Deep Learning Based Place Recognition for Challenging Environments

by

Devinder Kumar

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2016

Statement of Contribution

There is one conference paper (under preparation) that make up the majority of this thesis. The parts of the thesis that have been co-authored are listed below. Furthermore, the contribution from each author is also presented.

The majority of Chapter 1 , 3 and 4 are from *Deep Learning Based Place Recognition for Challenging Environments* that is under preparation to be submitted at a later conference. The paper is co-authored by me with Arun Das, who helped plan the initial experiments, Helmut Neher who helped in dataset creation for efficient experiments and in calculating the heading errors and lastly Prof David Clausi and Prof. Steven Waslander helped in planning the experiments and guiding the structure of the research.

Abstract

Visual based place recognition involves recognising familiar locations despite changes in environment or view-point of the camera(s) at the locations. There are existing methods that deal with these seasonal changes or view-point changes separately, but few methods exist that deal with these kind of changes simultaneously. Such robust place recognition systems are essential to long term localization and autonomy. Such systems should be able to deal both with conditional and viewpoint changes simultaneously. In recent times Convolutional Neural Networks (CNNs) have shown to outperform other state-of-the art method in task related to classification and recognition including place recognition. In this thesis, we present a deep learning based planar omni-directional place recognition approach that can deal with conditional and viewpoint variations together. The proposed method is able to deal with large viewpoint changes, where current methods fail. We evaluate the proposed method on two real world datasets dealing with four different seasons through out the year along with illumination changes and changes occurred in the environment across a period of 1 year respectively. We provide both quantitative (recall at 100% precision) and qualitative (confusion matrices) comparison of the basic pipeline for place recognition for the omni-directional approach with single-view and side-view camera approaches. The proposed approach is also shown to work very well across different seasons. The results prove the efficacy of the proposed method over the single-view and side-view cameras in dealing with conditional and large viewpoint changes in different conditions including illumination, weather, structural changes etc.

Acknowledgements

There are a number of people who I would like to thank that made my M.A.Sc. possible. All the following people helped me get to where I am today.

First off, I would like to thank both Dr. Steven Waslander and Prof. David Clausi for co-supervising me. I am grateful to both of you for giving the opportunity to study at the University of Waterloo at the WAVE and VIP Lab. I had a very positive learning experience during my time as a masters candidate.

I also want to thank Dr. Alexander Wong for sharing his knowledge with me and helping me with coming up ideas. I also greatly appreciate you making me work through some conceptual concepts regarding deep learning.

Finally, I want to thank three important persons in my life: my parents and my brother. Thank you mom and dad for always letting me choose my path and for guiding me whenever I diverted from it. Thank you Ravinder Kumar for your support and moral guidance during my masters. I greatly appreciate your encouragement.

Dedication

This is dedicated to my parents.

Thank you mom and dad for always being there.

Table of Contents

List of Tables

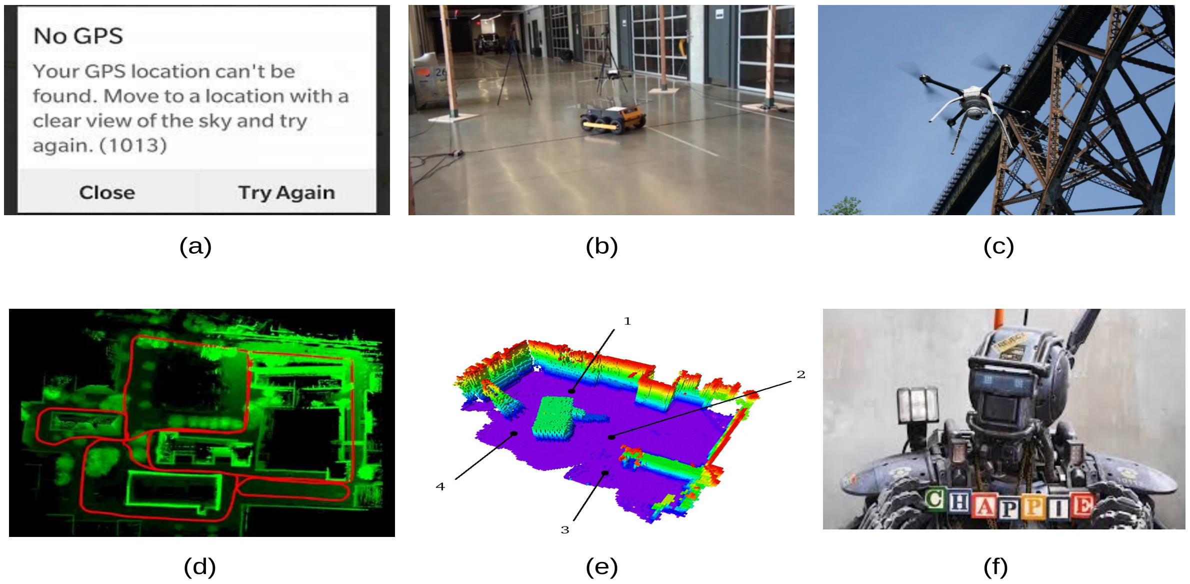
List of Figures

Chapter 1

Introduction

Visual place recognition is one of the most important parts of visual robot navigation, and can be defined as the task of identifying the same locations visited at different periods of time. It has gained lot of traction in recent times [29] due to its many advantages over other sensor modalities such as low cost, low power consumption, easy setup and ability to work even in areas with no GPS signal. Also, the potential applications of place recognition in the field of loop closures, localization, etc., makes it a vital task under the banner of Simultaneous Localization and Mapping (SLAM) [54] that needs to be solved. Place recognition, in particular, provides a reliable method for elimination of accumulated errors in SLAM, as revisited locations enable loop-closure in constructed maps, or geo-referenced pose estimates when performed relative to previously collected image sets in a fixed map. Other applications of place recognition include augmented reality, where the user obtains information about important places, monuments or texts from a single image taken with a smartphone camera and service robotics in the industrial space.

As the robots are tested over longer time periods in real world environments, it is becoming clear that perceptual change, caused by factors such as day-night cycles, varying weather conditions and seasonal change, remains a significant challenge for vision based place recognition methods. Though many advance have been made in the recent past [33, 56, 22], improving place recognition accuracy and reliability, however, still remain open problems. There are particular issues in the domain of changing environments (such as illumination changes, across seasons, structural changes in the environment, etc.) and viewpoint changes that affect place recognition accuracy, which need to be addressed to achieve long term autonomy. Therefore, there is a need to have robust place recognition systems that can deal with these changes, i.e, are conditional and viewpoint invariant. The biggest advantage of a such system would be for long term autonomy across months or


No GPS

Your GPS location can't be found. Move to a location with a clear view of the sky and try again. (1013)

Close Try Again

Figure 1.1: Demonstration of various application areas of visual place recognition such as (a) no GPS service availability, (b) indoors robotics, (c) near or under the bridge conditions for UAVs, (d) loop closures, (e) visual SLAM and (f) long term autonomy .

years, as there will not be a need to update the map with multiple copies of the same location under different conditions.

In recent times, many methods have been proposed to deal with conditional variances [41, 31, 22] including cross season changes [22, 50], illumination changes [30, 44] and viewpoint changes [8, 51] individually, but there are only a few methods [43, 32, 31] that deal with these invariances simultaneously. Also, the existing methods are either single-view [22, 51, 53] or side-view camera [43, 42] based approaches which makes them ineffective in dealing with extreme viewpoints changes that arise in unrestricted operations such as in parking lots or off-road.

Many of the above mentioned methods are based on deep learning or more specifically convolutional neural network (CNN) based methods. This comes as no surprise as CNNs have been gaining importance in most of the recognition and classification tasks [23]. Even when used as a generic feature extractor, CNNs outperform the state-of-the art methods for task other than classification [48]. The features extracted from CNNs have been demonstrated to be versatile and transferable that is, even though they have be trained to solve a particular task, that can be used to solve different problems often outperforming traditional methods. The success of CNN or deep learning is attributed to the reason that much of the design of these networks is inspired by the human visual processing pathway specifically to incorporate both the selectivity and tolerance or invariance qualities of the pathway. Another factor that is attributed to the success of CNNs is their ability to learn millions of parameters using large amounts of labeled data. This makes CNNs a very important tool in the domain of place recognition and broadly in the domain of computer vision and machine learning which is being realized by many researchers in the field. This is the reason behind the recent explosion in the field of computer vision field for methods related to deep learning including the field of place recognition, hence the many recent methods exploring the feasibility of CNNs in this area.

In this thesis, we present an omni-directional deep learning based method for cross season and appearance change invariant visual place recognition. Using a five camera system, we obtain a 360° degree planar omni-directional view of the environment (Fig. 1.2) and use deep features extracted from a pre-trained convolutional neural network (CNN) to perform visual place recognition, as shown in Fig. 1.4. To prove the efficacy of the presented approach over the conventional single and side-view camera approaches, we conduct the following experiments:

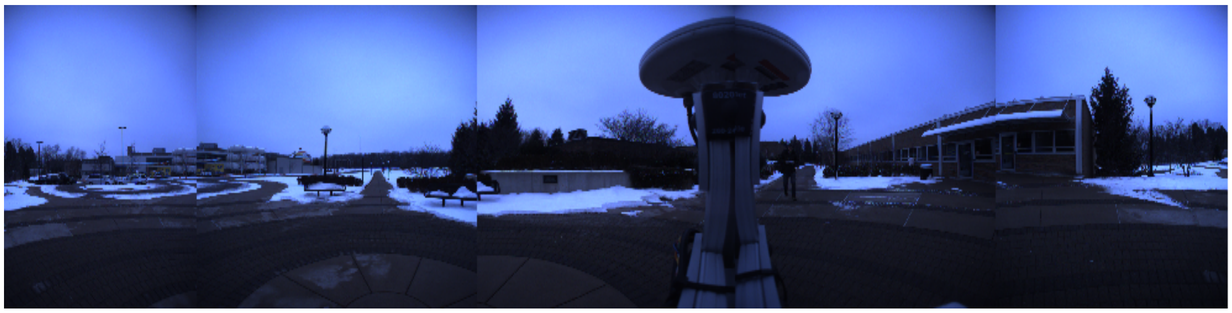
- Challenging changing environment place recognition on outdoors for four different time of the year (cross season: spring, summer, fall and winter) and snow laden environment vs snow laden environment with a gap of 1 year (same season, appearance changes). The dataset was collected using an omni-directional camera on a segway robot at the University of Michigan campus, as shown in Fig. 1.3

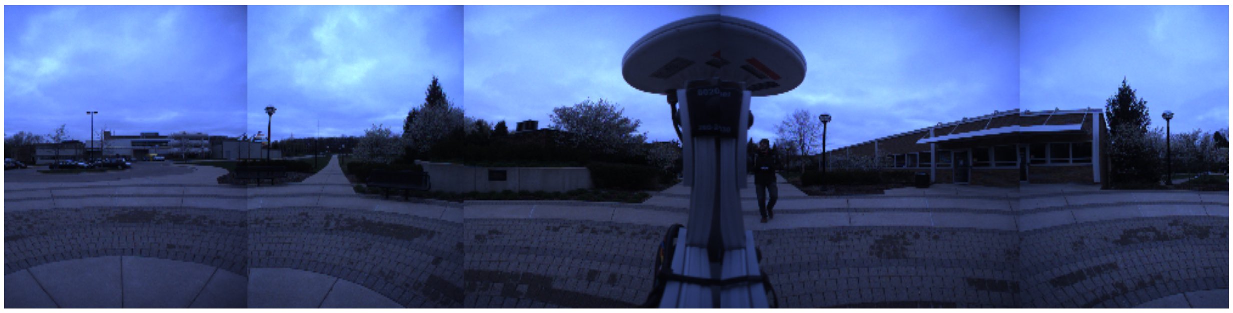
- Extreme viewpoint place recognition on the above datasets (cross season for summer and winter and same season) by querying the same place with five different orientations.

