depicts the convergence graph for the different model used in the experiment. From top to bottom the sequence of the model is as follows (a)AlexNet, (b)DenseNet-161, (c) DenseNet-201, (d) Inception-V3, (e)ResNet-101, (f) ResNet-101, (g) ResNet-201 25

List of Tables

3.1 Architecture of Different Models used for Pose Estimation 20
3.2 Hardware Used to Perform Experiment 21
3.3 Error in Test Set Distance Prediction 26
3.4 table presenting some of the result of test set by our trained DenseNet-161 . 27

Chapter 1

Introduction

For a couple of years we have seen a growing interest in developing and utilizing unmanned aerial vehicles (UAV's commonly referred as drone or quadcopters). They have found many applications such as search and rescue, aerial surveillance, military operations and film making, just to name a few. Historically, UAV's were, and sometimes still are, called drones because they often performed primitive and repetitive tasks. More difficult tasks needed the assistance of a human operator. Nowadays, growing importance is being assigned to the task of creating unmanned and more autonomous robots capable of doing yet even more demanding tasks.

In this thesis we attempted to carry out the task of autonomous pose correction for drone in flat corridor in indoor corridor surroundings of any building or mall, etc.

1.1 Background

UAV's are aircraft without any on-board pilot. They are either remotely controlled or by humans,in which case it is termed as remotely piloted vehicles (RPVs) or are programmed so to take on flight command responding according to navigating surrounding, in that case, it is termed as automated drones. UAV's are capable of sustainable flight in hostile regions where human reach are limited. In this present century, technology enabling UAV's to expand its role in almost all area of aviation.

One of the biggest advantages of this robotic platform is its ease of use and its flying characteristics which are similar to those of conventional helicopters. Unlike helicopters, quadcopters are equipped with four horizontally aligned rotors. Quadcopters take off and land vertically, hover in place and are able to fly any curve in 3D space. Together with their small size they are especially suitable for flying indoors and in dense environments. Quadcopters are inherently unstable and require sophisticated stabilizing control assistance to keep them manageable.

1.1.1 Parrot AR.Drone

The Parrot AR.Drone [2] was initially meant to serve as a toy for augmented reality games. Humans were the operator of these quadcopters with some device like remote controller or mobile. However, due to its low cost and availability it turned into an interesting platform for various research projects. In this thesis we will exploit some of the possibilities it opens up. First, we will take a look at the quadcopter's terminologies, then we will see what kind of hardware and software it has as well as all of its onboard sensors.

1.1.2 Technical Parameter

The Parrot AR.Drone quadcopter 2.0 [2] is propelled by four electric rotors with power of 14.5W each. These four rotors are framed in the corners of a cross supported by a lightweight carbon-fiber construction which lends it resistance against mechanical wear out. Each rotor has a dedicated 8 MIPS AVR microcontroller for better flight characteristics. The quadcopter is capable of flying at speeds of up to 10m/s. The weight of the whole device is 380g or 420g, depending on the hull of the quadcopter, which is either for indoor or outdoor flight. Depending on the thrust given from each rotor, the quadcopter can move in several directions. The first and the easiest one is when all rotors are rotating at the same speed. Then, at least in the ideal case, the quadcopter is either hovering in place, ascending or descending, depending whether the thrust is sufficient enough to overcome the gravitational pull. Apart from that, the quadcopter can rotate around its vertical axis. This can be achieved by simply increasing the speed of either of the two pairs of diagonally opposite rotors and decreasing the speed of the second pair of opposite rotors, which results in either clockwise or counterclockwise rotation. Furthermore, the quadcopter adjusts its pitch or roll by adding more power in one rotor and decreasing power in the opposite rotor, tilting the quadcopter and flying in the desired direction. The rotational angles are schematically depicted in Figure 1.1 The quadcopter's internals consist of a 1GHz 32 bit ARM Cortex A8 processor with 1GBit of DDR2 RAM running at 200MHz. Additionally, a Wi-Fi b/g/n device allows the creation of an ad-hoc network for external connection and communication with the quadcopter. The system is running a GNU/Linux with kernel version 2.6.32 with a versatile command line program BusyBox. The control program operating the quadcopter is distributed as a binary executable. For faster video processing the quadcopter has a dedicated digital signal processing (DSP) unit TMS320DMC64x running at 800 MHz. The quadcopter is controlled via WLAN through a set of commands sent from the ground to the quadcopter. The quadcopter acts upon the environment with its rotors. On the other hand, it has various sensors by which it perceives the surrounding environment. It disposes with two cameras. The frontal HD 720p camera has a resolution of 1280×720 pixels running at 30 fps with wide angle lens of 92 degrees. However, one of its many downsides is the fact that the images are subject to significant distortions and especially while in flight the images are

somewhat heavily blurred. Apart from the frontal camera there is also a camera facing downwards. It has a resolution of 480 × 360 pixels and is running at 60 fps. This camera is utilized internally by the Parrot AR.Drone 2.0 onboard software to enhance the quadcopter's stability and resilience against drift. Unfortunately, experiments have shown that this ability depends heavily on the texture of the surface. In poor conditions the quadcopter is subject to unwanted drift. The quadcopter is also equipped with a 3 axis gyroscope, which is able to measure the pitch, roll and yaw angles of the flying quadcopter with rotational speed up to 2000deg/s. The onboard 3 axis accelerometer measures acceleration in all three directions with precision of up to ± 0.05g.

For more accurate measurements the quadcopter is also equipped with a 3 axis

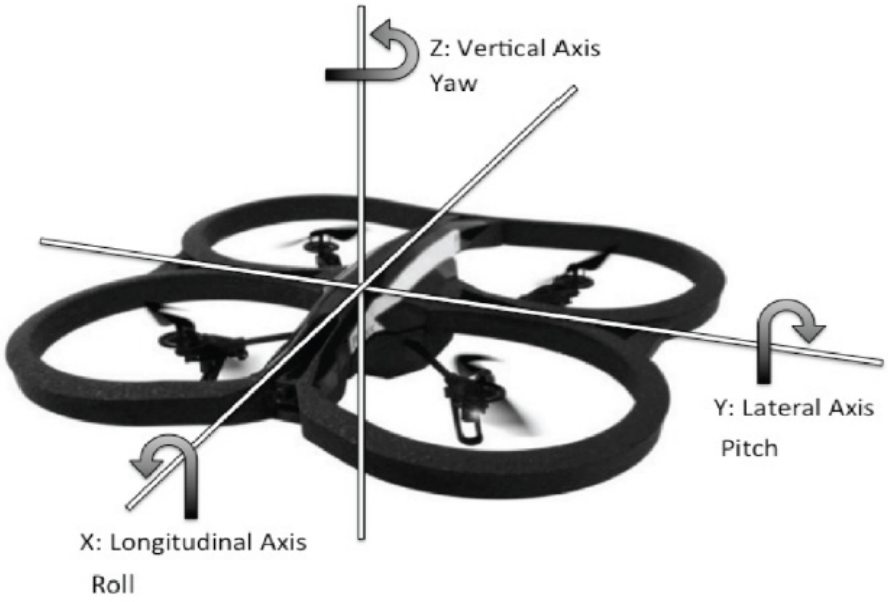


Figure 1.1: Degree of freedom for drone

magnetometer with precision of up to 6 degrees. Results from the magnetometer are combined with those from the gyroscope yielding more accurate results. To estimate the current flying altitude, the quadcopter utilizes two onboard sensors. The first one is an ultrasound sensor suitable for measuring altitude just above the ground (heavily exploited in the takeoff and land manoeuvres). The second sensor is a pressure sensor which aids in measuring the altitude of a quadcopter several feet above the ground where the ultrasound sensor does not give reasonable estimates. Precision of this sensor is up to 10Pa. Basically, the maximum altitude is limited only by the reach of the Wi-Fi signal. The standard battery provided along with the Parrot AR.Drone 2.0 has enough capacity for about ten minutes of flight.