  It is important to point out here that we use the NCLT dataset [3] to perform training and testing for different environmental conditions but the presented method can be implemented on any dataset with omni-directional images.

1.1 Thesis Contributions

The presented place recognition system is a deep learning based method which uses an omni-directional approach for four different seasons that improves the fundamental visual


(a) Winter 2012


(b) Spring 2012

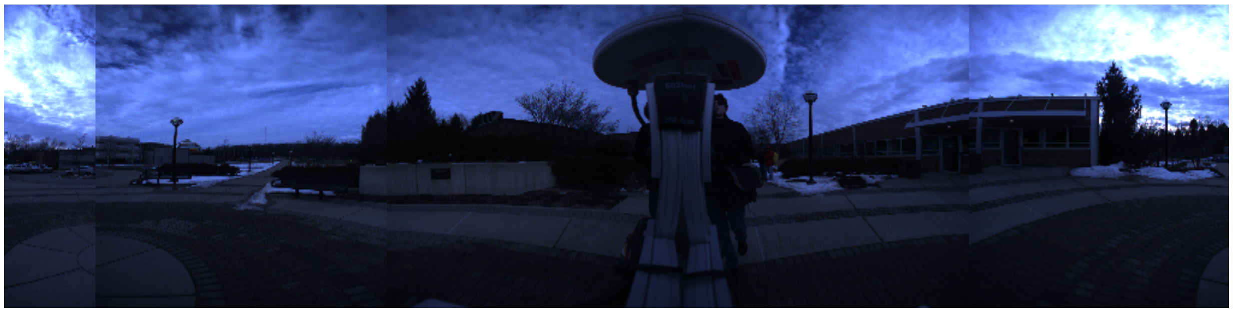

(c) Winter 2013

Figure 1.2: Sample of images identified as the same location by the method presented in this thesis. The images show the diversity of changes compared to each other including illumination, orientation and seasonal changes.

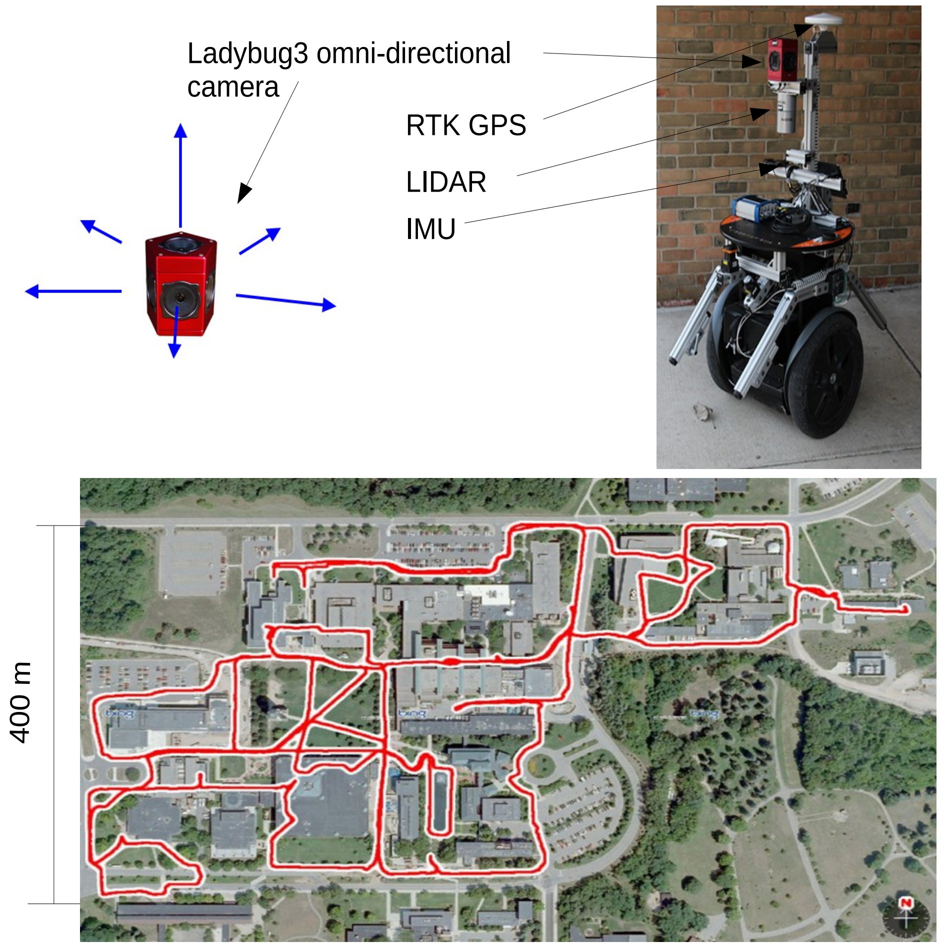


Figure 1.3: Figure showing the Segway robot with different mounted sensor including the Ladybug3 camera system with 6 camera cluster. The figure also shows the path followed by the Segway robot on north campus of the University of Michigan [3] for collecting the dataset. We make use of all cameras except the one facing upwards to create a planar omni-directional view of the scene for our place recognition system.

place recognition pipeline in comparison to conventional single and side-view camera approaches. It is important to note here that to do unbiased comparison with the single and side-view camera studies in the past, we do not perform any pre-processing or post-processing on the input and output data. It is certainly possible to improve the results further by performing post processing checks like sequence matching e.g. SeqSLAM [33], spatial continuity check [6] or using the lighting invariant transform [56] but we focus on performance of single point comparison with the single and side-view camera approaches in this work. To prove the efficacy of the presented approach over the recent approaches, we provide both quantitative (recall at 100% precision) and qualitative (confusion matrices) comparison to prior work.

The main advantages of the presented method are: 1) using omni-directional imaging instead of single and side-view cameras significantly improves the results, 2) the omni-directional approach can deal with extreme viewpoint changes arising due to changes in orientation.

1.2 Literature Review

In the past, numerous place recognition methods have been proposed for large scale environments, including FAB MAP [7, 8], typically used for appearance based loop closures, MonoSLAM [10] and SeqSLAM [33], which are used to handle considerably large degrees of perceptual change but there are very few long term place recognition systems [34, 50, 28]. Even these long term place recognition systems only account for little to no viewpoint variations [50, 32, 22]. This is one of the issues that is addressed in this thesis.

An ideal place recognition system that can be used for long term localization should be both condition and viewpoint invariant. In the recent past, place recognition systems have been proposed to deal with conditional changes such as illumination changes [30, 44], cross season & weather changes [22, 50] and day-night changes [33]. Most of these place recognition systems use training to dynamically model or predict changes in appearance [38, 32] or create large databases which account for all of these changes. This places limitations on these methods in terms of collecting large training sets and limited capabilities in terms of unseen changes. Also, many of the existing approaches lack robustness to significant viewpoint changes. In [22], the authors present a discriminative scene descriptor for single-view cross-season place recognition using naive Bayes nearest neighbor approaches. Milford et. al. [31] use a top-down single camera patch matching approach to deal with conditional changes. In [43, 41] Pepperell et. al. deal with viewpoint variations by using side-view camera approach with their SMART algorithm.

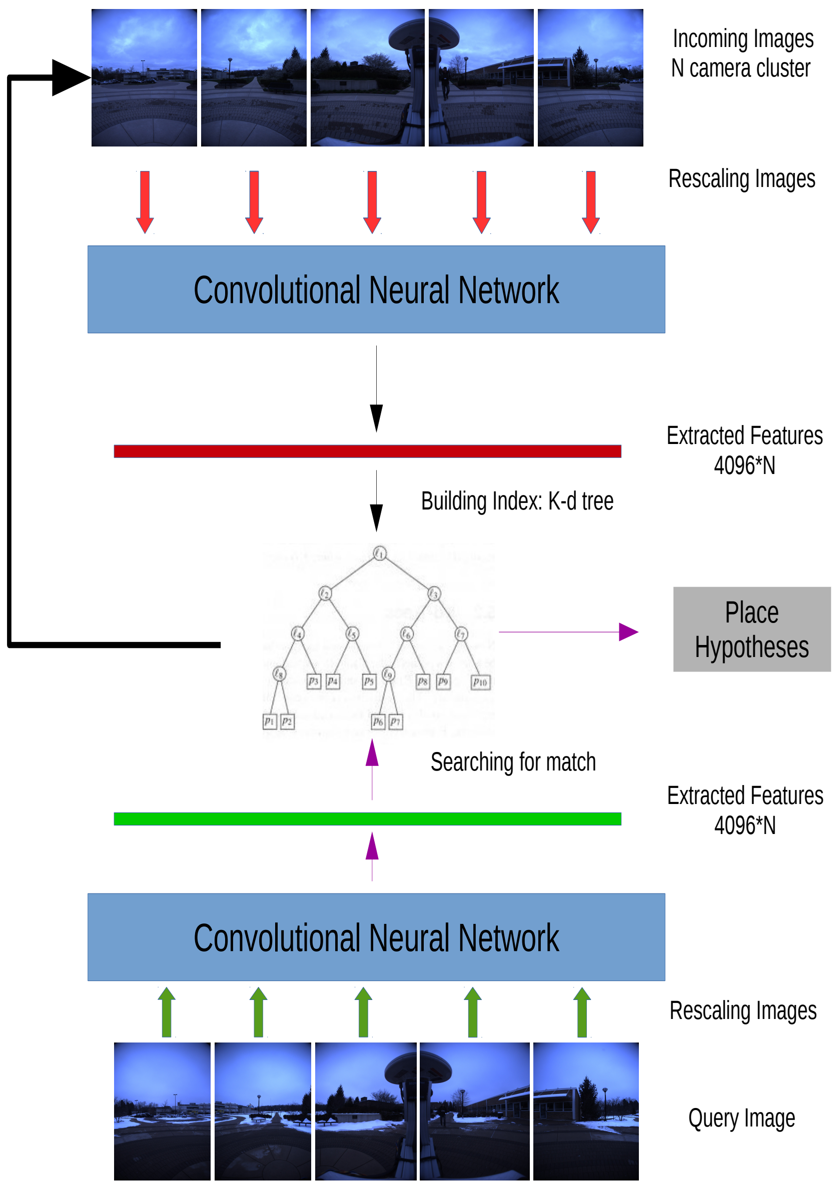


Figure 1.4: Overview of the planar omni-directional architecture for N camera cluster systems.

To deal with viewpoint changes, many keypoint based methods [51, 22] using SIFT, SURF and other keypoints have been proposed. These methods work very well for viewpoint changes under similar conditions. But a major shortcoming of such methods is when the conditions change for example such as day to night, or summer to winter. In some of these conditions, the images exhibit little texture with little to no keypoints detection [9]. Thus there are lot of these methods that use point based features as SIFT, SURF etc., in less challenging environmental conditions. Recently, ORB-SLAM [37] was proposed that uses ORB features [47] to perform place recognition using key-point features with considerable accuracy.

Deep learning based methods have been shown to be effective in dealing with conditional and viewpoint changes [32, 52, 15, 34] as they are able to extract generic descriptors from images that are invariant to seasonal or slight viewpoint changes [6]. One of the important recent study in this area is Milford et al. [32] where a CNN is used to create generic daytime single camera depth images that match with night time depth images with viewpoint changes hence dealing with condition and viewpoint changes simultaneously. Sunderhauf et al. [53] used edge boxes object proposal method combined with a mid-level convolutional neural network(CNN). The other methods (non deep-learning based) that do so were presented by Pepperell et. al. in [43, 41]. The authors extended the method by incorporating *path memory* for selecting past experiences candidate matches to improve the computational time [28]. A similar approach of learning and calibrating pre-location califiers for place recognition was proposed by gronat et al. [16]. But in all these studies mentioned above the maximum viewpoint changes are changes in resolution due to traveling on opposites side of roads [43, 41]. There are also other methods that use data from additional sensors such as lasers[40] or RGB-D cameras [13, 59]. These techniques use novel sensors that provide dense depth information as well as image data image data that has accelerated the development of dense mapping techniques [25, 59, 14].

Place recognition system efficacy can be increased by using hierarchical searching at the place level as well as at the vocabulary level. Mohan et. al. [34], proposed a coocurrent feature matrices based method to pre-select the most likely. Preselecting global environment can help in considerably reducing the search space, thus increasing the overall efficiency. Earlier methods used Hamming embedding [21] to perform more precise viewpoint matching, multiple assignment to reduce optimization error [5] and adaptive soft-assignment procedure for repeated structures in database [55].

In this thesis, we deal simultaneously with conditional changes and large viewpoint variants (as shown in Fig. 3.1) using a fast nearest neighbor search without the need of repeated data collection under different conditions. One important thing to note is that, many methods mentioned above use additional metrics such as temporal information to boost the results. As we would like to compare basic pipeline of place recognition for the presented approach mainly with single and side-view camera approaches, we refrain from using such additional information.

1.3 Thesis Outline

The rest of the thesis proceeds as follows: Chapter II describes the related background theory on deep learning in particular convolutional neural network (CNN) that is required to understand many concepts and algorithms presented in this thesis. The chapter lays down, in detail, the process of creating the CNNs and also formulates them mathematically. It also includes a description of the state-of-the art nearest neighbor search algorithm used in this thesis. Chapter III puts forward in detail the presented omni-directional approach. The presented method uses an omni-directional camera to capture planar omni-directional view of a scene and uses that to perform conditional and view-point invariant place recognition using CNN. Evaluation methodology and results are presented in Chapter IV along with some discussion on the obtained results. The evaluation methodology involves both quantitative and qualitative metrics to establish the efficacy of the presented omni-directional approach. The evaluation is done on a comprehensive dataset containing scenes from four different scenes. Finally we conclude with Chapter V where a summary of the work done is this thesis is presented. The chapter also includes a brief outline of the future work directions.

Chapter 2

Background

This chapter presents the background information required for the contributions made by this work. First general motivation for using deep learning is explained, followed by detailed explanation of convolutional neural network (CNN) design including pre-training. We end with an explain of Fast localization using Nearest Neighbor Approximation (FLANN) method used in this thesis.

2.1 Deep learning

In the past few recent years, there has been a resurgence of neural networks with multiple layers which were discovered first in the late 80's and early 90's. These multi-layered neural network come under the topic, known as *deep learning*, which is based on set of algorithms and mathematical models that try to learn high level abstractions in data using non-linear transforms with multiple layers processing.

Recently numerous studies that use deep learning have been presented with promising results in various field such as robotics [49, 45, 63, 27, 12], speech recognition [19], medical imaging [1, 24], scene understanding [63, 31, 58] etc. Apart from the fact that design of deep learning architectures in these studies were motivated from advancement of neuroscience and how information is interpreted and processed by the nervous system of humans, most of the success in the studies have been attributed to advancement of processing power (GPUs) and availability of large datasets of learning.

Various deep learning architectures have been proposed using both, supervised and unsupervised approaches for learning abstract features from given data. Deep learning

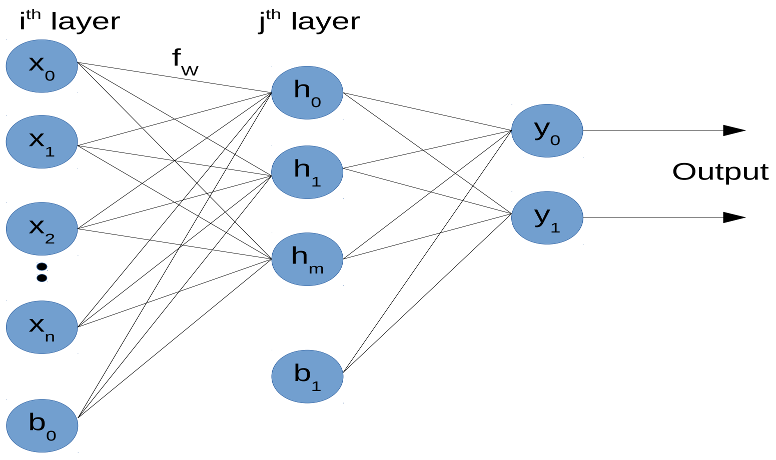


Figure 2.1: Illustration of a simple feed forward neural network with layer of input, one hidden layer with nodes represented as h_j and an output layer. f_w is the activation function.

architectures such as convolutional neural networks (CNN), deep belief networks, autoencoders, recurrent neural networks (RNNs) and Restricted Boltzman machine (RBM) have been proposed. Even though there are many different architectures, they have the same underlying pipeline, as shown in Fig. 2.1. Recent studies [26, 17, 4, 18, 60] have shown that CNNs are the best architectures to perform recognition and classification tasks. As this thesis uses a CNN for place recognition, the design architecture for CNNs have been explained in the following section.

2.2 Convolutional Neural Networks

Convolutional neural networks (CNNs or ConvNets) are feed-forward neural networks that are biologically inspired variants of the multi-layer perceptrons (MLPs) with learnable weights and biases. The connections between different layers of CNNs is based on the connections in visual cortex of animals (including humans), which is arranged in a manner that the neurons in the cortex respond to overlapping regions in the visual field. These overlapping sub-regions are called receptive fields. The sub regions are titled to cover the whole visual field. CNNs learn these sub-regions filters i.e., weights of the filters over the input space. CNNs make an explicit assumption that the input is images, which allows the encoding of certain specific properties to its architecture. Some of the properties are explained in the section below. For a better understanding of CNNs, brief overview of the feed forward neural networks is presented in this section followed by detailed description of CNNs including the design and training aspect.

2.2.1 Feed-forward neural networks

This section gives a description of the feed-forward neural networks, which are the most basic neural networks (NNs). A more detailed description of the feed-forward networks can be found in [20]. Usually feed-forward NNs consists of single input layer, one or more hidden layer followed by an output layer as shown in Figure 2.1. Consider an input consisting of labeled data $\{x^{(i)}, y^{(i)}\}$. Here $x^{(i)}$ and $y^{(i)}$ are the i^{th} input case (feature) and class label respectively. In the next step, using a complex non-linear function $f_{W(x)}$ usually known as activation function the neural network transforms the input feature vector $x_1^{(i)}, x_2^{(i)}, .., x_n^{(i)}$ into a higher dimensions complex representation of the input data. The function $f_{W(x)}$ is parameterized by weight matrix W that can be learned. Using the learned weights the feed-forward NNs propagate an input through the different layers in the network to the output to make a decision. As shown in the Figure 2.1, the activation of the j^{th} hidden layer's i^{th} unit can be represented as,

$$z_i^j = \sum_{k=0}^{n} W_{ik}^j x_{ik} + b_j \tag{2.1}$$

$$a_j = f(z_i^j) \tag{2.2}$$

Here b_j is the bias term. The activation function f in most of the cases is a non-linear function such as sigmoid $\frac{1}{1+\exp^{-x}}$ or tanh. Activation of each unit of hidden layers can be calculated using the above mentioned equations. Since the activation unit value of a unit in layer relies on the activation units of the preceding layer, we initiate from the first hidden layer and proceed through the network layer-wise. Propagating information through a neural network in this manner is referred to as *forward-propagation*. Using the forward-propagation step, at the end of the network, we obtain $y_1, y_2, .., y_k$ labels. These labels are then used to perform classification.

During training, the objective is to learn the parameters of the weight matrix W. Generally the parameters are learned by minimizing some kind of *objective* or *loss function*. The objective function can be as simple as least squares error or a much more complex one, such as cross entropy [46]. One of the popular methods to minimize the objective function is *back-propagation*. The readers are encouraged to go through [2] for a detailed description of gradient descent.

2.2.2 CNN architecture overview

Using the information provided above, in this section we will now describe the different components involved in a convolutional neural network's architecture starting briefly with convolution and later describing different layers such as non-linearity layer, fully connected layer and different functions such as pooling or sub-sampling etc.

2.2.3 Convolution

For the sake of completeness, we briefly describe convolution in this part. For simplicity, consider an gray-scale image I as:

$$I = \{1, 2, ..., p_1\} \times \{1, 2, ..., p_2\} \to W \subseteq \Re(i, j) \longmapsto I(i, j) \tag{2.3}$$

such that the image I can be represented by an array of size $p_1 \times p_2$. Given the filter $F \in R^{(2h_1+1 \times 2h_2+1)}$, the discrete convolution of the image I with filter F is given by :

$$(I * F)_{a,b} = \sum_{c=-h_1}^{h_1} \sum_{c=-h_2}^{h_2} F_{c,d} I_{a+c,b+d} \tag{2.4}$$

where the filter F is a typical filter image of odd dimensions.

One thing to note while using convolution is that the behavior of the convolution operation should be carefully defined at the border of the images. There are many useful convolutional filters that are constantly applied in many applications. One such filter is Gaussian filter which is used to provide smoothing effect.

2.2.4 Layers

Using the information above, we now introduce the different layers and operation between the layers that are used in a typical CNN. Any complex CNN such as ones used in [62] can be constituted by stacking the below layers together.