1.2 Challenges

Strict constraint weight and power consumption in quadcopters, the choice of sensors, algorithms inflicts a stiff challenge of performance. To enforce some degree of automation ego motion and scene estimatiion are required for quadcopters, but which is not the case with any vehicular robots, which makes the quadcopter automation task a bit challenging.

Limited Payload: Quadcopters are highly constrained by weights. It is the pressure created by the rotors to keep the quadcopter hovering. During different context of application, like navigating a drone autonomously, use of different specialized sensors, cameras or lens increases weights, which makes it sensible to create severe hazards during flight.

Limited Power: Quadcopters have battery attached onboard from which it obtains its power supply. The experimental process might not put burden on the onboard power supply. Hence quadcopter equipments must be light in terms of weight and computation to enable the quadcopters to perform effectively.
These challenges inhibits the use of best component which degrades the quality of data, which further create challenge for algorithm selection.

Constant Motion: Comparing to other robots(like vehicular robot), the one major constraint comes into play is, drones never pause their navigation before acquire data from the onboard devices. This causes a problem of instability in data acquisition and further attenuate data quality.

Degree of Freedom: For robots like ground vehicles , they enjoy only two degree of freedom, they can move to any point in two dimension. In this case only heading i.e. yaw angle is to be controlled. But quadcopters enjoys 3 degree of freedom i.e. yaw as well as roll and pitch. This requires planing to be done in 3D space.

1.3 Current State of the Art

We here, describe briefly about the progresses made in the field of UAV system, while pointing to applicability to working condition. Previous work on UAV navigation can be divided into several categories. These achievements are categorized in following ways :

Tracking with Motion Capture Systems: This category deals with accurate quadcopter control and their workings are quite accurate [3–5]. All these approaches uses motion capturing system. These methods lean on advanced [6–8] tracking scheme. Hence their use

are restricted to predefined controlled lab environment.

Localization and Mapping with Depth Sensors: Some techniques use range sensors like infrared sensors, or RGB-D sensors [9], or laser rangefinder. Bry et al. [10] presented a state estimation method using laser rangefinder and inertial measurement unit to navigate autonomously in GPS [10] denied environment. Roberts et al. [11] used a single ultrasonic sensor and some infrared sensors and presented autonomous navigation which are even capable of collision avoidance.

Localization and Mapping with Cameras: SLAM technique uses range or visual sensors to design 3-dimensional map [6–8] of the unfamiliar indoor environment in transit of the flights, along with finding its own location on the map [6–8]. Bachrach et al. [12] use a laser rangefinder for SLAM implementation to generate the map for the unknown environments. Celik et al [13] showed indoor navigation using monocular camera [14, 15] with the help of SLAM technique. In outdoor navigations are localized using of GPS[16]. Ram et at. [17] proposed drone navigation in corridor which uses vanishing point localization for collision avoidance.

Navigation with Learning Based Algorithm: ALVINN [18] showed how artificial neural network (ANN) [19] mimics a human driver and efficiently performed autonomous vehicle driving. Kim[20] shown navigation in indoor enviroenment as a classification task with deep learning.

1.4 Motivation and Objectives

Accurate drone localization is a key for its navigation. A large number of experiments in this field have been performed till date. The current state-of-the-art methods provided good progress in challenging area, however the dependency of previous methods on sophisticated hardwares makes the process bulky and slow, which in turn puts undue burden on power consumption. These factors lay the foundation of our first research objective.
First Objective: propose a comprehensive method that must make use of minimum support from hardwares.
Majority of methods for navigation relies on heavy sensors or depth or stereo camera to achieve self-governing navigation at the cost of bulky computation and high power usance. Approaches using bulky sensor and specialized cameras are inefficiently degrading the drone's own performance and inhibits to fly with its full capability. One of the famous techniques called Simultaneous Localization and Mapping(SLAM) [7] aids autonomous navigation process using 3D map generation of the environment of transit. This approach is also not feasible from drone perspective as it is computationally very expensive.These

aspects of bulky processing culminates in the genesis of the next objective

Second Objective: proposed method must have fast receive and response process and also the high computation part must be taken away from the quadcopter's side for efficient power utilization.

Localization of moving robot to face straight requires precise knowledge of its present pose and periodic update over time. It suffer from various challenges such as the process must be fast enough as the device must not be allowed to hover at every tilted pose. Hence the overall process must be fast and efficient.

Since we are dealing with indoor environment, these surrounding majority made up of walls and contains less features,also it cannot take advantage of GPS [16]. These hurdles take us to our next objective.

Third Objective: the proposed method must make use of the attributes of the environment of transit to localize quadcopter.

The problem with indoor navigation is, we cannot rely upon the use of GPS signal at any point of navigation in any indoor environment, as GPS signal loosens its exactness and strength with the shields. Also during cloudy or rainy day one can even not observe GPS signal.

1.5 Contribution

Our work presented here, provides two main contribution.

1. We presented a working model for autonomous localization of flying device (i.e, drone) whick makes the device to face straight when it is at the middle of the corridor with the use of only one onboard monocular camera.

2. We created our own dataset for training our deep neural network, which captures around all possible scenario for the drone to localize itself in a flat corridor where the class label is the *distance* of the intersection of the horizontal axis of the image and the axis along the floor of the corridor passing from the device situated at the centre of corridor, in the image captured from onboard camera.

1.6 Thesis Layout

This thesis is organized into four chapters, where each of them portrays the details of the research in different layers. The layout of the thesis is enumerated as follows —

Chapter 2: Overview of Deep Learning This chapter briefly explains various the components of a famous deep learning class called convolutional neural network [1] and

their working. It also put forward various state-of-the-art ILSVRC [21] winner models features. Lastly, it includes a short intro to transfer learning technique, which describes the way to bring into use these state-of-the-art models pre-trained weights for own purpose.

Chapter 3: Drone Pose Estimation using Deep Learning Models In this chapter, we presented the whole experiment of our thesis from pose estimation algorithm to the output and accuracy of the the working models.

Chapter 4: Discussion and Conclusion This chapter wraps up our thesis by highlighting the insights and the concluding remarks of the presented models. Beside this, the scope for further improvements is also enumerated towards the end.

Chapter 2

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.

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 CPU's, 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 GPU's are optimized for very fast matrix multiplication. Machine learning has successfully grown to be a major computer science discipline with extensive applications in science and engineering for many years. While machine learning has become leading within the field of AI, it does have its problems. It is particularly time consuming and is still not a true measure of machine intelligence as it relies on human resourcefulness to come up with the abstraction that allow computer to learn. A primary challenge to machine learning is the lack of adequate training data to build accurate and reliable models. Learning in machine learning applications depends on hand-engineering features where the researcher manually encodes relevant information about the task at hand and then there is learning on top of that. This contrasts with deep learning which tries and gets the system to engineer its own features as much as is viable. in many realistic situations. When quality data are in short supply, the resulting models can perform very poorly on a new domain, even if the learning algorithms are best chosen. Deep learning has different range of aplicabilty, these are as follows:

Unsupervised Deep Network : Unsupervised learning is the learning task of finding out a function that best describe the hidden structure from the data whose ground truth value data are not known. Since class labels for data are not available, we can not evaluate the performance. This learning computes joint probability of data distribution or Bayes rule.
Supervised Deep Network : Supervised learning is the learning task of finding out a function that best describe the hidden structure from the data by taking into account the ground truth value. Here, since class labels for data are available, there exist way to evaluate the performance. This learning computes posterior distributions of classes conditioned on the visible data. They provide discriminative power to model.
Hybrid Deep Network : Here the objective is discrimination aided by result of unsupervised deep networks. Also, the case is possible when the discriminative feature of supervised technique aid the performance of deep generative or unsupervised model.

2.1 Convolutional Neural Network (ConvNet/CNN) [1]

Convolutional Neural Network are similar to ordinary neural networks which are made up of neurons and have learnable weights and biases. The wights are multiplied to input to neuron and optionally follows non linearity. They continue to have loss function on their last layer. The difference between this network and ordinary neural network is that ConvNet use pixel value as input rather than feature vectors. This makes the forward function work efficiently and reduces parameters. Unlike ordinary neural network, neurons in ConvNet are arranged in 3 dimension. The two dimension refer to the spatial size of the filter while the third dimension refer to the no of channel of the input. The neurons at one layer is connected to small region of layer above it.

3D Volumes Of Neuron

Convolutional Neural Networks take advantage of the fact that the input consists of images and they constrain the architecture in a more sensible way. In particular, unlike a regular Neural Network, the layers of a ConvNet have neurons arranged in 3 dimensions: width, height, depth. (Note that the word depth here refers to the third dimension of an activation volume, not to the depth of a full Neural Network, which can refer to the total number of layers in a network.). The neurons in a layer will only be connected to a small region of the layer before it, instead of all of the neurons in a fully-connected manner.

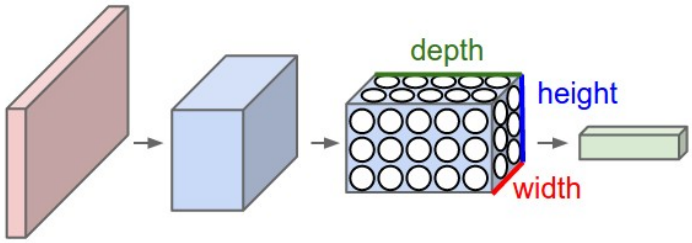


Figure 2.1: Neurons Attached in 3 Dimensions

Layers Used in ConvNet [1]

We use three main types of layers to build ConvNet [1] architectures: Convolutional Layer, Pooling Layer, and Fully-Connected Layer and some activation function. We will stack these layers to form a full ConvNet architecture.

Convolutional Layer : Convolutional layers apply a convolution operation to the input, passing the result to the next layer. The convolution emulates the response of an individual neuron to visual stimuli. Each convolutional neuron processes data only for its receptive field.

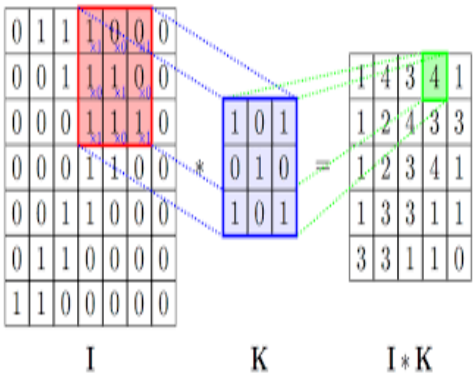


Figure 2.2: Convolutional Layer

Pooling : Convolutional networks may include local or global pooling layers, which combine the outputs of neuron clusters at one layer into a single neuron in the next layer. For example, max pooling uses the maximum value from each of a cluster of neurons at the prior layer. Another example is average pooling, which uses the average value from each of a cluster of neurons at the prior layer.

Activation Function : In neural networks, activation functions are used to enable the network to learn non-linear functions. There are several activation function e.g. Rectified Linear Unit (ReLu) which performs element wise and produce maximum among the 0 and the activation value i.e. *f(x)=max(0,x)* where x is the activation value.

Tanh is a hyperbolic tangent function which also applied element wise, and produce the hyperbolic tangent of the activation value i.e. $f(x) = \frac{exp^x - exp^{-x}}{exp^x + exp^{-x}}$ where x is the activation

value.

Fully Connected Layer :Fully Connected (fc)layers depicted in Figure 2.4, connect every neuron in one layer to every neuron in another layer. It is in principle the same as the traditional multi-layer perceptron neural network (MLP) [22].

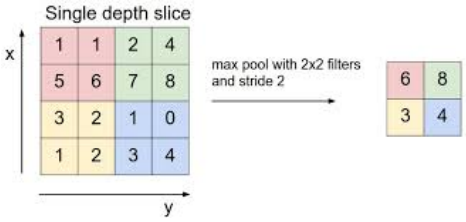


Figure 2.3: Max Pool Layer

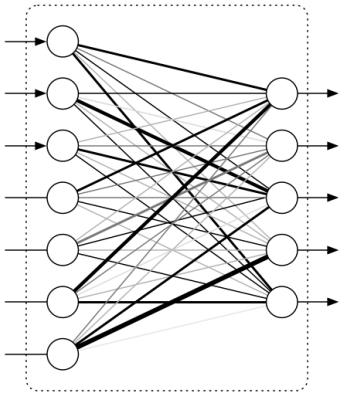


Figure 2.4: Fully Connected Layer

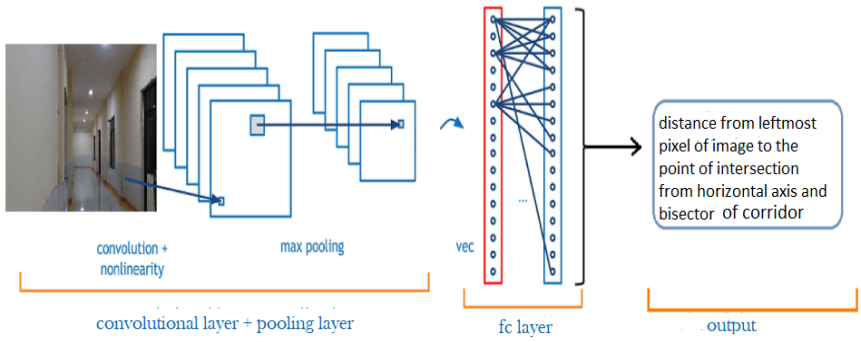


Figure 2.5: CNN Model

2.2 Recent CNN Models

This section will summarize important recent developments in the field of computer vision and convolutional neural network. Here are some of the most important networks that have used CNN [1] as their core part in learning model.

AlexNet [1](2012)

The paper, titled "ImageNet Classification with Deep Convolutional Networks", proposed a deep model in 2012 termed termed as AlexNet. This model was used in Imagenet Large Scale Visual Recognition Challenge (ILSVRC) [21] in 2012. For the first time in ILSVRC,

CNN model was used to achieve a top 5 test error to be 15.4%. Before this top 5 error was 26.2%.