Convolutional Layer

Consider a convolutional layer l which uses an input that comprises of $n_1^{(l-1)}$ feature maps from the previous layer, each of size $n_2^{(l-1)} \times n_3^{(l-1)}$. If $l = 1$ i.e., it is the first layer of the

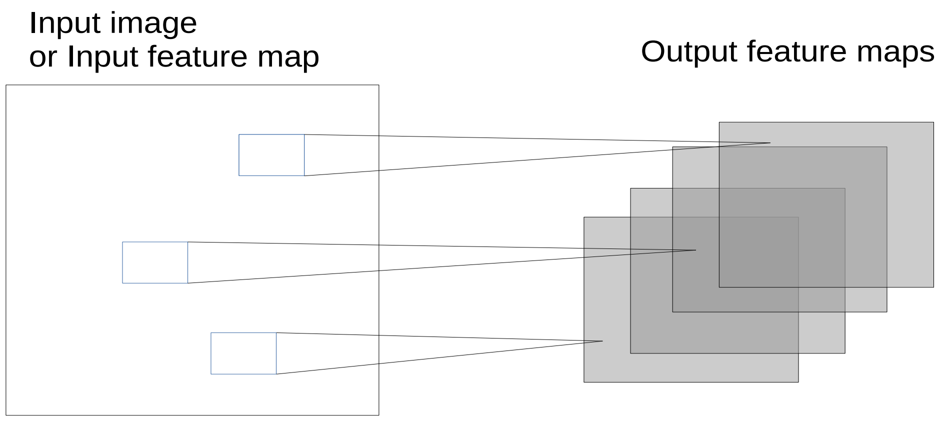


Figure 2.2: Illustration of a single convolutional layer. If layer l is a convolutional layer, the input image (if l = 1) or a feature map of the previous layer is convolved by different filters to yield the output feature maps of layer l.

convolutional neural network then the input is the raw image I consisting of one or more channels. The output of the layer l consists of $n_2^{(l)} \times n_3^{(l)}$. The m^{th} feature map in layer l which is denoted as $fm_m^{(l)}$, is computed as:

$$fm_m^{(l)} = b_m^{(l)} + \sum_{q=1}^{n_1^{(l-1)}} F_{m,q}^{(l)} * fm_q^{(l)} \tag{2.5}$$

here $b_m^{(l)}$ is the bias matrix and $F_{m,q}^{(l)}$ is the convolutional filter of size $2h_1 + 1 \times 2h_2 + 1$ which joins the q^{th} feature map in the $(l-1)$ layer with the m^{th} feature map in layer l. As described before, $n_2^{(l-1)} \times n_3^{(l-1)}$ are effected by the border regions. When implementing the discrete convolution in the valid region i.e., the region of pixels where the sum of equation 2.4 is defined properly, we obtain the feature maps of sizes:

$$n_2^{(l)} = n_2^{(l-1)} - 2h_1^{(l)} \tag{2.6}$$

$$n_3^{(l)} = n_3^{(l-1)} - 2h_2^{(l)} \tag{2.7}$$

A visual example of convolution is shown in Fig. 2.2. Usually the fixed feature map $fm_m^{(l)}$ is constructed using the same filters, that is $F_{m,q}^{(l)} = F_{m,r}^{(l)}$ for $q \neq r$. Also, the sum in equation 2.5 may be implemented on part of the input feature maps. For better understanding of the convolutional layer and it's working as stated using equation 2.5, we can rewrite the equation 2.5 in terms of multi-layered perceptron. The layer l with feature maps $fm_m^{(l)}$, contains $n_2^{(l)} \times n_3^{(l)}$ units arranged in two dimensional arrays. The output at any position (a, b) for a unit is given by:

$$(fm_m^{(l)})_{a,b} = (b_m^{(l)})_{a,b} + \sum_{q=1}^{n_1^{(l-1)}} (F_{m,q}^{(l)} * fm_q^{(l)})_{a,b} = (b_m^{(l)})_{a,b} + \sum_{q=1}^{n_1^{(l-1)}} \sum_{c=-h_1^{(l)}}^{h_1^{(l}} \sum_{d=-h_2^{(l)}}^{h_2^{(l)}} (F_{m,q}^{(l)})_{c,d} (fm_q^{(l-1)})_{a+c,b+d} \tag{2.8}$$

In the equation above, filter $F_{m,q}^{(l)}$ and bias $b_m^{(l)}$ contains the trainable weights. As explained in the later sections, sub-sampling or pooling can be used to minimize the effect of noise and distortion. The idea is to skip a fixed number of pixels, both in horizontal and vertical directions, before applying the filter again.

Non-linearity layer

If a layer is a non-linearity layer and the input to the layer is given by $n_2^{(l)}$ feature maps then the output is given as $n_1^{(l)} = n_1^{(l-1)}$. Each map is of size $n_2^{(l-1)} \times n_3^{(l-1)}$ such that $n_2^{(l)} = n_2^{(l-1)}$ and $n_3^{(l)} = n_3^{(l-1)}$ are given by:

$$fm_m^{(l)} = \varphi(fm_m^{(l-1)}) \tag{2.9}$$

where φ is a non linear function, also known as activation function, used for layer l and is implemented point-wise. In many cases, a non-linearity layer is included in a convolutional layer and not represented individually. For the sake of clarification, it is mentioned separately in this thesis.

Rectification

Let a layer, l, be a rectification layer with an input consisting of $n_1^{(l-1)}$ feature maps of size $n_2^{(l-1)} \times n_3^{(l-1)}$. The value of each component of the feature maps is defined by:

$$fm_m^{(l)} = |fm_m^{(l)}| \tag{2.10}$$

The above equation is sometimes also represented as:

$$fm_m^{(l)} = \max(0, fm_m^{(l)}) \tag{2.11}$$

For both equations, the values are calculated point wise such that the output feature map size is unchanged i.e., equal to input. Many past experiments [62, 61, 19] have shown that rectification helps in achieving better performance in the CNN. This layer can also be incorporated into a convolutional layer, but we follow the convention of representing this operation as an independent layer. The rectification layer can sometimes be also represented as Rectified Linear Unit (ReLU).

Local Contrast Normalization Layer

Local contrast normalization encourages local competitiveness among different adjacent units in feature map and units at the same locations in different feature maps. For explanation, consider a layer l be a contrast normalization layer. Given $n_1^{(l-1)}$ feature maps of size $n_2^{(l-1)} \times n_3^{(l-1)}$, the output layer then is obtained as $n_1^{(l)} = n_1^{(l-1)}$ feature maps, same as input. One type of local contrast normalization (subtractive) can be formulated as :

$$fm_m^{(l)} = fm_m^{(l-1)} + \sum_{q=1}^{n_1^{(l-1)}} G_{\mu,\sigma} * fm_q^{(l-1)} \tag{2.12}$$

Here $G_{\mu,\sigma}$ is the Gaussian filter with mean, μ, and standard deviation of σ.

Feature Pooling

Feature pooling or sub-sampling's main motivation is to impart robustness to the feature maps from noise and distortions. Pooling is obtained generally by reducing the resolution of the input image or feature map which can be performed in different ways. For better explanation, consider a pooling layer l. Its output comprises $n_1^{(l)} = n_1^{(l-1)}$ feature maps of smaller sizes than the input. In general, pooling is performed by dividing the input image or feature map into non-overlapping windows and then sub-sampling from each window separately. This can be done in the following two ways:

- Average pooling : This is performed by taking the average of the values present in a particular window.

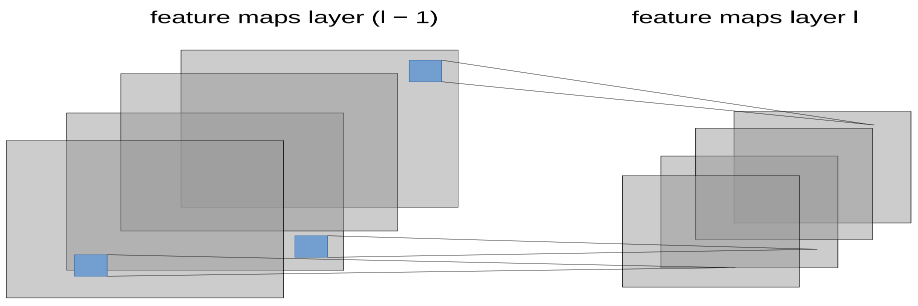


Figure 2.3: Illustration of a pooling and sub-sampling layer. If layer l is a pooling and sub-sampling layer and given $n_1 = 4$ feature maps of the previous layer, all feature maps are pooled and sub-sampled individually. Each unit (l) in one of the $n_1 = 4$ output feature maps represents the average or the maximum within a fixed window of the corresponding feature map in layer (l1).

- Max pooling: This is performed by selecting the maximum value from a particular window and using that to create the reduced feature map.

A visual illustration of pooling is shown in Fig. 2.3. As discussed in [**?**], max pooling performs better for the tasks of recognition and classification. It also helps to get faster convergence during training. Pooling also helps in reducing the problem of over-fitting on training sets [23, **?**].

Fully Connected Layer

Let l be a fully connected layer. The layer l needs an input of $n_1^{(l-1)}$ feature maps of size $n_2^{(l-1)} \times n_3^{(l-1)}$, which then for m^{th} unit computes:

$$fm_m^{(l)} = K(z_m^{(l)}) \qquad where, \tag{2.13}$$

$$z_m^{(l)} = \sum_{q=1}^{n_1^{(l-1)}} \sum_{c=1}^{n_2^{(l-1}} \sum_{d=1}^{n_3^{(l-1)}} w_{m,q,a,b}^{(l)} (fm_q^{(l-1)})_{c,d} (fm_q^{(l-1)})_{a+c,b+d} \tag{2.14}$$

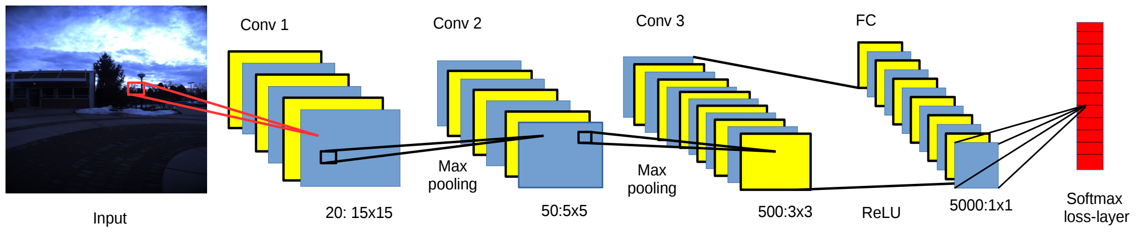


Figure 2.4: The architecture of the typical convolutional neural network, alternates between convolutional layers including hyperbolic tangent non-linearities and sub-sampling layers. In this illustration, the convolutional layers already include non-linearities and, thus, a convolutional layer actually represents two layers. The feature maps of the final sub-sampling layer are then fed into the actual classifier consisting of an arbitrary number of fully connected layers. The output layer usually uses softmax activation functions.

where $w^{(l)}_{m,q,a,b}$ denotes the connecting weights between the unit at position (a, b) in the q^{th} feature map of layer $(l-1)$ and the m^{th} unit of layer l. In practice, the convolutional layers are used to learn a feature hierarchy and one or more fully connected layers are used to condense the learned features which are then used for the purpose of classification. It should be pointed out that the fully connected layers already contain non-linearities while for the convolutional layers the non-linearities are separated in their own layer.

CNN Full Architecture

This section explains the typical architecture of a convolutional neural network that can be constructed using different layers explained in the earlier sections of this chapter. A typical CNN is shown in Fig. 2.4. The CNN shown comprises of three convolutional layers and in between them contains pooling and ReLU layers. The network at the end has a fully connected layer, features from which are then used for classification. The dimensions of each layer are shown in the Fig 2.4. Back-propagation and drop-out is used in the training similar to what is used in feed-forward networks as explained earlier in the chapter.

In this thesis, we use a pre-trained VGG-s CNN of 19 layers which was trained on ImageNet dataset of 1000 classes with 1.2 million example cases. For a more detailed explanation on the VGG-s network, the readers are encouraged to read through [57].

2.3 Fast Library for Approximate Nearest Neighbors: FLANN

One of the prominent bottleneck in terms of implementation speeds for many robotics algorithms is searching for most similar matches to high dimensional feature vectors i.e., nearest neighbor searching. This problem becomes even worse in the cases where the search space is huge, for example in the application area of long term autonomy. To deal with this problem, in this thesis, we make use of an approach based on priority search k-means tree which is popularly known as fast library for approximate nearest neighbors (FLANN) [36]. Even though there is another well known approach known as randomized k-d forest in this framework, but we choose the priority search approach as it produces results with higher precision comparatively. The remainder of this part of the chapter describes the underlying algorithm used in this thesis to determine the nearest neighbor search.

2.3.1 Priority Search K-Means Tree Algorithm

The priority search k-means tree algorithm produces higher precision results by exploiting the natural structure present in the data. The priority search k-means computes full distance across all the dimensions which leads to the better clustering of the data points compared to the other popular k-d tree approach which divides the data into only one dimension at a time. Also, the presented priority k-means tree approach is different than the other k-means approaches [39], as the presented approach uses a better and improved version of the k-means tree. The presented k-means tree approach uses *best-bin-first* strategy, which improves the performance of the approximate k-d tree searches. The *best-bin-first* strategy is explained below.