The AlexNet [1] is composed of 5 convolutional layer, max pool layer, dropout layer and fully connected layer. The network was designed to classify among 1000 class labels.

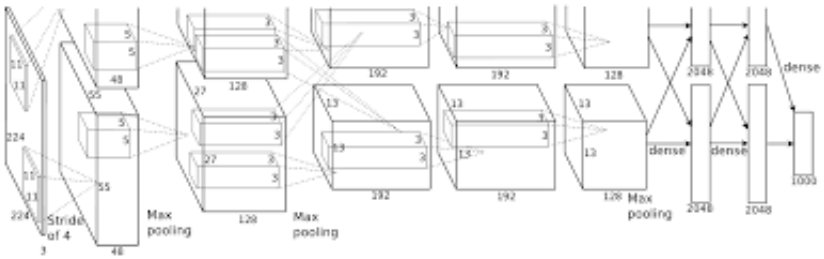


Figure 2.6: AlexNet [1] Model

ZFNet [23](2013)

In 2013, the winner of ILSVRC [21] is a network called as ZFNet [23] which achieved 11.2 % top 5 error rate. This network can be imagined as some fine tuning of the previous winner mode AlexNet [1]. Here, every layer of the trained CNN citekrizhevsky2012imagenet is attached to DeconvNet which has a path back to pixel value. This helps to cross-examine the activation at any point of time.

VGG Net (2014) [24]

This model is presented to ILSVRC 2014. It doesn't become the winner of the challenge but still its error rate achieved is 7.3% error rate. It is a 19 layer model. The main contribution made by this model was it used only 3×3 filter size throughout the network also kept the stride and padding strictly to 1.

Using 3×3 convolution instead of of bigger spatial size, the author depicted that using two 3×3 filter back to back mimics 5×5 and three mimics 7×7 filter size. This approach reduces the number of parameter in the network. The main notion depicted by this network is CNN have to have deep layers in order to achieve the hierarchical representation.

GoogLeNet [25] (2015)

This deep network is one of the coolest network in till now presented. This network was the winner of ILSVRC 2014 and the error rate of this network is 6.7%. This is a 22 layer non-linear network. It uses Inception block which makes it different from other network as

its allow to perform pooling operation or a convolution operation in parallel. It has 12× lesser no. of paremeter than Alexnet.

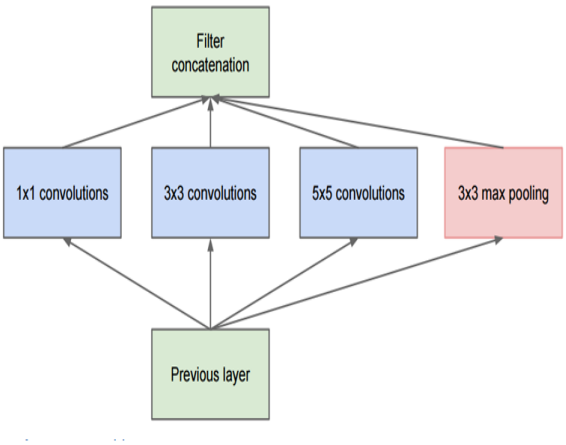


Figure 2.7: Inception Block

ResNet [26](2015)

ResNet became the winner of ILSVRC 2015 by reducing the top 5 error rate to 3.6%. The concept here is that activation generated by one layer of convolution, like conv-relu-pool, are augmented with the original image for the second layer input.
The main motive of doint this was the layers at any step of deep propagation must get full chance to access the features learned so far as the case till now, but also directly pull out features according to it.

DenseNet [27](2016)

Instead of concatenating the input directly to activation generated at any layer , DenseNet [27] concatenated the activation generated at any layer with the activation of all the previous layer.
In forward propagation, any layer get lower level features along with the higher layer features, and during backward propagation, allows all gradient to reach to respected place easily as all the layer are connected from one another.

2.3 Transfer Learning in Deep Learning

Transfer learning is a technique of using the knowledge of one area to another domain. Transfer learning is achieved by using the pre-trained weight on one dataset over other dataset by fine-tuning. The benefit of transfer learning is that it makes the learning fast by

providing the prior knowledge to the neural network.

Transfer learning is performed in many ways like freezing the low level features and retrain the high level features or by replacing last fc layer with new or by augmenting convolutional layer followed by last fc by replacing the old fc layer etc., depending on the size of the dataset and its similarity with the original dataset.

Chapter 3

Drone Pose Estimation using Deep Learning Models

Several approaches and their pros and cons were discussed in Section 1.3. This section summarize the the work we followed to achieve the goal of the experiment. A brief summary of the drone pose estimation is presented along with some images captured from the drone camera in order to validate the proposed algorithms.

3.1 Solution Hypothesis

The problem here is to by some means, mimic a experienced pilot's course of action to localize quadcopter safely. Manual programming for such tasks are quite complex. Several heuristics were used in several approaches which uses GPS [16], sensor or sophisticated cameras and others. But these approaches have their own advantages and disadvantages as explained.

Localization or navigation of any device in any surrounding need information of the attribute of the surrounding of transit for its pose estimation, but in our case these informations with onboard monocular camera is not available. To obtain the deep attributes, like intrinsic features of the surrounding for the tiltness of the quadcopter, we need deep neural network to learn these features from the input which here is the images captured from the monocular camera attached to the drone. Hence to learn deep feature from input image, we make use of famous deep learning class called ConvNet [1] which takes images as their input.

3.2 Proposed Methodology

Our approach of drone localization is bounded to correctify the tiltness of the drone while in transit and also when it is at the center of the corridor, such that the drone must face straight. We have presented here our algorithm which decide wether the tilted drone to yaw left or yaw right and is the foundation of our custom dataset. Also a brief description of deep learning architectures used for the estimation of tilted pose of the drone is presented here to

summarize the working model of our experiment.

3.3 Localization Algorithm

To enable quadcopters to localize autonomously to face straight, we came up with a novel approach that aid a quadcopter to localize itself in indoor scenario with only a single monocular camera. In our experiment, the input is solely a RGB image captured from the camera attached to the quadcopter.
The whole idea of the experiment is that any object lying on any flat corridor, the axis passing along the surface of the corridor through the object will subtend right angle to the horizontal axis passing through the object. This observation is valid in both 3D world as well as in 2D world. This forms the basic intuition our experiment.

Applying this to our drone localization, things remain same and we use this property to enable the drone to navigate as well as localize in flat corridor. When the drone is at the center of the flat corridor, the axis passing along the floor of the corridor like bisecting the area of the floor of the corridor into two half(we use the term bisector for this axis) will make 90° with the horizontal axis of the image captured from the drone camera.
Similarly we have noted that this angle of intersection is less than 90° when the drone is at the left side from the center of the corridor and in the same way the angle of intersection is greater than 90° for the case when the drone is at right side from the center of the corridor. These things distinguishes the three different location possible for the drone to be at any point navigation in indoor corridor scenario.

But still the relative location of the drone is not the only component of navigation. There might be a case that drone might be in tilted position. To counter this, our approach will be to bring the drone to the center of the horizontal axis perpendicularly against the trajectory at any point of navigation. But when the drone is at center of this horizontal axis and is tilted then we would like to find the distance of the drone from the bottom left pixel of the image. We would decrease this distance to 0 in order to bring the drone to face straight and forward, by some yaw command.
Some images depicting these cases sustained by the algorithm described so far. Taking into account, when the drone is at the center of the corridor, there exit three cases. Figure 3.2 depicts the scene when the drone is at the center and facing straight. Figure 3.1 and Figure**??** depicts the scene when the drone is at center but tilted, the bisector is making 90° with the horizontal axis.

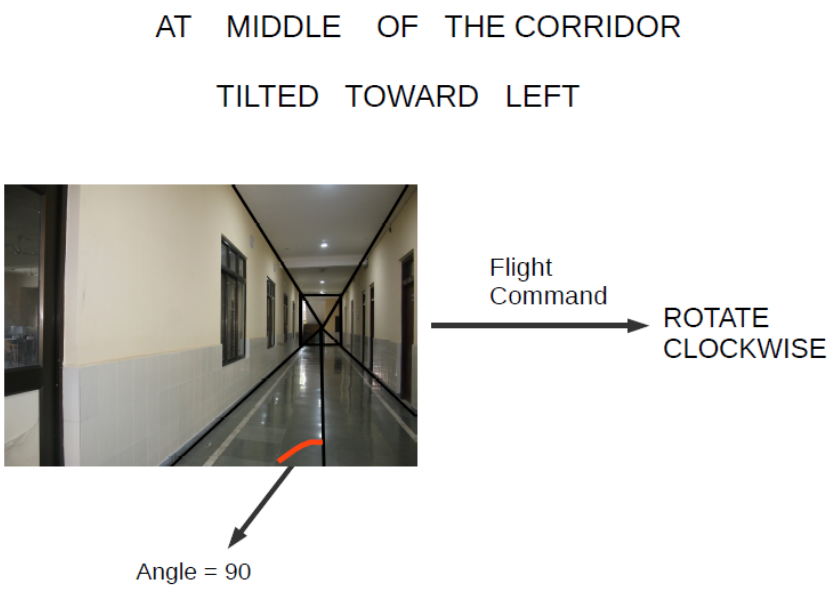


Figure 3.1: Drone at Center tilted Left

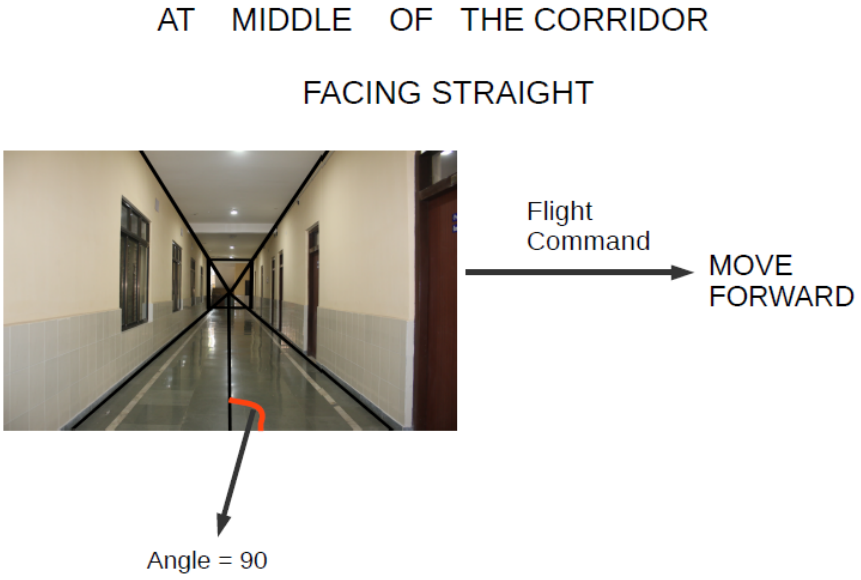


Figure 3.2: Drone at Center facing Straight

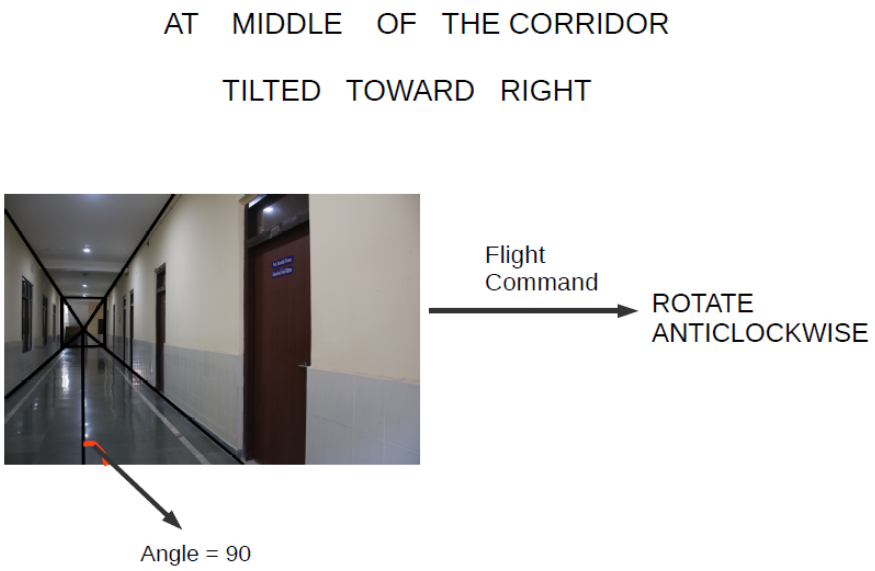


Figure 3.3: Drone at Center tilted Right

We have proposed two algorithms. The Algorithm1 will take care the tilted condition, presently moving over the bisector (described in sec3.3) of the corridor, by calculating the distance of the intersection point of the bisector and horizontal axis in image space from the left most pixel of the image

.

Algorithm 1: DistFind →Distance of point of intersection between the Bisector and Horizontal Axis of the Image

Input: IMAGE of the transit environment coming from Algorithm 2

Result: number of pixels from bottom-left pixel of the image and intersection point

1 Normalize pixel between 0 to 1;
2 RGB to BGR conversion;
3 Normalize with mean and standard deviation of ImageNet-10000 [21] data set;
4 DistPix = Trained_Model(IMAGE);
5 **return** DistPix;

The Algorithm 2 will take the output of Algorithm 1 and predict a flight command for the drone in order to make the flying drone to face straight.

Algorithm 2: Flight_Command →Flight command to make drone to face straight

Input: **IMAGE** taken from the onboard camera

Result: flight command for drone to localize

```
d=DistFind(IMAGE);
if d < image.width/2 then
    yaw right ;
else if d > image.width/2 then
    yaw right;
else
    move forward ;
```

3.4 Proposed Architecture

We have used a quite a few number of pre-trained deep learning model presented in ILSVRC since 2012 and their variants. Originally these pre-trained model were used in majority for classification task. We used transfer learning technique where the original module weight were used and again fine-tuned by further training. Also classification layer of all the original models are replaced with convolutional and fc layer.