Best-Bin-First Approach: Algorithm Description

In the initial stage of the implementation of the priority search k-means tree algorithm, the data region is divided into K distinct regions to create the k-means tree by using k-means clustering. This is applied recursively to create distinct regions for each point at different levels. The recursion is halted if the number of obtained points in a region is smaller than K.

After the creation stage, in the next stage the tree is then searched by traversing from root to closest leaf, following at each inner node the branch with the closest cluster centre to the query point, and adding all unexplored branches along the path to a *priority queue*. After the addition, sorting of the priority queue is done.The order is than changed from the point closest in distance to the query up-to the branch being added to the queue. Once an initial traversal is done, the algorithm continues traversing the tree, always from the top branch in the queue.

Choosing the factor K, known as the branch factor, is vital for getting good search performance. The FLANN framework includes an algorithm for finding the optimum branch factor. More details regarding this can be found in [36]. Another important factor in the priority search k-means tree approach is the maximum number of iterations factor. Too few iterations can result in less optimum search performance results though the experiments in [36] prove that even smaller iterations (around 7) can give up to 90% of precision.

The FLANN framework has been tested on many different datasets to prove its effectiveness. Thus considering that we used this framework in this thesis to find the closest matches to the incoming scenes in our datasets. The details regarding the implementation of FLANN in this thesis can be found in the next chapter.

2.4 Summary

This chapter provided the necessary background that is important to understand the working and implementation of convolutional nerual networks and the state-of-the art nearest neighbor search algorithm (FLANN). The above sections formulated CNNs and FLANN mathematically and explained the process of creating and inner working of CNNs and FLANN algorithm. By using the above presented theoretically background, the next chapter lays out in detail, how these methods are implemented to create a robust visual place recognition system.

Chapter 3

Methodology

This section describes the components of the planar omni-directional place recognition system depicted in Fig. 1.4. A planar omni-directional multi-camera system consisting of N cameras is used to capture a 360° degree planar view of the scene. The set of N images is then input into pre-trained convolutional neural network to obtain the feature set as output. We use a CNN trained on imagenet instead of training a separate CNN because as seen in Gomez-Ojeda et.al. [15], even after training a CNN with 2 million images for the specific task of place recognition, it fails to outperform the pre-trained CNN at many datasets. This output is then used to build an kd-tree index from a fast nearest neighbor approximation approach. When a query image set is given to this kd-tree index, it gives the approximate closest matched place using L2 distance. Individual modules of the omni-directional place recognition system are explained below.

3.1 Omni-directional Feature Extraction

To obtain the omni-directional deep features for each scene, we input each image individually into conventional neural network. We use a deep convolutional neural network from the MatconvNet library [57] trained on ImageNet [11] which consist of 1.2 million images from 1000 classes of object. The CNN consists of 20 layers, out of which 5 are convolutional layers, 3 are fully connected layers and the rest are rectification linear unit (ReLu) and pooling layers. When an image I, from ${C_j}^{\text{th}}$ camera, from the N camera cluster for a scene is input into network, it is first down-sampled to 231×231 as required by the CNN network and is then passed through the various layers in the network. The penultimate fully connected layer is used as the feature extraction layer, which produces a 4096 element feature vector f_{ij}. This process is repeated for each image from the N cameras, in a planar omni-directional scene for $j = 1$ to N resulting into a feature set F_i of $4096 * N$ length for the i^{th} scene in the dataset.

$$F_i = [f_{i1}, f_{i2}, .., f_{iN-1}, f_{iN}] \tag{3.1}$$

3.2 Building Search Space for Fast Place Recognition

Using the $4096 * N$ dimensional feature set F_i obtained from the previous Sub-section 3.1 for the i^{th} image in a given dataset, we input this feature set into a kd-tree using Fast Library for Approximate Nearest Neighbors (FLANN) [35] which uses nearest neighbor search to extract the closest distance match from a large database quickly. The kd-tree is built using the features obtained from feature extraction module for all the images in a dataset and by further synthetically creating different orientation variations of the same scene (described in Sub-section 3.3 below). The kd-tree built by this process provides a fast method to query a large database for matching locations, which is essential for long term autonomy.

3.3 Extreme Viewpoint Generation via Orientation Changes

To achieve viewpoint invariance, we synthetically create a range of different orientations of a particular scene. The procedure to create such viewpoint changes is shown in Fig. 3.1. We rotate the feature vector f_{ij} as shown in Eqn. 3.2, in order to automatically create the N different orientations of the same scene which result in extreme viewpoint changes with respect to individual cameras.

$$\begin{aligned} d_{i1} &= [f_{i1}, f_{i2}, f_{i3}, .., f_{iN}] \\ d_{i2} &= [f_{iN}, f_{i1}, f_{i2}, .., f_{iN-1}] \\ &\vdots \\ d_{iN} &= [f_{i2}, f_{i3}, f_{i4}, .., f_{iN}, f_{i1}] \end{aligned} \tag{3.2}$$

$$D = \sum_{j=1}^{N} \sum_{i=1}^{M} d_{ij}^{T} \tag{3.3}$$

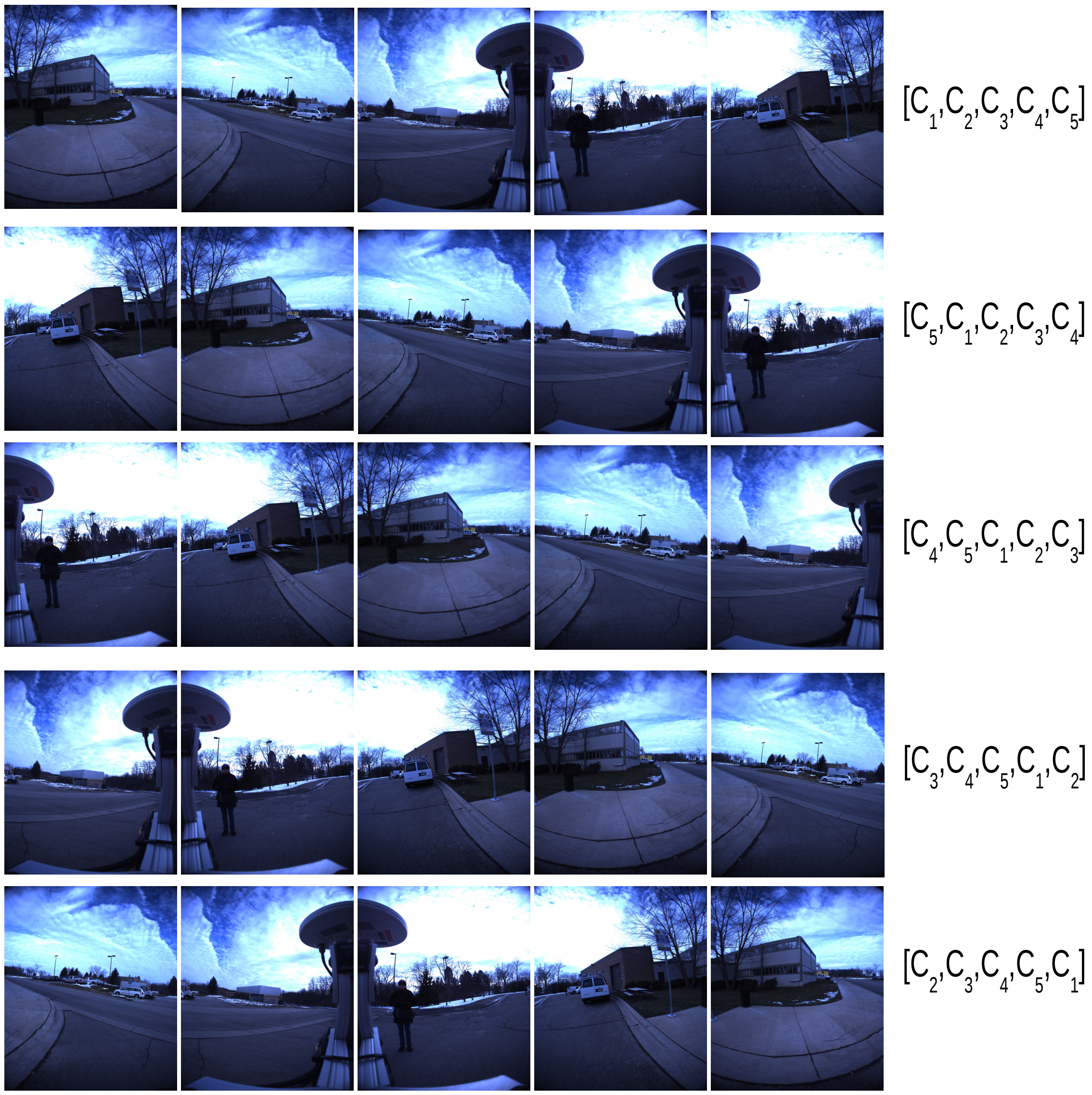


Figure 3.1: Visual representation for synthetic creation of viewpoint changes by changing the camera orientation resulting in N different orientations. C_j is the j^{th} camera image in an N camera cluster of planar omni-directional camera system.

Here d_{ij} is the j^{th} camera orientation for the i^{th} image in a dataset with M frames and D is the collection of all the orientations. All orientations in D are added to the higher dimensional database k-d tree to deal with large viewpoint changes in the future transversal of the same location.

Once all the orientation from every image in the dataset is added to the kd-tree, we stop the training and fix it for the testing. It is possible to expand the kd-tree online, but we do not consider this option in this work. As shown in Fig. 1.4, once we fix the kd-tree, we use the same feature extraction process as described in Sub-section 3.1 for the incoming query image. We extract a feature length of $4096 * N$ for the down-sampled query image and input it into the kd-tree. The kd-tree provides the index to the closest match to the query image from the index database build earlier (Fig. 1.4). As we already know the orientation of all the indexes in the kd-tree, we get both the place hypotheses and orientation of the query image. It vital to point out here that, in the present work we use fixed rotations of $72°$ to create different orientations. A possible extension of this would be consider finer rotations as well, which is currently not addressed in this work.

It is also important to describe here that there were many other extensions that we tried, for example removing sky from the images before extracting features to constraint the seasonal and other environmental vulnerabilities in the images. We implemented this with our system, but the state-of-the art techniques which were to do this also mis-classified the snow on the ground as sky and decreased the over all place recognition accuracy. Hence, we decided against using this in the present system. We also implemented the binarizing feature vectors and reducing number of feature vectors from 4096 to 1000 using PCA, but both of these techniques decreased the accuracy of the system.

3.4 Summary

The above sections in this chapter explained the process of the creating of the presented omni-directional place recognition along with the detailed explanation of the various components involved in the system. The chapter explains the process of extracting features from the images using the pre-trained CNN and building the search space for fast searching. The next chapter uses the presented omni-directional system to conduct various comprehensive experiments for different environmental conditions. The next chapter explains the experimental set-up, dataset used to conduct these experiments and the results thus obtained from the experiments.