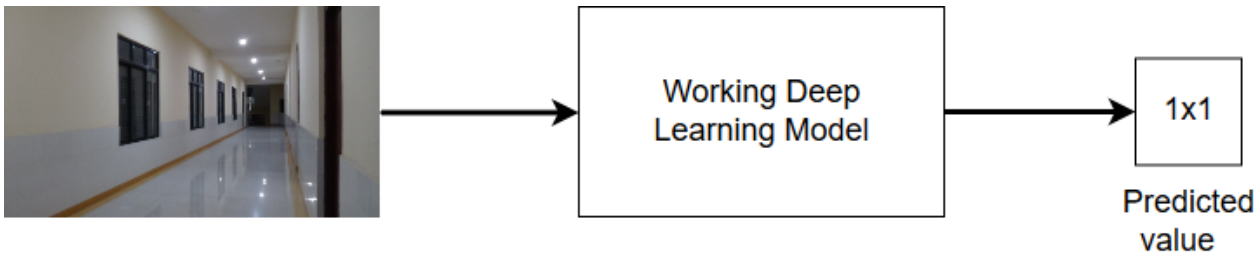


Figure 3.4: fig explains the role of working model

Deep Model Architecture Used			
Pre-Trained Model	**Augmented Layers**	**Output Layer**	**Output**
ALEXNET	-	FC(4096,1)	(1,1)
DenseNet-161	CONV2D(2208,1024,1) CONV2D(1024,128,5) CONV2D(128,16,1)	FC(96,1)	(1,1)
DenseNet-201	CONV2D (1920,1024,1) CONV2D(1024,128,5) CONV2D(128,16,1)	FC (96,1)	(1,1)
INCEPTION-V3	**Main**:CONV2D (1920,1024,1) CONV2D(1024,128,5) CONV2D(128,16,1) **AUX**:CONV2D (768,128,4) CONV2D(128,32,2) CONV2D(128,16,(1,2))	FC (256,1) FC (640,1)	(1,1) (1,1)
ResNet-50	CONV2D(2048,1024,1) CONV2D(1024,128,5) CONV2D(128,8,1)	FC(96,1)	(1,1)
ResNet-101	CONV2D(2048,1024,1) CONV2D(1024,128,5) CONV2D(128,8,1)	FC(96,1)	(1,1)
ResNet-152	CONV2D(2048,1024,1) CONV2D(1024,128,5) CONV2D(128,8,1)	FC(96,1)	(1,1)

Table 3.1: Architecture of Different Models used for Pose Estimation

3.5 System Setup

To carry out this experiment, several hardware and software dependencies are needed to be met. This section will summarize the hardware used in the experiment followed by our custom dataset. And at last we describe the training process.

3.5.1 Hardware Platform

In our experiment we used Parrot AR drone 2.0 [2] attached with monocular camera facing forward, some ultrasound sensors to keep track of ground altitude and an onboard computer. The frontal camera 720 capturing wide range of 92 degree. This camera is HD 720p camera

and has a resolution of 1280 $\times$ 720 pixels. The frames generated at 30 frames per second. The quadcopter is attached with 3-axis gyroscope which measures yaw,pitch and roll anglea and 3-axis accelorometer to measures acceleration in all three directions. These frames generated from the drone are then sent to the host machine, which process these frames, over Wifi, which in turn send the motion command to the drone.
Our host machine has the following specifications

SYSTEM SPECIFICATION	
ITEM	Specification
Memory	15.6 GiB
Processor	Intel Xeon(R) CPU E5-2620 v3 @ 2.40$GHz \times 12$
Graphics	Quadro M2000/PCIe/SSE2
OS	ubuntu LTS 14.04
OS TYPE	64-bit

Table 3.2: Hardware Used to Perform Experiment

3.5.2 Dataset

There were many image dataset are available featuring indoor surrounding but the problem associated with them is that these public dataset lack the ground truth values. There is no benchmark dataset for this experiment.
Also labeling data manually require huge effort and are prone errors. Hence we created our own dataset capturing around three different location in a corridor featured as left, right, center and at each location three possible scenario i,e tilted to left, tilted to right and facing straight.The dataset is created in different building, brightness etc. to cover almost all possible scenarios. Also the height of the corridor is kept fixed at 1 meter above the ground, which is sufficient to travel any flat corridor. Shown below some captured images and labeled images of our custom dataset.

Raw Data

(a) (b) (c)

Figure 3.5: images captured from the onboard camera of drone when it is at the center of the corridor. The first, second and third columns show the images, when the UAV is left, center and right tilted, respectively.

Labelled Data

(a) (b) (c)

Figure 3.6: labelled images presented in fig.3.5 The first, second and third columns show the images, when the UAV is left, center and right tilted, respectively. The red line is drawn in order to find the ground truth.

We used data augmentation technique to increase the size of the data set. The data augmentation was first done in AlexNet [1]. Our augmentation technique basically includes flipping in image 3.7 and zooming in image 3.8.

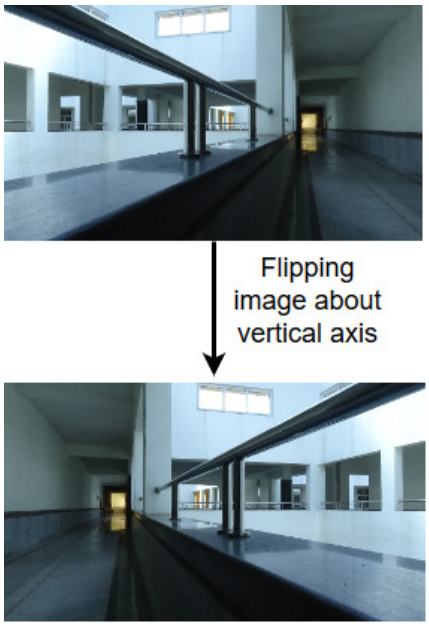


Figure 3.7: image captured from onboard camera is flipped

Figure 3.8: image captured from onboard camera is zoomed variably

3.6 Performance Evaluation

We used several deep models to estimate the distance of the point of intersection between the bisector and the horizontal axis. To train these models we used 21000 images for training and 300 images for testing. Few samples of these images are shown in section 3.5.2. Following two subsection summarize the loss function and the result of training and testing process. The distance here is termed in number of pixels of the image.

3.6.1 Train Set Result

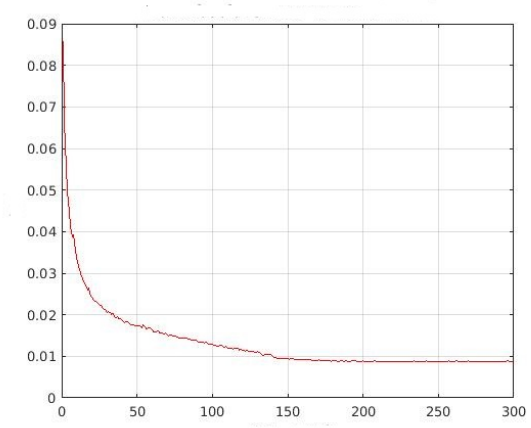
For distance estimation we used only one training loss function i.e. Mean Absolute Error (MAE).

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

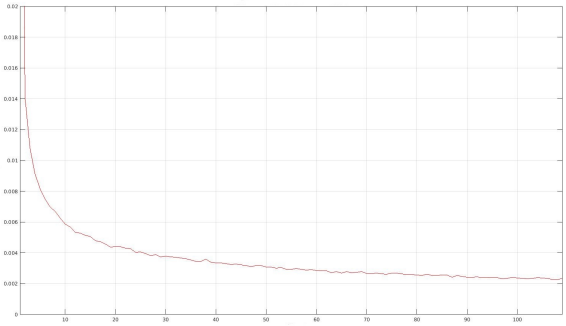
Here,
$\hat{y}$ is the prediction made on all the training examples
y is ground truth value of all the examples of training set.
$\hat{y}_t$ is the prediction made on t^{th} training example
y_t is ground truth value of t^{th} training set.
n is the total number of training example.