Chapter 4

Experimentation

4.1 Experimental Set-up

Experiments were performed on two different datasets that are derived from North Campus Long-Term (NCLT) Vision and Lidar dataset [3]. NCLT dataset was created by multiple runs of a segway robot on the University of Michigan's North Campus over a period of around 1.5 years. The Segway robotics platform was equipped with many sensors including a Ladybug3 omnidirectional camera, a Velodyne HDL-32E 3D lidar, two Hokuyo planar lidars, an inertial measurement unit (IMU), a single-axis fiber optic gyro (FOG), a consumer grade global positioning system (GPS), and a real-time kinematic (RTK) GPS. The Pointgrey Ladybug3 (LB3) uses six 2 megapixel cameras with a resolution of 1600x1200 with five CCD camera in a horizontal ring and the last one positioned vertically. The five horizontal cameras provide an almost 360° planar view. The undistored images of the dataset were collected at 5 Hz and were stored in JPEG compressed format.

To create the databases (cross season, and same season) used in this thesis, we choose the data collected on five different dates: 2012-01-22, 2012-03-31, 2012-06-15, 2012-09-28 and 2013-01-10. Using this data, we form the following databases:

- Cross season : 2012-01-22 (winter), 2012-03-31 (spring), 2012-06-15 (summer) and 2012-09-28 (fall) (Fig. 4.1)
- Same season with gap of 1 year : 2012-01-22 (winter 2012) and 2013-01-10 (winter 2013)

More details about these derived databases are presented in Table 4.1 and Table 4.2.

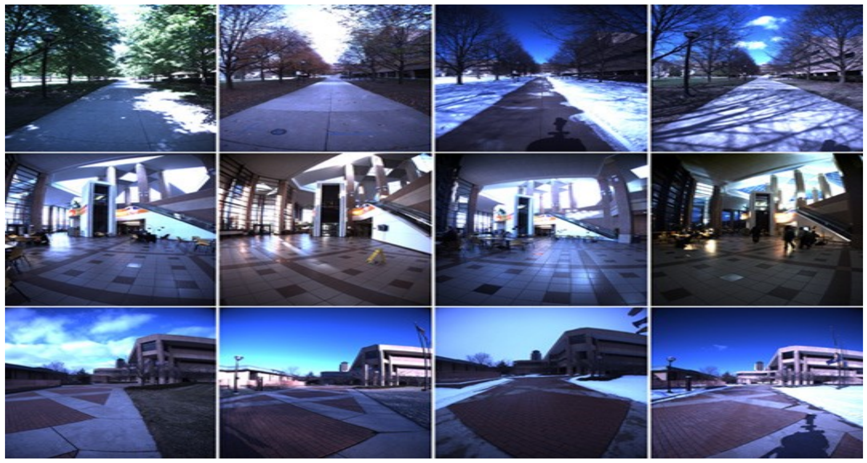


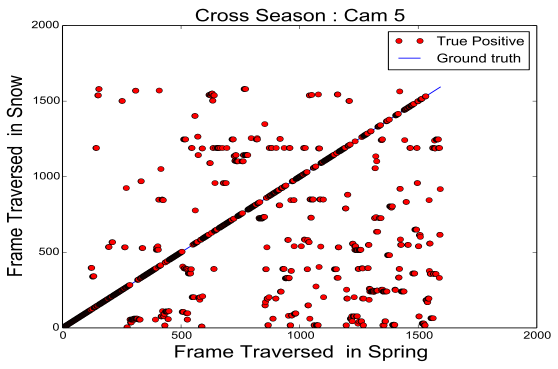
Figure 4.1: Illustrations for three different scenes across four different seasons: spring, summer, fall and winter from the NCLT dataset [3].

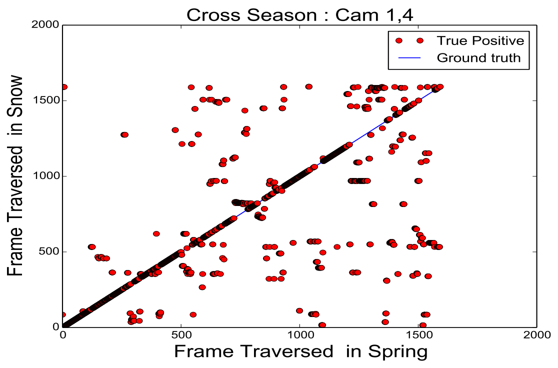
4.2 Ground Truth

Ground truth was determined by matching the GPS data obtained from real-time kinematic (RTK) GPS. We select GPS co-ordinates 1 m apart from one sequence (e.g. winter 2012) and then match with the same GPS co-ordinate in another sequence (e.g. spring 2012). To extract the images associated with the GPS co-ordinate we match UTIME of both the GPS and imaging sensor. The final results were manually matched through visual inspection and the orientation data was verified by the heading provided by the IMU sensor.

4.3 Pre- and Post-Processing

We perform minimal pre-processing on the images. We selected a fixed window for size of 700×1200 pixels in the center of the image, which removes most of the sky and part of the ground, leaving the most relevant information in the image untouched. The fixed window was selected to remove the non-essential information from scenes which might hinder in the place recognition performance. Each image is then scaled and downsampled to 231×231 as required by MatconvNet CNN. We do not perform any post processing on the matching results and consider only L2 distance as a metric for determining accuracy. This approach was chosen to maintain a fair comparison of the presented approach with single and side-view camera approaches.





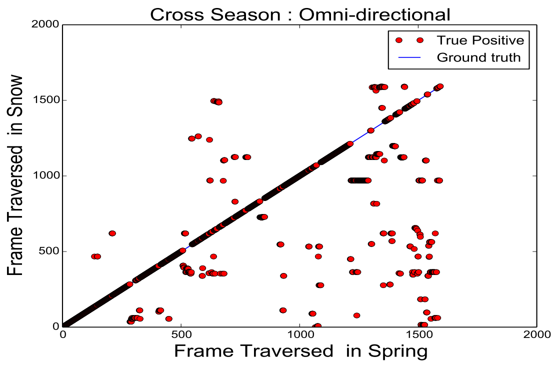


Figure 4.2: Confusion matrices belonging to the best performing individual camera 5, best performing side-view camera 1 & 4 and the presented omni-directional approach for a cross season dataset (winter (snow) vs spring). The confusion matrices are obtained by first traversing in spring to create a database and then traversing the same location in winter with snow (query).

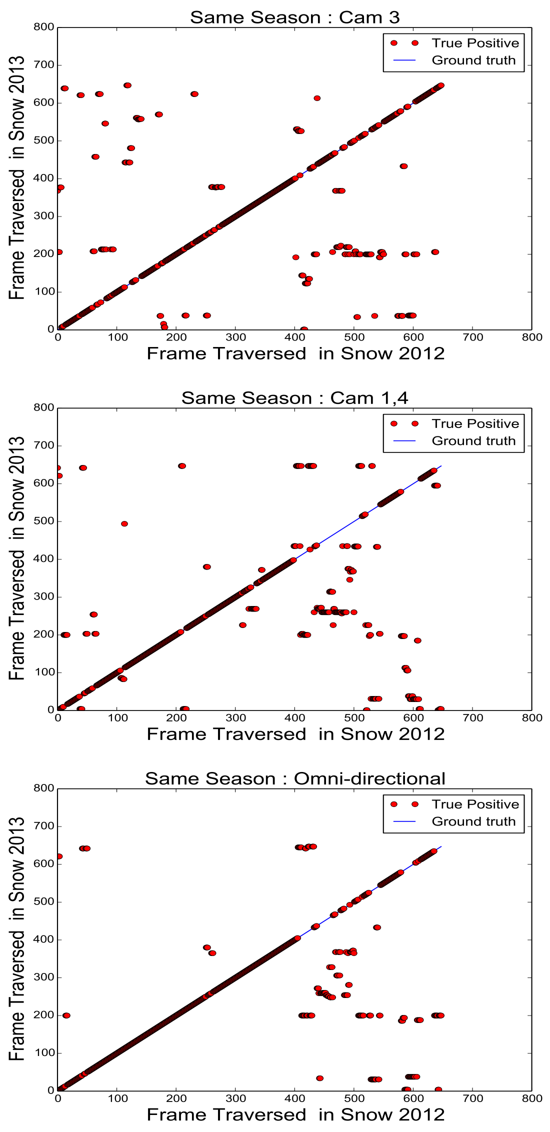
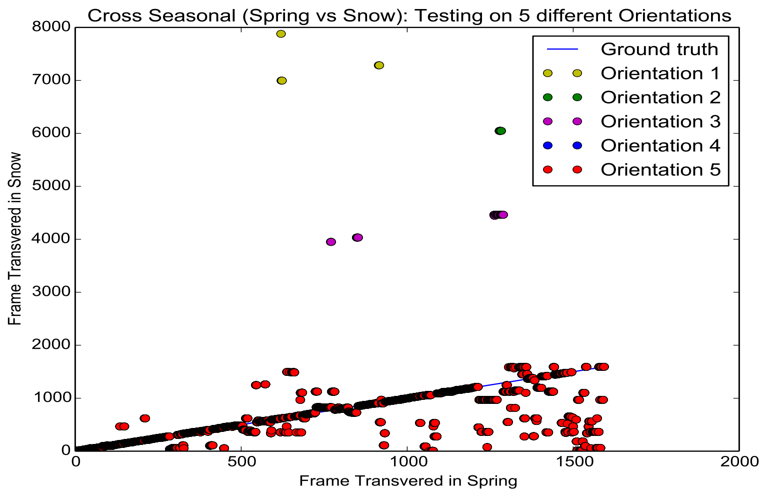


Figure 4.3: Confusion matrices belonging to the best performing individual camera 3, best performing side-view camera 1 & 4 and the presented omni-directional approach for a same season dataset across a gap of one year (snow 2012 vs snow 2013). The confusion matrices are obtained by first traversing in winter 2012 to create a database and then traversing at the same location in winter 2013 with snow (query).



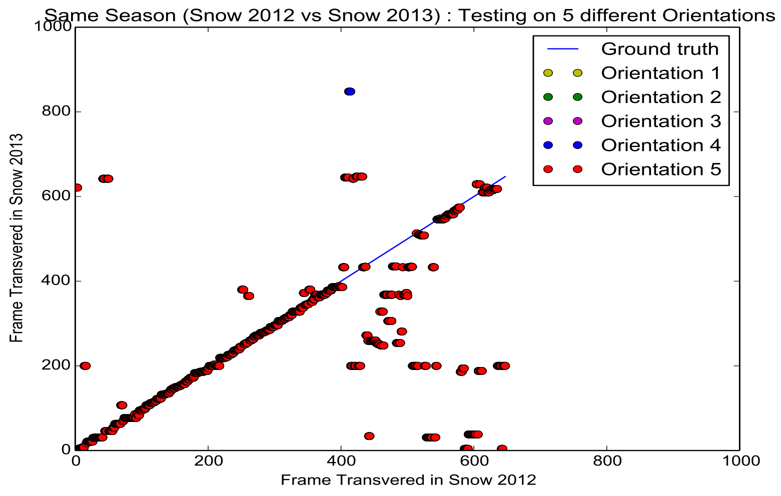


Figure 4.4: Figure showing the confusion matrix results obtained by querying two different datasets with five different orientations at the same location. The left confusion matrix is for cross season database (snow vs spring) and the right confusion matrix is for same season (winter 2012 vs winter 2013). From the figure, it is evident that the presented omnidirectional approach produces similar results even if queried with different orientations thus producing a robust system that can handle extreme viewpoint changes.

Table 4.1: Description of cross season database used in the thesis, which are derived from NCLT [3] database.

Database Name (cross season)	**Total Distance**	**Total Frames**	**Description**
Winter	0.65 km	648	traversal done in Jan 2012 (2012-01-22)
Spring	1.59 km	1594	traversal done in March 2012 (2012-03-31)
Summer	1.59 km	1594	traversal done in June 2012 (2012-06-15)
Fall	1.59 km	1594	traversal done in September 2012 (2012-09-08)

Table 4.2: Description of same season database used in the thesis, which are derived from NCLT [3] database

Database Name	**Total Distance**	**Total Frames**	**Description**
Same Season (Winter'12 vs Winter'13)	0.65 km	648	first traversal in winter 2012 and querying in winters 2013

4.4 Experiments and Results

In this Section, we present three different types of experiments and the results thus showing the place recognition performance of the omni-directional approach in different scenarios. First a comparison to single-view and side-view camera approaches is done with omni-directional approach for small set (winter and summer) derived from cross season dataset and the same season dataset. We also present results of the performance of the presented approach for extreme viewpoint changes created from individual test images using the same season in the same manner as described when constructing viewpoint invariant databases in Section 3.3. We end with evaluating the performance of the presented approach on the full cross season database, presenting place recognition results from all four seasons.

4.4.1 Place Recognition Comparison: Omni-directional vs Single and Side-View Camera Approaches

To prove the efficacy of the presented omni-directional approach over the conventional single and side-view camera approaches, we calculate the place recognition performance of the approaches over the two datasets. In this part, we only use a smaller part of the cross season dataset i.e., winter vs summer only, and full same season dataset as mentioned in Table 4.1 and Table 4.2. Recall performance at 100% precision is the criterion of interest, as false positives are undesirable in visual place recognition. The results of place recognition in terms of recall values at 100% precision for a sequence length of 10 m for each individual cameras in the 5 camera cluster, side-view combinations of these cameras along with the omni-directional approach is shown in Table 4.3. From Table 4.3 it is evident that the omni-directional approach outperforms both the single view and side-view camera approaches for both the datasets, which indicates the efficacy of the presented approach over the single-view and side-view camera methods.

We also present the best raw confusion matrix results in Fig. 4.2 and Fig. 4.3 for cross season and same season datasets for each method respectively. In the figures, we can notice that most of the orientations belonging to same locations are identified as such (overlapping), and only few orientations are identified as different places (green dots in cross season and blue in same season figure). From both figures, it is visually evident that the omni-directional approach generates far fewer false positives as compared to individual and side-view cameras.

4.4.2 Performance Under Extreme Viewpoint Changes

To state the effectiveness of the presented approach under viewpoint changes, we create 5 different orientations of each scene in both the datasets (Table 4.1 and Table 4.2) as explained in Sub-section 3.3. Similar to the above Sub-section 4.4.1, we use part of the cross-season dataset (winter and spring) and full same season dataset. This results in 7970 frames in the mini cross season and 3240 frames in the same season dataset. We then query each orientation separately for both datasets. We plot the confusion matrices thus obtained for both datasets in Fig. 4.4. Inspecting the raw confusion matrices in Fig. 4.4, it is clear that even after changing the orientation significantly, the presented approach creates very few outliers in both cases. Outliers here are defined as the place hypotheses results that are different from the ones that were identified while training and testing on single orientation omni-directional images (Fig. 4.2 & 4.3). This observation is verified by the results shown in Table 4.4. From the confusion matrices (Fig. 4.4) and Table 4.4, we can conclude that the presented method does not increase the number of outliers due to synthetically created orientations.

We also calculate the recall at 100% precision values for these cases, which remain the same for all the orientations. Both of these results demonstrate that the omni-directional approach can withstand extreme viewpoint changes without any significant change in performance. It is also interesting to note here that this same approach can also be used to find the orientation. This can be easily done by labeling each orientation in the dataset differently and then matching the orientation of the query image with the existing orientations in the dataset.

To further analyze the effectiveness of the presented omni-directional approach in dealing with viewpoint changes, we evaluated the error percentage with respect to the difference in heading angle of the training and testing set without the synthetic orientations. The results of this experiment for both dataset are demonstrated in Fig. 4.5. Observing the histogram we can conclude that there is a significant growth in the error rate as a function of orientation change in the same season dataset, but not in the mini cross season dataset. This suggests that the error percentage might be a function of number of samples (cross season: 1594×5 and same season: 648×5). Also, the error percentage is greater for same season, one reason for such performance might be because of the similarity of scenes covered in snow that lead to a harder place recognition problem, and a greater sensitivity to orientation. For the cross season, as the matched images have significant differences, the sensitivity is not as pronounced as in the case of the same season dataset.

4.4.3 Cross Season Place Recognition Performance

To prove that the presented omni-directional approach works for all weather conditions, we evaluate the presented approach over four different weather conditions as mentioned in the cross season dataset (Table 4.1). The experiments were done by training the network on one of the seasons separately and then testing individually on the rest of the three seasons. As the winter part contains smaller number of frames, training and testing for all the seasons is done using only 648 frames in the case where the network is trained on winter season. When the CNN is trained on rest of the seasons, the training and testing is done on full length of 1594 frames, except when testing is done against winter season. In that case, frames are only used up-to the length of 648 i.e., of 0.65 km. The results thus obtained, pertaining to recall at 100% precision are shown in Table 4.5.

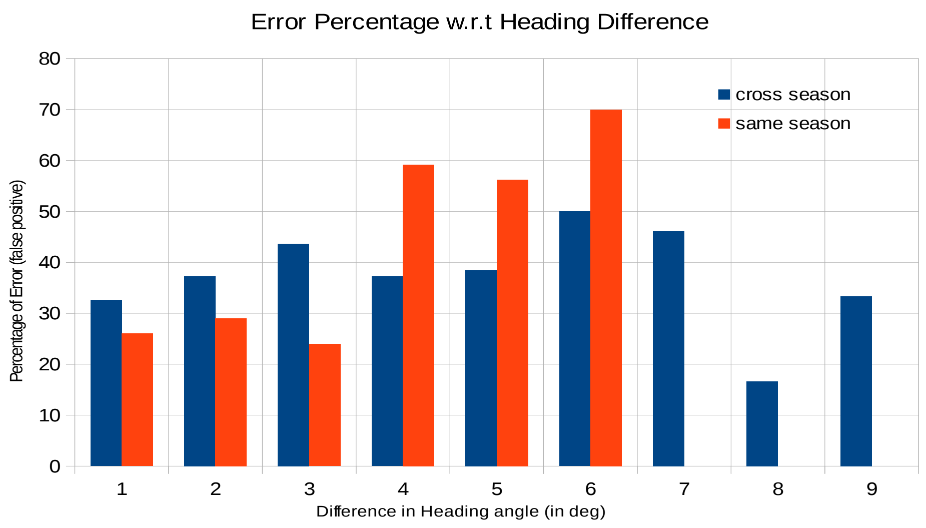


Figure 4.5: Histogram showing the percentage of locations mis-classified with respect to the difference in the heading angle between training set and testing set.

From the Table 4.5, it is clearly evident that even though the presented place recognition system works with decent results, the results are better when there are only gradual changes in the weather. For example, the best results are produced while training and testing for summer and fall. The main reason for such results is the lack of major changes in the appearance of the season as the two sequences were collected close to each other (mid summer and early fall). Similarly, the results for winter and summer are the worst compared to the rest of the set as the difference in scenes due to major changes between sunny and clear conditions in summer and grim with ice covered conditions in winters. Also, it is important to note that as training and testing was done on different seasons albeit from the same dataset. A natural extension to this is to use other datasets with omni-directional images to either train and test in various other conditions, which would be an interesting and vital step. However, this method cannot be extended to datasets with only single-view or side-view images. But as more and more sensors are installed in robots these days we hope the presented results will encourage other researches in the field of place recognition to move towards omni-directional imaging.

Table 4.3: Recall values (%) at 100 % Precision for two different Datasets (Cross Season: Winter'12 vs Spring'13 & Same Season: Winter'12 vs Winter'13). Best results are highlighted in bold.

	Cam No.	**Cross Season (%)**	**Same Season (%)**
Single-View	01	39	47
	02	36	36
	03	37	48
	04	36	44
	05	42	42
	Avg	38	43.4
Side-View	1, 2	39	54
	1, 3	51	61
	1, 4	55	63
	1, 5	50	61
	2, 3	49	53
	2, 4	49	55
	2, 5	52	52
	3, 4	47	57
	3, 5	54	58
	4, 5	51	52
	Avg	49.7	56.6
Omni-dir.	1, 2, 3, 4, 5	**66**	**71**

4.5 Summary

This chapter explains the experimental set-up and the various experiments conducts using a dataset that contains wide variety of conditional and viewpoint changes. The chapter first explains the creation of the dataset including the ground truth used to verify the correctness of the collected data. The chapter then explains the various pre-processing techniques involved in the experiments. After this, various experimental scenarios and obtained results from the respective experiments are explained including a comparison of single and side-view camera approaches. A discussion is also presented for the obtained results. The next chapter summaries the whole work presented in this thesis and also provides a short discussion on the future directions for presented work.

Table 4.4: Number and percentage of outliers introduced for different testset with kd-tree trained on 5 different orientations (d1-d5).

Orientation Testset	**Cross Season**	**Same Season**
$d_1 : 1, 2, 3, 4, 5$	46 (0.029%)	3 (0.004%)
$d_2 : 5, 1, 2, 3, 4$	33 (0.020%)	3 (0.004%)
$d_3 : 4, 5, 1, 2, 3$	27 (0.017%)	3 (0.004%)
$d_4 : 3, 4, 5, 1, 2$	46 (0.029%)	3 (0.004%)
$d_5 : 2, 3, 4, 5, 1$	46 (0.029%)	3 (0.004%)
Avg	**40 (0.025%)**	**3 (0.004%)**

Table 4.5: Recall values (%) at 100 % Precision for four different seasons (spring, summer, fall and winter of 2012 in the cross season dataset. Best results are highlighted in bold.

Training/Testing set	**Spring**	**Summer**	**Fall**	**Winter**
Spring	–	**75**	71	66
Summer	**76**	–	72	63
Fall	71	**73**	–	68
Winter	64	64	**67**	–

Chapter 5

Conclusion

This chapter provides a summary of the work presented in this thesis along with discussion on the avenues of futures directions.

5.1 Summary

This thesis proposes a deep learning based visual place recognition using planar omnidirectional imaging. Unlike many recent deep learning based approaches, which use the single or side-view camera approach for place recognition, the presented approach is robust to viewpoint changes without using any additional information from any other source or device. The presented approach is shown to work well for cross seasonal changes for four different weather conditions namely, winter, spring, summer and fall. Along with this, the presented approach is also shown to work for long term structural changes by using imaging data from the same location after a gap of a year i.e., winter 2012 and winter 2013.

Our presented technique enhances the overall performance of place recognition by 20% & 22% over the single camera technique and 8% & 7% over side-view camera approach for recall at 100% performance on two different real datasets that deal with illumination, seasonal changes and structural changes over 1 year respectively. We also demonstrate that the presented technique has the capability to handle large changes in viewpoint or orientation. We believe both of these improvements are important steps toward creating more robust place recognition systems that can withstand difficult and changing environments.

The presented approach for place recognition is also shown to work across different weather condition throughout the year. The presented approach produces a good 76%

recall at 100% precision for summer and spring testing and does well for the rest of the seasons too.

5.2 Future Work

In future, we want to explore the performance of the omni-directional approach for place recognition in the following different scenarios:

- For unmanned aerial vehicles
- For key-frame selection
- Fine-grained rotations
- Using saliency maps for place recognition

As large viewpoint changes are common in aerial vehicles, it will be interesting to see the results in this domain.While GPS has long been used as a localization sensor on both piloted and unpiloted air vehicles, localization using GPS is not possible in indoor or cluttered outdoor environments where GPS is generally not available. Thus it is important to have a GPS independent system for aerial vehicles. The initial experiments will hopefully provide a direction to improve and implement a working place recognition system for aerial vehicles.