MAE is a measure of difference between two variables. Assume $\hat{y}_t$ and **y** are variables that is observations that express the same attribute. Examples of $\hat{y}_t$ versus **y** include comparisons of predicted versus ground truth. Mean Absolute Error (MAE) is the average vertical distance between each point of the varible in comparision.

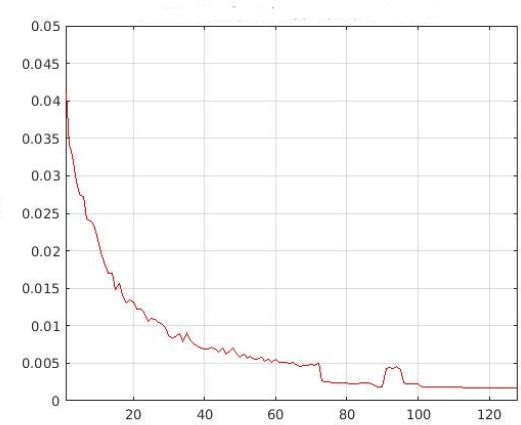
Convergence graph for different model trained with our custom dataset in our experiment is shown in figure 3.9 below, where the x-axis describes the no of iteration while y-axis describes the training loss.



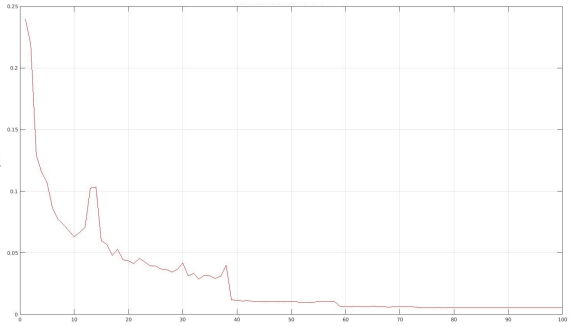
(a)



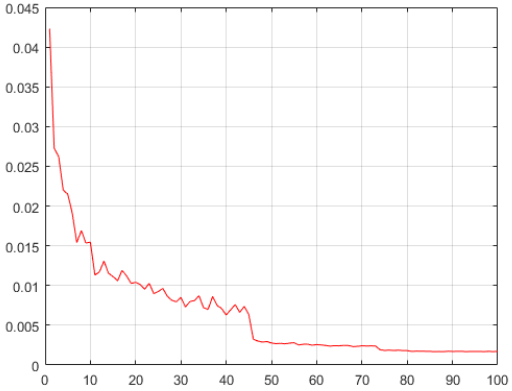
(b)



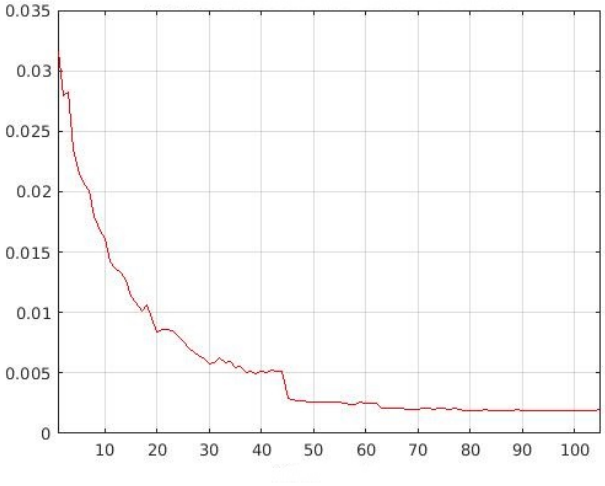
(c)



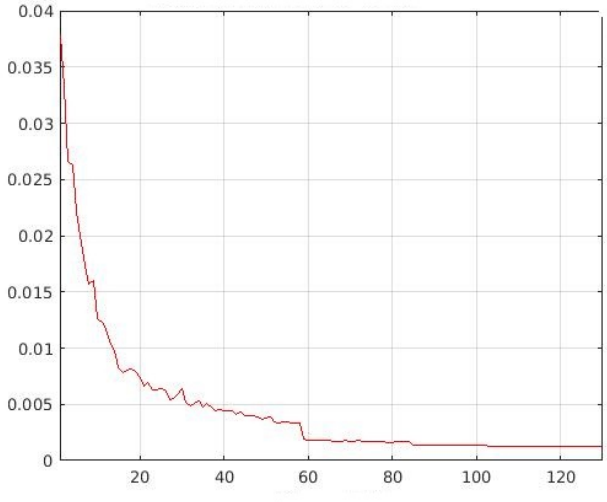
(d)



(e)



(f)



(g)

Figure 3.9: figures depicts the convergence graph for the different model used in the experiment. From top to bottom the sequence of the model is as follows (a)AlexNet, (b)DenseNet-161, (c) DenseNet-201, (d) Inception-V3, (e)ResNet-101, (f) ResNet-101, (g) ResNet-201

3.6.2 Test Set Result

We used three metrics to measure the performance of our proposed model over test set for angle prediction.

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

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

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

Where,

$\hat{y}$ is the prediction made on all the training examples
y is ground truth value of all the examples of training set.
$\hat{y}_t$ is the prediction made on t^{th} training example
y_t is ground truth value of t^{th} training set.
n is the total number of training example.
The absolute error is the measure of the difference between the exact value and the approximation. The relative error is the absolute error divided by the magnitude of the exact value. The mean square error is the square of the difference between exact value approximated over entire example.

Result for Distance Estimation by Different Model Used(pixels)			
Model Name	**Mean Square Error**	**Mean Absolute Error**	**Mean Relative Error**
AlexNet	5.5677	27.0064	54.1467
DenseNet-161	0.0326	2.5060	1.557
DenseNet-201	0.0828	3.6442	12.1421
Inception-V3	0.0687	3.1364	10.4852
ResNet-50	0.0473	2.7485	9.0729
ResNet-101	0.1186	4.2163	14.6806
ResNet-201	0.06617	3.4258	10.8230

Table 3.3: Error in Test Set Distance Prediction

From the table we can see that, DenseNet-161 is giving the best result for all the three metrics. Hence, we consider DenseNet-161 as our final model. Some test result has been shown in the table3.6.2.

Distance Estimation by DenseNet-161 on Test Images(pixels)				
Images	**Ground Truth**	**Predicted Distance**	**Difference**	**Location**
	**53.6**	**52.172**	**1.42**	CS-Dept
	**44**	**48.13079834**	**4.13079834**	TIIR
	**174.93**	**169.0135**	**5.91**	Main Building
	**156.2**	**157.84**	**1.581**	TIIR
	**257.6**	**258.7**	**1.1725**	Physics Dept
	**262.3999939**	**266.3128357**	**3.912841797**	TIIR

Table 3.4: table presenting some of the result of test set by our trained DenseNet-161

Our model is working fine. The worst case prediction differs about 9 pixels from the

ground truth . We have considered around ± 20 pixel for the distance prediction to be work effectively. The best case prediction by our model differs around 0 pixel from the ground truth.

Chapter 4

Discussion and Conclusion

This thesis focus on drone localization for effective drone navigation across any flat corridor. This research used design of a fully convolutional deep architecture to extract the deep intrinsic feature of the environment of transit. This architecture facilitates end-to-end training. We examine and propose localization method from different discipline, namely by distance between the intersection of the bisector of the corridor and the horizontal axis from leftmost point of the 2D image.

Enabling autonomous localization with the single monocular camera frames requires a smart combination of methods and hardwares from various fields, and knowledge about their specific strengths and weaknesses. We managed to apply these methods in our own ways, creating a fully functional and modular system.

We presented the result produced from different model used for training. The model used in our thesis are all winner of ILSVRC challenge of their times and these model originally performed classification task. But our case is regression task, the final output of the model is not class label rather value. Also our dataset is larger in size and is totally different than Imagenet [21] dataset.

The non-reliance of our approach over dedicated onboard hardwares is kept null apart by using only a single monocular camera. The high burden task of estimation is done by the host machines, receive the frames from the drone via wlan connection between the device and host machine and after the estimating the distance will send flight command to the device. We devised our own custom dataset as the already available dataset available has no ground truth as a pixel distance , means our approach is totally novel and global to be used with any other robot device for flat corridor. The approach we followed can be easily applicable for any robot moving in any flat corridor like environment. The approach proposed in here is applied to flying device like drone but can be used for vehicular robots also.

Future Work

The research findings out of this thesis have shown multiple promising directions for further investigations to improve localization accuracy as well as to accelerate the overall process.

We can formalize future work as follows:

- Our approach is a localization approach which always tries to bring th e device to the center while transit. When the device is at the center of the corridor the most exact flight command would be to proceed ahead. But the case of when to stop is not dealt in this thesis, but can be achieved using some prior related techniques
- Our approach is has considered the flight environment to be flat always with different dimension, but another case is that , the flight environment may not be always be flat, it might also be curvy.
- More extreme condition and more building scenarios could be augmented to our dataset to enhance diversity of our dataset which in turn could help in more fine learning of deep models.

References

[1] Krizhevsky, A., Sutskever, I., and Hinton, G. E., 2012. "Imagenet classification with deep convolutional neural networks". In Advances in neural information processing systems, pp. 1097–1105.

[2] Parrot AR Drone 2.0. `https://www.parrot.com/global/drones/parrot-ardrone-20-elite-edition`. Accessed: 2018-03-15.

[3] Mellinger, D., and Kumar, V., 2011. "Minimum snap trajectory generation and control for quadrotors". In Robotics and Automation (ICRA), 2011 IEEE International Conference on, IEEE, pp. 2520–2525.

[4] Mellinger, D., Michael, N., and Kumar, V., 2012. "Trajectory generation and control for precise aggressive maneuvers with quadrotors". *The International Journal of Robotics Research,* **31**(5), pp. 664–674.

[5] Müller, M., Lupashin, S., and D'Andrea, R., 2011. "Quadrocopter ball juggling". In Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, IEEE, pp. 5113–5120.

[6] Checchin, P., Gérossier, F., Blanc, C., Chapuis, R., and Trassoudaine, L., 2010. "Radar scan matching slam using the fourier-mellin transform". In Field and Service Robotics, Springer, pp. 151–161.

[7] Engel, J., Schöps, T., and Cremers, D., 2014. "Lsd-slam: Large-scale direct monocular slam". In European Conference on Computer Vision, Springer, pp. 834–849.

[8] Mei, C., Sibley, G., Cummins, M., Newman, P., and Reid, I., 2011. "Rslam: A system for large-scale mapping in constant-time using stereo". *International journal of computer vision,* **94**(2), pp. 198–214.

[9] Huang, A. S., Bachrach, A., Henry, P., Krainin, M., Maturana, D., Fox, D., and Roy, N., 2017. "Visual odometry and mapping for autonomous flight using an rgb-d camera". In *Robotics Research*. pp. 235–252.

[10] Bry, A., Bachrach, A., and Roy, N., 2012. "State estimation for aggressive flight in gps-denied environments using onboard sensing". In Robotics and Automation (ICRA), 2012 IEEE International Conference on, IEEE, pp. 1–8.

[11] Roberts, J. F., Stirling, T., Zufferey, J.-C., and Floreano, D., 2007. "Quadrotor using minimal sensing for autonomous indoor flight". In European Micro Air Vehicle Conference and Flight Competition (EMAV2007), no. LIS-CONF-2007-006.

[12] Bachrach, A., He, R., and Roy, N., 2009. "Autonomous flight in unknown indoor environments". *International Journal of Micro Air Vehicles,* **1[?]**(4), pp. 217–228.

[13] Çelik, K., and Somani, A. K., 2013. "Monocular vision slam for indoor aerial vehicles". *Journal of electrical and computer engineering,* **2013**, pp. 4–1573.

[14] Achtelik, M., Achtelik, M., Weiss, S., and Siegwart, R., 2011. "Onboard imu and monocular vision based control for mavs in unknown in-and outdoor environments". In Robotics and automation (ICRA), 2011 IEEE international conference on, IEEE, pp. 3056–3063.

[15] Blösch, M., Weiss, S., Scaramuzza, D., and Siegwart, R., 2010. "Vision based mav navigation in unknown and unstructured environments". In Robotics and automation (ICRA), 2010 IEEE international conference on, IEEE, pp. 21–28.

[16] Abbott, E., and Powell, D., 1999. "Land-vehicle navigation using gps". *Proceedings of the IEEE,* **87**(1), pp. 145–162.

[17] Padhy, R. P., Xia, F., Choudhury, S. K., Sa, P. K., and Bakshi, S., 2018. "Monocular vision aided autonomous uav navigation in indoor corridor environments". *IEEE Transactions on Sustainable Computing*.

[18] Pomerleau, D. A., 1989. "Alvinn: An autonomous land vehicle in a neural network". In Advances in neural information processing systems, pp. 305–313.

[19] Schalkoff, R. J., 1992. *Pattern recognition*. Wiley Online Library.

[20] Kim, D. K., and Chen, T., 2015. "Deep neural network for real-time autonomous indoor navigation". *arXiv preprint arXiv:1511.04668*.

[21] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al., 2015. "Imagenet large scale visual recognition challenge". *International Journal of Computer Vision,* **115**(3), pp. 211–252.

[22] Ruck, D. W., Rogers, S. K., Kabrisky, M., Oxley, M. E., and Suter, B. W., 1990. "The multilayer perceptron as an approximation to a bayes optimal discriminant function". *IEEE Transactions on Neural Networks,* **1**(4), pp. 296–298.

[23] Wan, L., Zeiler, M., Zhang, S., Le Cun, Y., and Fergus, R., 2013. "Regularization of neural networks using dropconnect". In International Conference on Machine Learning, pp. 1058–1066.

[24] Simonyan, K., and Zisserman, A., 2014. "Very deep convolutional networks for large-scale image recognition". *arXiv preprint arXiv:1409.1556*.

[25] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., et al., 2015. "Going deeper with convolutions". Cvpr.

[26] He, K., Zhang, X., Ren, S., and Sun, J., 2016. "Deep residual learning for image recognition". In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778.

[27] Huang, G., Liu, Z., Weinberger, K. Q., and van der Maaten, L., 2017. "Densely connected convolutional networks". In Proceedings of the IEEE conference on computer vision and pattern recognition, Vol. 1, p. 3.

0