One of the most important area in the visual SLAM field is key-frame selection. Key-frames are the scenes/images in the SLAM database that are used as anchor points in the world to accurately localize the autonomous vehicles. In future, we want to explore the possibilities of using omni-directional visual place recognition system to select robust key-frames that increase the effectiveness of the visual SLAM algorithms.

We also want to improve the viewpoint invariance of the presented approach by incorporating fine grained rotations of the scenes with stitched and masked images for larger training sets. The inclusion of fine grained rotations in the dataset will help in providing a more robust place recognition that can handle any and all type of view-point changes.

Along with above mentioned scenarios, we also want to explore the possibility of using saliency maps with out presented omni-directional system for performing visual place recognition. The saliency maps can be used to only focus on important areas of a particular scene and then the information presented in the map cab be used for perform further matching in the database.

We hope to implement the above mentioned scenarios to create a robust place recognition system that can simultaneously deal with conditional and viewpoint variations.

References

[1] Yognand Balagurunathan, Sanja Antic, Heidi Chen, Matthew Schabath, Yuhua Gu, Hua Wang, Ronald Walker, Robert Gillies, Peirre Massion, and Thomas Atwater. Radiomic analysis for improved lung cancer prediction of indeterminate pulmonary nodules. *American Journal of Respiratory Critical Care Medicine*, 191:A6119, 2015.

[2] Wray Buntine and Andreas Weigend. Bayesian back-propagation. *Complex systems*, 5(6):603–643, 1991.

[3] Nicholas Carlevaris-Bianco, Arash K Ushani, and Ryan M Eustice. University of michigan north campus long-term vision and lidar dataset. *International Journal of Robotics Research (IJRR)*, pages 1–13.

[4] Tsung-Han Chan, Kui Jia, Shenghua Gao, Jiwen Lu, Zinan Zeng, and Yi Ma. Pcanet: A simple deep learning baseline for image classification. *IEEE Transactions on Image Processing*, 24(12):5017–5032, 2015.

[5] Xi Chen, Arpit Jain, Abhinav Gupta, and Larry S Davis. Piecing together the segmentation jigsaw using context. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2001–2008. IEEE, 2011.

[6] Zetao Chen, Obadiah Lam, Adam Jacobson, and Michael Milford. Convolutional neural network-based place recognition. *Australia Conference on Robotics and Automation (ACRA)*, 2014.

[7] Mark Cummins and Paul Newman. Fab-map: Probabilistic localization and mapping in the space of appearance. *International Journal of Robotics Research (IJRR)*, 27(6):647–665, 2008.

[8] Mark Cummins and Paul Newman. Appearance-only slam at large scale with fab-map 2.0. *International Journal of Robotics Research (IJRR)*, 30(9):1100–1123, 2011.

[9] Arun Das, Devinder Kumar, Abdelhamid El Bably, and Steven L Waslander. Taming the north: Multi-camera parallel tracking and mapping in snow-laden environments. In *Field and Service Robotics*, pages 345–359. Springer, 2016.

[10] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(6):1052–1067, 2007.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[12] Andreas Eitel, Jost Tobias Springenberg, Luciano Spinello, Martin Riedmiller, and Wolfram Burgard. Multimodal deep learning for robust rgb-d object recognition. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 681–687. IEEE, 2015.

[13] Felix Endres, Jürgen Hess, Nikolas Engelhard, Jürgen Sturm, Daniel Cremers, and Wolfram Burgard. An evaluation of the rgb-d slam system. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1691–1696. IEEE, 2012.

[14] Ross Finman, Liam Paull, and John J Leonard. Toward object-based place recognition in dense rgb-d maps. In *ICRA Workshop Visual Place Recognition in Changing Environments, Seattle, WA*, 2015.

[15] Ruben Gomez-Ojeda, Manuel Lopez-Antequera, Nicolai Petkov, and Javier Gonzalez-Jimenez. Training a convolutional neural network for appearance-invariant place recognition. *arXiv preprint arXiv:1505.07428*, 2015.

[16] Petr Gronat, Guillaume Obozinski, Josef Sivic, and Tomas Pajdla. Learning and calibrating per-location classifiers for visual place recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 907–914, 2013.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

[19] Toni Hirvonen. Classification of spatial audio location and content using convolutional neural networks. In *Audio Engineering Society Convention 138*. Audio Engineering Society, 2015.

[20] Jarmo Ilonen, Joni-Kristian Kamarainen, and Jouni Lampinen. Differential evolution training algorithm for feed-forward neural networks. *Neural Processing Letters*, 17(1):93–105, 2003.

[21] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Improving bag-of-features for large scale image search. *International Journal of Computer Vision*, 87(3):316–336, 2010.

[22] Tanaka Kanji. Cross-season place recognition using nbnn scene descriptor. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 729–735, 2015.

[23] 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.

[24] Devinder Kumar, Alexander Wong, and David A Clausi. Lung nodule classification using deep features in ct images. *Computer and Robot Vision (CRV), 2015 12th Conference on*, pages 133–138, 2015.

[25] Mathieu Labbé and François Michaud. Online global loop closure detection for large-scale multi-session graph-based slam. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2661–2666. IEEE, 2014.

[26] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[27] Ian Nathaniel Lenz. *Deep Learning For Robotics*. PhD thesis, Cornell University, 2016.

[28] Chris Linegar, Winston Churchill, and Paul Newman. Work smart, not hard: Recalling relevant experiences for vast-scale but time-constrained localisation. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 90–97. IEEE, 2015.

[29] Stephanie Lowry, Niko Sünderhauf, Paul Newman, John J Leonard, David Cox, Peter Corke, and Michael J Milford. Visual place recognition: A survey. *IEEE Transactions on Robotics*, 32(1):1–19, 2016.

[30] Will Maddern, Alex Stewart, Colin McManus, Ben Upcroft, Winston Churchill, and Paul Newman. Illumination invariant imaging: Applications in robust vision-based localisation, mapping and classification for autonomous vehicles. In *Visual Place Recognition in Changing Environments Workshop, IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[31] Michael Milford, Walter Scheirer, Eleonora Vig, Arren Glover, Oliver Baumann, Jason Mattingley, and David Cox. Condition-invariant, top-down visual place recognition. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5571–5577, 2014.

[32] Michael Milford, Chunhua Shen, Stephanie Lowry, Niko Suenderhauf, Sareh Shirazi, Guosheng Lin, Fayao Liu, Edward Pepperell, Cesar Lerma, Ben Upcroft, et al. Sequence searching with deep-learnt depth for condition-and viewpoint-invariant route-based place recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 18–25, 2015.

[33] Michael J Milford and Gordon F Wyeth. Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1643–1649, 2012.

[34] Mahesh Mohan, Dorian Gálvez-López, Claire Monteleoni, and Gabe Sibley. Environment selection and hierarchical place recognition. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5487–5494, 2015.

[35] Marius Muja and David G Lowe. Flann, fast library for approximate nearest neighbors. In *International Conference on Computer Vision Theory and Applications (VISAPP09)*, 2009.

[36] Marius Muja and David G Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2227–2240, 2014.

[37] Raul Mur-Artal, JMM Montiel, and Juan D Tardós. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[38] Peer Neubert, Niko Sunderhauf, and Peter Protzel. Appearance change prediction for long-term navigation across seasons. In *European Conference on Mobile Robots (ECMR)*, pages 198–203, 2013.

[39] David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2161–2168, 2006.

[40] Rohan Paul and Paul Newman. Fab-map 3d: Topological mapping with spatial and visual appearance. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2649–2656. IEEE, 2010.

[41] Edward Pepperell, Peter Corke, and Michael Milford. Towards vision-based pose- and condition-invariant place recognition along routes. In *Australasian Conference on Robotics and Automation(ARAA)*, 2014.

[42] Edward Pepperell, Peter I Corke, and Michael J Milford. All-environment visual place recognition with smart. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1612–1618, 2014.

[43] Edward Pepperell, Peter I Corke, and Michael J Milford. Automatic image scaling for place recognition in changing environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1118–1124, 2015.

[44] Ananth Ranganathan, Shinichi Matsumoto, and David Ilstrup. Towards illumination invariance for visual localization. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3791–3798, 2013.

[45] Stéphane Ross, Narek Melik-Barkhudarov, Kumar Shaurya Shankar, Andreas Wendel, Debadeepta Dey, J Andrew Bagnell, and Martial Hebert. Learning monocular reactive uav control in cluttered natural environments. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 1765–1772. IEEE, 2013.

[46] Reuven Y Rubinstein and Dirk P Kroese. *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning.* Springer Science & Business Media, 2013.

[47] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. IEEE, 2011.

[48] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 806–813, 2014.

[49] Richard Socher, Brody Huval, Bharath Bath, Christopher D Manning, and Andrew Y Ng. Convolutional-recursive deep learning for 3d object classification. In *Advances in Neural Information Processing Systems*, pages 665–673, 2012.

[50] Niko Sünderhauf, Peer Neubert, and Peter Protzel. Are we there yet? challenging seqslam on a 3000 km journey across all four seasons. In *Workshop on Long-Term Autonomy, IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[51] Niko Sünderhauf and Peter Protzel. Brief-gist-closing the loop by simple means. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1234–1241, 2011.

[52] Niko Sunderhauf, Sareh Shirazi, Feras Dayoub, Ben Upcroft, and Michael Milford. On the performance of convnet features for place recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4297–4304, 2015.

[53] Niko Sunderhauf, Sareh Shirazi, Adam Jacobson, Feras Dayoub, Edward Pepperell, Ben Upcroft, and Michael Milford. Place recognition with convnet landmarks: Viewpoint-robust, condition-robust, training-free. *Robotics: Science and Systems (RSS) XII*, 2015.

[54] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[55] Akihiko Torii, Josef Sivic, Tomas Pajdla, and Masatoshi Okutomi. Visual place recognition with repetitive structures. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 883–890, 2013.

[56] Ben Upcroft, Colin McManus, Winston Churchill, Will Maddern, and Paul Newman. Lighting invariant urban street classification. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 1712–1718. IEEE, 2014.

[57] Andrea Vedaldi and Karel Lenc. Matconvnet: Convolutional neural networks for matlab. In *23rd Annual ACM Conference on Multimedia*, pages 689–692, 2015.

[58] Xin Wang, Kumar, Nicolas Thome, Matthieu Cord, and Frederic Precioso. Recipe recognition with large multimodal food dataset. In *Multimedia & Expo Workshops (ICMEW), 2015 IEEE International Conference on*, pages 1–6. IEEE, 2015.

[59] Thomas Whelan, Michael Kaess, Hordur Johannsson, Maurice Fallon, John J Leonard, and John McDonald. Real-time large-scale dense rgb-d slam with volumetric fusion. *The International Journal of Robotics Research*, 34(4-5):598–626, 2015.

[60] Saining Xie, Tianbao Yang, Xiaoyu Wang, and Yuanqing Lin. Hyper-class augmented and regularized deep learning for fine-grained image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2645–2654, 2015.

[61] Dani Yogatama and Noah A Smith. Bayesian optimization of text representations. *arXiv preprint arXiv:1503.00693*, 2015.

[62] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4694–4702, 2015.

[63] Tianhao Zhang, Gregory Kahn, Sergey Levine, and Pieter Abbeel. Learning deep control policies for autonomous aerial vehicles with mpc-guided policy search. *arXiv preprint arXiv:1509.06791*, 2015.